Thèse

présentée pour obtenir le grade de docteur de

l'Ecole nationale Supérieure des Télécommunications

Spécialité : Informatique et Réseaux

Mohammad Ashiqur Rahaman

Processus Agiles d'échange de Document

Modèles, Interoperabilité et Sécurité

soutenue le 30 juine 2010 devant le jury composé de

Président	**Pascal Urien**	**Télécom ParisTech, France**
Rapporteurs	**Joachim Posegga**	**Universtät Passau, Allemagne**
	François Charoy	**Université Henri Poincaré, France**
Examinateurs	**Andreas Schaad**	**SAP AG, Allemagne**
	Abdelmadjid Bouabdallah	**Université de Technologie de Compiègne, France**
Directeur de thèse	**Yves Roudier**	**EURECOM, France**

PhD Thesis

Ecole Nationale Supérieure des Télécommunications

Computer Science and Networks

Mohammad Ashiqur Rahaman

Document-based Agile Workflows Models, Interoperability, and Security

Defense date: June, 30, 2010

Committee in charge:

Chairman	**Pascal Urien**	**Télécom ParisTech, France**
Reporters	**Joachim Posegga**	**Universtät Passau, Germany**
	François Charoy	**Université Henri Poincaré, France**
Examiners	**Andreas Schaad**	**SAP AG, Germany**
	Abdelmadjid Bouabdallah	**Université de Technologie de Compiègne, France**
Thesis Advisor	**Yves Roudier**	**EURECOM, France**

. . . To my parents (A.S.M. Shafiqur Rahman and Mrs. Jahanara Rahman), wife (Sumaiya Haider), and sisters (Ferdousi Rahman Shanta and Farhana Rahman Tithi).

If we knew what we were doing, it wouldn't be called research, would it?
- Albert Einstein -

Acknowledgements

My thesis work having come to an end it is time to look back over the past three years and thank the people without the support of whom conducting my PhD research would not have been possible.

I would like first to express my deep gratitude to my supervisor Yves Roudier and SAP mentor Andreas Schaad. I learned a lot from our discussions and without their support these past three years would not have been successful. I will always be grateful to them.

I would like then to thank the SAP Research team in France for giving me the opportunity to perform my PhD thesis in the best possible conditions. I am especially thankful to the R4eGov team: Andreas Schaad, Cédric Hebert and Henrik Plate. Working with you was a pleasure. Collaborating with the other SAP team members was also inspiring. To name a few Sarath Indrakanti, Khaled Galoul and Philip Miseldine. Many thanks also go to my great fellow PhDians Alessandro Sorniotti, Azzedine Benameur, Joern Franke, Paul El Khoury and Sabir Idrees for their continuous support and to Abdelhakim for his support on the initial implementation.

Last but not least, I would like to thank Pascal Urien, Joachim Posegga, François Charoy, and Abdelmadjid Bouabdallah for being part of the thesis committee.

Funds for this work were provided by SAP Labs France and by the European Commission (FP6 R4eGov project).

Mohammad Ashiqur Rahaman, March 2010

Résumé en Français

Organisations et entreprises doivent faire face à des changements constants dans leur structure ou leurs marchés pour des raisons évidentes de compétitivité. Ceci requiert une agilité obtenue par l'adoption de modèles d'entreprise plus flexibles et basés sur le traitement de l'information.

Les concepts de workflow et de processus métier sont aujourd'hui essentiels aux environnements d'entreprise. La tendance à plus d'agilité nécessite un découplage accru entre la modélisation et l'exécution de ces processus, au regard du workflow classique qui consiste en un enchaînement figé de tâches concrètes. Cette flexibilité entraîne cependant de nouveaux problèmes d'interopérabilité et de sécurité, notamment du fait de la gestion d'échanges inter-entreprises entre acteurs a priori inconnus.

La contribution principale de cette thèse est l'introduction de la notion de workflow de documents agile. Une méthodologie est d'abord présentée qui repose sur une approche déclarative du traitement des documents échangés, organisant et combinant dans un cadre unifié les buts et règles métier stratégiques. L'utilisation de modèles à base d'ontologies pour l'annotation sémantique des données des documents est proposée comme solution pour l'interopérabilité. Cette approche fournit également le mécanisme de base d'une infrastructure de communication décentralisée assurant la distribution des documents entre acteurs d'un workflow de documents agile. Des mécanismes de sécurité sont enfin définis afin d'assurer un contrôle d'accès fin aux documents ainsi que leur intégrité, de même que la gestion dynamique des clés des acteurs du workflow.

I Introduction

Un workflow (littéralement "flux de travail", c'est-à-dire un enchaînement de tâches) permet la modélisation d'un ensemble de tâches à accomplir et des différent acteurs impliqués dans la réalisation d'un processus métier, qui représente les interactions sous forme d'échanges de documents entre divers acteurs tels que: les humains, les applications et services. Il fournit

en outre, à chacun des acteurs, les informations nécessaires pour la réalisation de ces tâches. Les tâches ordinaires tel que: "demander le prix d'un produit" et "vérifier la disponibilité d'un produit dans le stock", et leurs règles de gestions dans "workflow d'achat de produits en ligne" et leur ordre d'exécution sont connus pour un tel workflow comme des tâches qui s'exécutent régulièrement dans le processus métier dans le "workflow d'achat de produits en ligne". Néanmoins, dans le cas des scénarios métier plus évolués, intitulés processus métier agiles, agiles, tels que: le processus de traitement des urgences médicales, de fusion de compagnies, etc., "il peut être difficile de déterminer les tâches appropriées ou leur ordre d'exécution à cause des particularités de ces scénarios". Dans ce contexte, les approches de modélisation de processus métier agile [Fea, ABP, BPT, BAA, BPS, WW05] ont récemment gagné de l'attention au point de vue stratégique. Du point de vue technique, les workflows flexibles sont plus appropriés pour implémenter de tels scénarios [AB00, ADO00, Tag01, MPvdA07, vdAP06, vdAW05, vdAJ00]. L'agilité nécessite plus de flexibilité dans la modélisation de processus métier et dans son exécution, parmi d'autres besoins [BBB06a, MSB08]. Cependant, cette flexibilité implique de nouvelles stratégies d'interopérabilité et de sécurité [DBL08, SYY$^+$08, Dog97, SMLP05] et elle introduit de nouveaux défis pour la recherche dans les systèmes de gestion de workflow pour assurer l'interopérabilité, un faible couplage et la sécurité de la distribution des documents entre acteurs. Permettre une exécution flexible et distribuée d'un workflow dans un cadre agile n'est pas bien compris dans les propositions actuelles ou se paie au prix fort, par exemple en perdant des interfaces stables pour l'échange de données entre acteurs d'un système B2B par exemple, ainsi que de fréquentes interactions avec un back-end [vdAW05]. Le système de workflow agile à base de document proposé dans cette thèse est un système flexible qui répond aux problèmes mentionnés ci-dessus.

Au cours des années, les méthodes et technologies de workflow ont évolué à partir des solutions à base de papier aux workflows basés sur des services web comme en témoignent des solutions produites par des acteurs industriels majeurs comme Microsoft [Hola], IBM [IBMa], SAP [SAPb]. Ces solutions peuvent être classifiées en deux classes:

- **Workflows à base de tâches:** L'ensemble de tâches métier et leur ordre d'exécution, ainsi que leurs règles de gestion associées permettent de modéliser un workflow à base de tâches. Cette approche est convenable pour la modélisation des processus métier ordinaires, et elle est la plus dominante parmi les vendeurs logiciels IBM [IBMa], SAP [SAPb], Microsoft [Hola], et Tibco [Holb].

- **Workflows flexibles:** Un ensemble prédéfini de tâches métier organisées d'une façon flexible, au niveau de leur ordre d'exécution ainsi que dans l'application des règles métier qui leurs sont associées. Les composants exécutables dans ce genre de workflow peuvent être liés d'une manière dynamique et flexible. Les exemples d'un tel type de workflow sont: les workflows déclaratifs [MPvdA07, vdAP06] et les workflows à base d'état [vdAW05] etc. qui demeurent cependant de l'ordre de la recherche notamment académique.

Aucun des types de workflows mentionnés ci-dessus ne considère les aspects de flexibilité, par exemple lorsque les tâches et les acteurs métier ne sont pas connus avant, de plus les solutions existantes sont limitées pour permettre la modélisation et l'exécution un workflow agile. Par exemple: elles ne supportent pas la sélection des tâches métier appropriées à l'exécution, et la liaison flexible et dynamique vers des composants appropriés selon leur état en temps-réel

(services Web, EJBs, etc. La solution dans ce contexte, est d'assurer suffisamment de flexibilité durant la phase de conception lorsque les tâches métier appropriées sont déterminées, et durant l'exécution quand une tâche métier est liée à un composant (e.g., Web Services, EJBs).

I.1 Méthodologie pour les Processus Agiles d'échanges de Documents

Notre objectif est de fournir plus de flexibilité pour l'acteur d'un workflow à base de document (*DocWF*) durant la conception et l'exécution de *DocWF*. Notre approche combine la technique de modélisation sémantique de processus métier et la liaison flexible vers des composants réutilisables (voir la Figure 1). Les deux étapes de processus sont:

1. **Détermination de tâches métier:** Déterminer les tâches métier appropriées en exécutant le processus de correspondance entre l'objectif de processus métier concerné et les annotations des tâches du modèle BPMN dans un répertoire organisé (i.e., une base de connaissances (*KB*)).

2. **Liaison avec les services:** Exécuter le même processus de correspondance par le service de découverte basé sur la sémantique pour lier une tâche vers des services concrets.

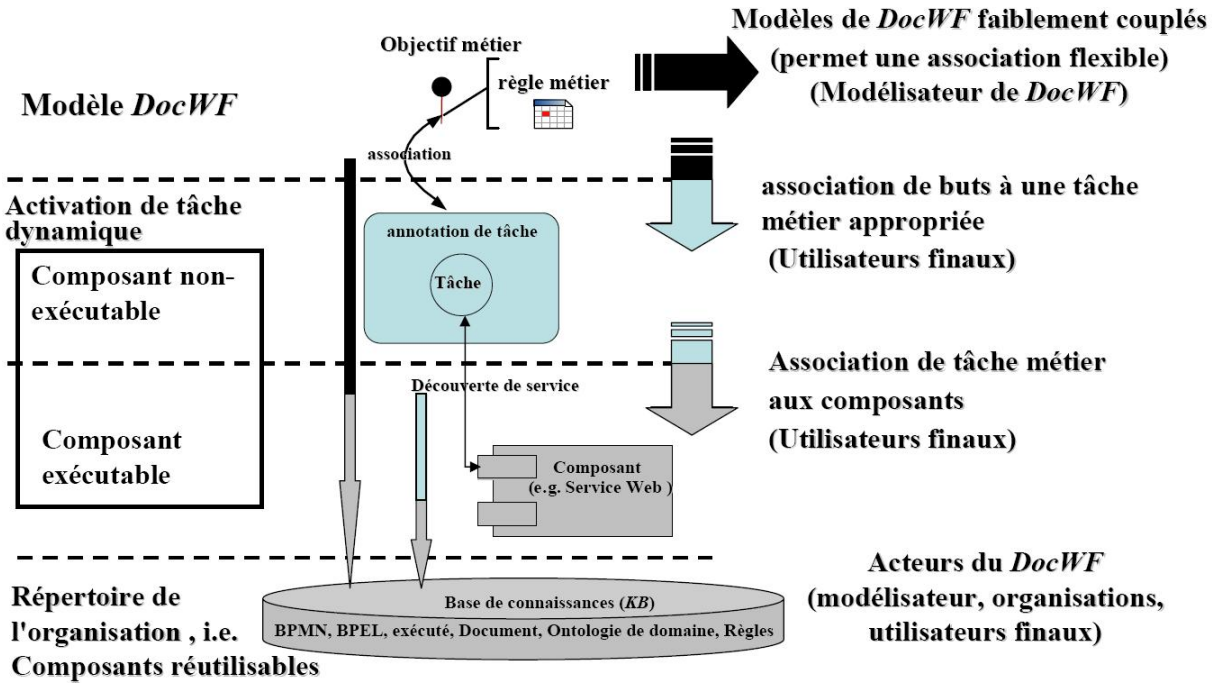


Figure 1: Association flexible de modèles de workflow de documents agile (*DocWF*) à la conception à des composants exécutables.

Les concepteurs d'un workflow à base de document (*DocWF*) peuvent spécifier les objectifs métier d'un workflow agile sous une forme abstraite. Ils peuvent spécifier les règles métier de gestion associées aux objectifs métier. Ces règles spécifient les contraintes sur les documents et les objectifs métier. Les modèles de processus sont systématiquement attachés à des

composants exécutables leur de l'exécution. Pour déterminer la future liaison entre les tâches et les services (i.e., composants), nous utilisons des techniques de correspondance sémantique dans la base de connaissances du pair concerné. La base de connaissances (knowledge-base) contient la description sémantique des processus métier des organisations (e.g., BPMN sémantique) et la description du processus métier exécutable (e.g., BPEL), ce qui permet d'effectuer des recherches par une chaine de caractères qui correspond aux objectifs d'un processus métier par exemple avec les annotations des tâches BPMN et des annotations des tâches avec de la sémantique des web services.

I.2 Besoins pour l'interopérabilité et la Sécurité

La conception d'un système de workflow de documents agile (i.e., *DocWF*) ne doit pas seulement considérer les besoins fonctionnels mais doit aussi prendre en compte les besoins d'interopérabilité et de sécurité suscités par les scénarios associés.

I.2.1 Interoperabilité

Les workflows agiles ne peuvent pas être modélisés avant de spécifier les tâches métier. Dans cette thèse, nous introduisons des notions de haut-niveau d'abstraction dans le processus de modélisation tel que les objectifs métier d'un workflow et les règles de gestion métier qui peuvent êtres renseignés par les acteurs concernés. De tels modèles de processus étant proches du niveau d'expression de la stratégie métier, ils sont extrêmement faiblement couplés avec les tâches métiers effectives et les composants exécutables. L'exécution de tels workflows peut être déclenchée par la réception d'une donnée envoyée par un autre acteur (peut être un autre workflow) qui représente une des extrémités du workflow. Etant donné ceci, la modélisation des données d'un workflow basé sur la sémantique est nécessaire pour l'interprétation sémantique des échanges de données entre les extrémités (acteurs). Finalement, l'infrastructure de communication a aussi besoin de la sémantique pour supporter le découplage faible d'échange de données entre les acteurs du workflow a priori inconnus.

I.2.2 Sécurité Informatique

Les documents sont la seule interface entre les acteurs, à priori, inconnus où les acteurs concernés produisent des documents dont la structure est très détaillée qui ont besoin d'êtres distribués vers des paires appropriés. Les producteurs de documents ne peuvent pas distribuer durant la production car les consommateurs légitimes ne sont pas connus avant l'exécution. De plus, les nœds légitimes de documents qui sont sémantiquement liés peuvent êtres issus de plusieurs producteurs différents. L'infrastructure de communication permet la distribution de document sélective en offrant un découplage faible. Comme tout système complètement décentralisé, l'exécution du workflow *DocWF* a besoin lui-aussi de solutions appropriées concernant la sécurité. Contrairement au workflow distribué pour les systèmes de gestion où la tâche d'authentification est le problème principal lié à la sécurité, l'exécution du workflow *DocWF* introduit des nouveaux défis concernant la sécurité par rapport à l'infrastructure de communication car le découplage faible de la communication entre les acteurs est réalisée uniquement par des échanges des documents. La sécurité des documents inclut, par exemple, la protection

de l'intégrité des documents, la mise en IJuvre d'autorisations à grain fin sur les documents et leur vérification a posteriori.

Le chiffrement des documents peut être utilisé comme technique pour mettre en IJuvre des droits à grain fin sur les documents. Cependant, le principal défi est le calcul d'un groupe de clefs associé à un *concept* d'ontologie en limitant les interruptions de l'édition du document dues à des synchronisations dans le *DocWF*. De plus, tracer les actions d'acteurs a priori inconnus est aussi délicat. La plupart des workflows distribués des systèmes de gestion sont généralement dédiés, soit à la gestion des droits dŠaccès au niveau des tâches métier [LZS09, ACM01, CLW05, KPF01] soit à garantir les exécutions de séries de tâches métier prédéfinies [MM07].

II Interoperabilité des Processus d'échanges de Documents Agile

Afin de permettre l'interopérabilité à travers les frontières organisationnelles lors de l'échanges de données sous forme de documents, le partage public d'une ontologie de domaine s'avère nécessaire. Une ontologie décrit la sémantique des données du worflow associées aux scénarios agiles dans une forme syntaxique neutre, comme OWL [OWL], et constitue ainsi une interface pour l'échange de données entre acteurs d'un *DocWF*.

Une telle interface sémantique permet à un acteur de changer son modèle de données quand nécessaire sans affecter les interfaces existantes et de transmettre des descriptions de ses données. Comme mentionné, l'ontologie est la seule interface indispensable entre acteurs d'un *DocWF*, alors même que ceux-ci ne se connaissent pas nécessairement. Les acteurs d'un *DocWF* produisant des documents de structure détaillée avec des annotations appropriées, ceux-ci pourront être distribués à des consommateurs de manière sélective, d'après les droits obtenus sur la description sémantique des documents. Une telle forme d'échange faiblement couplée entre acteurs sera assistée par une infrastructure de communication de nœds spécialisés.

Afin d'assurer la distribution efficace des documents, ces nœds sont déployés selon une architecture hiérarchique basée sur l'ontologie choisie dans laquelle les nœds partagent la charge de distribuer des portions des documents annotées de manière appropriée aux acteurs intéressés.

III Sécurité du Processus d'échange de Documents Agile

Nous présentons des solutions pour assurer la sécurité de l'exécution d'un *DocWF*. Ces solutions utilisent des mécanismes de contrôle d'accès distribué à grain fin pour des documents "XML d'entreprise". Ces mécanismes s'appuient sur la technique cryptographique de groupe à base d'arbre de Diffie-Hellman (TGDH) [KPT00]. Nous adaptons cette technique pour permettre à un groupe d'acteurs avec les mêmes intérêts pour un concept sémantique de calculer indépendamment une clé de groupe associée à ce *concept*. En particulier, cette technique vise à controler l'accès d'un groupe d'acteurs possédant les mêmes droits à une portion de document par son chiffrement avec cette clé calculée pour ce groupe. Le calcul de la clé permet à un acteur légitime de chiffrer/déchiffrer les documents de manière autonome et fine et interdit à des acteurs malveillants d'effectuer des accès non autorisés comme par exemple une modifica-

tion, destruction, insertion ou même déplacement de nœds non souhaitée. La version de TGDH adaptée contraint de plus les possibilités de génération d'une nouvelle clé quand de nouveaux acteurs se joignent au group ou quittent l'environnement afin d'assurer l'échange de documents de manière non-disruptive pour une période de temps étendue pour l'exécution d'un *DocWF*. La traçabilité est aussi permise pour une vérification a posteriori grace à ce mécanisme combiné à des annotations de sécurité particulières.

IV Organisation et Contribution de la Thèse

Chaque chapitre de cette thèse correspond à un bloc de construction d'un système de workflow de documents agile ou de son environnement d'exécution. Les quatre principaux composants mentionnés dans cette section sont décrits Figure 2. Le composant central est le système de workflow qui doit remplir les besoins fonctionnels attendus dans les scénarios d'échange de documents agile. L'interopérabilité entre organisation et le faible couplage des documents est assuré par une technique de modélisation des documents à base d'ontologie, qui permet d'effectuer des annotations sémantiques sur les documents échangés. Elle repose aussi sur des algorithmes de comparaison de documents afin d'assurer la convergence des documents, besoin indentifié dans le chapitre introductif comme crucial pour le système de *DocWF*. D'autre part, le système peut aussi utiliser une infrastructure de communication distribuée par routage et acheminement sélectif des documents aux acteurs légitimes sur la base de politiques de contrôle d'accès définies par les producteurs de documents. Finalement, le bloc de construction lié à la sécurité concerne différents besoins de sécurité. Le reste de ce manuscrit est décrit en détail dans la suite.

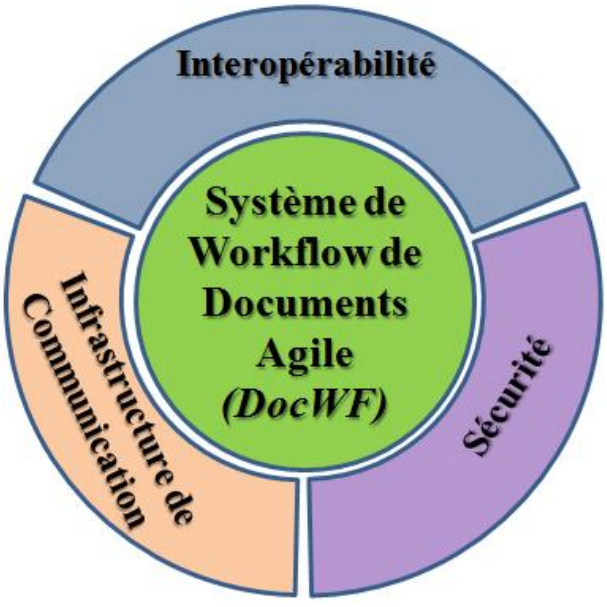


Figure 2: Organisation de la thèse en termes de blocs de construction de base.

Chapitre 2 Préliminaires et Survol Technique:

Dans ce chapitre, nous introduisons des notions de base utilisées tout au long de la thèse. En particulier, nous introduisons les bases des workflows à base de tâches, des workflows de documents et des workflows de documents agiles, et nous discutons de leur rapport avec les processus métier. Nous discutons ensuite brièvement des infrastructures de communication, en particulier de type publication/souscription, en rapport avec l'infrastructure décentralisée suggérée dans cette thèse. Nous discutons enfin le problème du traitement de grands volumes de documents métier et des problèmes de sécurité avant de conclure.

Les Chapitres 3, 4, 5, et 6 constitutent le cœur de cette thèse et décrivent les solutions que nous avons conçues pour atteindre l'objectif fixé dans ce chapitre introductif. Des scénarios illustratifs séparés sont introduits dans différents chapitres pour motiver chaque solution proposée. Chaque chapitre contient des références croisées aux chapitres précédents pour indiquer des notions de base présentées précédemment et nécessaires.

Chapitre 3 Workflows de Documents Agiles:

Nous nous intéressons dans ce chapitre à l'approche de modélisation à faible couplage d'un workflow de documents agile et au principe de son exécution. Cette approche de modélisation s'appuie sur un vocabulaire adapté tel les objectifs métiers et règles métier associées. Les principes d'exécution d'applications structurées de la sorte s'appuient sur l'activation dynamique de tâches, par exemple suite à la satisfaction de buts décrits par une sémantique de haut niveau et au travers de techniques de découverte de services sémantique. Les règles métier, qui capturent différents concepts tels que la dépendance des objectifs et le traitement en cas de contenu adapté, peuvent être formellement représentées par des automates d'états finis. Ces représentations peuvent être utilisées à l'exécution afin de détecter des erreurs de modélisation.

Chapitre 4 Interopérabilité des Workflows de Documents Agiles:

Dans ce chapitre, nous nous attaquons au problème de l'interopérabilité entre instances de workflows de documents agiles et proposons une interface de documents métiers basée sur l'utilisation d'une ontologie partagée par les acteurs en présence afin d'assurer une représentation stable mais flexible. La stabilité provient du fait que l'ontologie décrivant la sémantique des différentes parties d'un document change peu et est indépendante des modèles de données syntaxiques sous-jacents. La flexibilité des applications est accrue car les modèles de données individuels peuvent évoluer de manière indépendante sans affecter l'interface sémantique au niveau de laquelle les politiques peuvent aussi être spécifiées, même à grain fin, alors que leur application s'effectue au niveau syntaxique de XML. Des annotations sémantiques spécialisées sont possibles permettant à un destinataire appropriée de comprendre le contenu du document même si sa connaissance locale est basée sur un vocabilaire différent. Finalement, la gestion des documents de manière distribuée repose aussi sur ces principes et sur des algorithmes de comparaison des éditions effectuées sur un document développés dans ce chapitre.

Chapitre 5 Infrastructure de Communication pour les Workflows de Documents Agiles:

Dans ce chapitre, nous présentons l'infrastructure de communication décentralisée et les protocoles de communication associés pour permettre des échanges de documents faiblement couplés, L'infrastructure se base sur une méthode de publication/souscription sémantique afin d'assurer une distribution sélective des documents durant l'exécution d'une application basé sur un workflow de documents agile. La conception de l'infrastructure suggérée est fortement liée avec l'ontologie du domaine métier sous-jacent afin de faciliter l'intégration des interfaces de données définies au Chapitre 4.

Chapitre 6 Sécurité des Workflows de Documents Agiles:

Dans ce chapitre, nous présentons des solutions de sécurité pour permettre l'exécution de workflows de documents agiles dans des environnements malveillants. Ces solutions décrivent des mécanismes de contrôle d'accès pour les documents XML structurés avec une granularité fine. Ces mécanismes s'appuient sur la technique TGDH (tree-based group Diffie-Hellman) [KPT00]. Nous adaptons cette technique pour permettre à un groupe d'acteurs de calculer la clé de groupe indépendamment, chacun des membres du groupe partageant les mêmes droits d'accès à un document, ces droits étant définis au niveau sémantique mais mis en application au niveau syntaxique par chiffrement de nœds. La conception des mécanismes suggérés est fortement couplée avec les politiques sémantiques spécifiées au Chapitre 4 et à la distribution sélective de documents du Chapitre 5.

Annexes: L'Annexe A fournit des détails sur l'implémentation de l'infrastructure de communication et l'Annexe B décrit certaines bibliothèques de sécurité implantées. Enfin, l'Annexe C illustre l'formelle workflows de documents agiles du Chapitre 3 avec des règles de transition. Le lecteur est invité à lire les annexes dans l'ordre.

Le travail de recherche effectué par l'auteur dans le cadre de cette thèse a conduit à un certain nombre de publications scientifiques [RRS, RRS09a, RRS09c, RRS09b, RRMS09, RRS08, RS07a, RSR06, RMS06] et de brevets industriels dont les idées essentielles sont présentées dans ce manuscrit.

V Conclusion

Nous avons discuté dans cette thèse la conception d'un système de workflow de documents agile pour les processus métier agiles. Nous avons spécifié les modèles de workflows indépendants de toute tâche métier basés sur les buts métier et des règles logiques associées. Les acteurs d'un workflow déterminent proactivement les tâches appropriées et quels composants peuvent les remplir, tels que des Services Web, EJBs ou des humains. Nous avons fourni des solutions sémantiques pour assurer l'interopérabilité des échanges de documents avec des partenaires initialement inconnus. Nous avons conçu une infrastructure de communication décentralisée

permettant la distribution sélective de documents comportant des annotations sémantiques et offrant un support aux workflows décrits précédemment. Nous avons aussi décrit des solutions de sécurité offrant les garanties requises pour l'exécution d'un tel système et tenant compte des vulnérabilités de l'infrastructure de communication. Enfin, nos solutions d'interopérabilité et de sécurité sont complètement modulaires, ce qui rend possible de déployer des systèmes de *DocWF* légers adaptés pour un scénario particulier, en fonction du caractère plus ou moins critique de chaque élement du système.

Différentes directions de recherches futures peuvent être envisagées au vu des résultats présentés dans cette thèse. Par exemple, le processus de détermination d'une tâche métier à l'exécution décrit est limité à la correspondance d'une simple chaîne de caractères avec des annotations de tâches BPMN. Ce processus pourrait intégrer des mécanismes de mise en correspondance plus sophistiqués tels que des combinaisons de buts, ou l'utilisation de pré- ou post-conditions. En ce qui concerne la liaison entre tâches métiers et composants, des mécanismes de substitution de services sémantiques [FGIZ08] pourraient être introduits afin de fournir une meilleurs précision de l'adaptation du comportement aux services disponibles. Pour les scénarios critiques, l'exécution conforme avec une séquence de tâches pré-spécifiée est souvent requise par le cadre législatif sur les processus d'entreprise. Une telle assurance ne peut cependant être déterministe dans le cadre d'un *DocWF* puisque il n'y a pas d'exécution connue à l'avance, mais des solutions basées sur les buts métier pourraient être trouvées.

Abstract

Today's business organizations must deal with constant changes such as organizational structure changes and market changes. These changes drive the rapid evolution of their workflows or business processes that allow them to be competitive. This required agility demands organizations to increasingly adopt flexible and information processing oriented business models. The concepts of flexible workflows or agile business processes and their underlying technologies such as SOA and semantic web have become the main enablers to implement agile business scenarios [BPS, WW05]. The trend of agility demands more flexibility in business process modeling and its execution, among other needs [BBB06a, MSB08]. For instance, business process models can be made more loosely coupled with their actual executions. This can be done by integrating strategy level notions, such as, business goals and business rules during modeling of business processes as opposed to modeling by a plain sequence of concrete business tasks [EAB$^+$03, HMNS] and thus making the business process models independent of actual business tasks and their later executions. Determination of suitable business tasks and their corresponding binding to concrete services for later executions can also be performed at runtime based on contextual information as captured in business documents. Such concrete services can be realized by suitable IT components such as Web Services. This flexibility of loosely coupled process models enables information structures of interacting actors to evolve while being interoperable with other distributed actors. This flexibility however comes at the expense of interoperability and security [DBL08, SYY$^+$08, Dog97, SMLP05] and introduces new research challenges. As opposed to usual workflow management systems these challenges include supporting interoperability between a priori unknown actors and ensuring secure document exchanges between them during an execution.

The main contribution of the thesis is the design of a document-based agile workflow system by specifying design and run time principles [RRS09a] and appropriate solutions to enable interoperability [RRMS09, RRS09b]. A communication infrastructure [RRS09c, RRS] and security solutions [RRMS09, RRS08] to implement agile business processes are equally provided. The design and run time principles that we developed are captured in a special modeling approach for document-based agile workflows. Such a modeling approach relies on a declarative [MPvdA07, vdAP06] technique in order to organize and combine various strategic business concerns such as business goals, business rules associated to processing data in the business documents in a unified framework. Regarding interoperability, the proposed document-based agile workflow system includes an ontology-based approach for data representation in documents and their semantic annotations to enable nondisruptive document exchanges between actors. In order to support a loosely coupled and selective document distribution between actors during an execution of a document-based agile workflow, it also includes an ontology-driven decentralized communication infrastructure. To this end, security solutions associated with documents,

such as, fine-grained document access control and document integrity and associated with the communication infrastructure, such as, key management, are developed to ensure secure document exchanges between a priori unknown actors during an execution of a document-based agile workflow.

Contents

List of Figures

Acronyms and Notations

Acronyms

BPEL	Business Process Execution Language
BPMN	Business Process Modeling Notation
CDL	Choreography Description Language
concept	An Ontology Class Representing a Business Domain Concept
DocWF	Document-based Agile Workflow Application/Agile Workflow Application
DOM	Document Object Model
ESB	Enterprise Service Bus
HTTP	Hypertext Transfer Protocol
JDOM	JAVA Document Object Model
LTL	Linear Temporal Logic
OWL	Ontology Web Language
RDF	Resource Description Framework
SAX	Simple API for XML
sBPMN	Semantic Business Process Modeling Notation
SMTP	Simple Mail Transfer Protocol
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SPARQL	Simple Protocol and RDF Query Language
UML	Unified Modeling Language
URI	Uniform Resource Identifier
UDDI	Universal Description, Discovery and Integration
WSDL	Web Service Definition Language
XML	Extensible Markup Language
XACML	Extensible Access Control Markup Language

Mathematical notations

$\mathcal{M}$	Message space
$\mathcal{K}$	Key space
H, h_1	Cryptographic hash function
PK_c^i	Public Diffie Hellman (DH) key associated to a business domain ontology *concept*, c, of peer P_i
SK_c^i	Private DH key associated to a business domain ontology *concept*, c, of peer P_i
CK_c	Group secret key associated to a business domain ontology *concept*, c
m_K	Encryption using a key K on a message m
$[m]_{SK}$	Signature using a private key SK on a message m
$(m_1 \mid m_2)$	Concatenation of messages m_1 and m_2
$MS_{P_i}(m)$	Merkle signature by peer, P_i, using its private DH key, SK_c^i, on a message m
$MS_{P_i}(d, \hat{d})$	Merkle hash path of peer, P_i. The Merkle hash path is associated to an updated document, d, with respect to the root of that document, i.e., $\hat{d}$.

Publications based on this Thesis

Here is the list of publications that have been written and published in several international conferences, workshops and journals since the beginning of this Ph.D. thesis.

International Conferences

[RRS09c] "**Towards Secure Content Based Dissemination of XML Documents**" [1] [2]
Rahaman, Mohammad Ashiqur and Roudier, Yves and Schaad, Andreas, in the proceedings of the Fifth International Conference on Information Assurance and Security (IAS 09), August 18-20, 2009, X'ian, China.

[RRS09b] "**A Secure Comparison Technique for Tree Structured Data**" [1] [2]
Rahaman, Mohammad Ashiqur and Roudier, Yves and Schaad, Andreas, The Fourth International Conference on Internet and Web Applications and Services (ICIW 2009), May 24-28, 2009 - Venice/Mestre, Italy.

[RRMS09] "**Ontology-based Secure XML Content Distribution**"[1] [2]
Rahaman, Mohammad Ashiqur and Roudier, Yves and Schaad, Andreas, IFIP SEC 2009, 24th International Information Security Conference, May 18-20, 2009, Pafos, Cyprus.

[RRS08] "**Distributed Access Control For XML Document Centric Collaborations**"[1] [2]
Rahaman, Mohammad Ashiqur and Roudier, Yves and Schaad, Andreas, EDOC '08: Proceedings of the 2008 12th International IEEE Enterprise Distributed Object Computing Conference.

[RS07a] "**SOAP-based Secure Conversation and Collaboration**"[1] [2] [3]
Rahaman, Mohammad Ashiqur and Schaad, Andreas, ICWS '07: Proceedings of the International Conference on Web Services 2007, IEEE Computer Society.

[RMS06] "**An Inline Approach for Secure SOAP Requests and Early Validation**" [1] [2] [3]
Rahaman, Mohammad Ashiqur and Marten, Rits and Schaad, Andreas, OWASP 2006. European Conference on Open Web Application Security Project. May 30-31, 2006, Leuven, Belgium.

International Workshops

[RRS09a] "**Document-based Dynamic Workflows: Towards Flexible and Stateful Services**" [1] [2]
Rahaman, Mohammad Ashiqur and Roudier, Yves and Schaad, Andreas, in the proceedings of SCC 2009, 2009 IEEE International Conference on Services Computing, September 21-25, 2009, Bangalore, India.

[RSR06] "**Towards Secure SOAP Message Exchange in a SOA**"[1] [2] [3]
Rahaman, Mohammad Ashiqur and Schaad, Andreas and Marteen, Rits, SWS '06: Proceedings of the 3rd ACM workshop on Secure web services.

International Journals

[RRS] "**A Publish/Subscribe Model for Secure Content-Driven XML Dissemination**" [2]

Rahaman, Mohammad Ashiqur and Roudier, Yves and Schaad, Andreas, in the special issue of Journal of Information Assurance and Security (JIAS), volume 6, http://www.mirlabs.org/jias/.

[1] The paper was presented in a conference by the author of this dissertation
[2] The main author of the paper is the author of this dissertation
[3] Some related work on SOAP message level security by the author of this dissertation

Patents based on this Thesis

Here is the list of patents that have been written since the beginning of this Ph.D. thesis.

1. **"Methods of Preventing XML Rewriting Attacks Against SOAP Messages".**
   Mohammad Ashiqur Rahaman and Maarten Rits. 2006P00195EP-Application Number:06290879.3, Publication Number:1860587, Grant Number: 1860587
   2006P00195US-Publication Number:US-2007-0277225-A1

2. **"Secure Time-based Auction Protocol".**
   Mohammad Ashiqur Rahaman and Maarten Rits.
   2006P00554EP-Application Number:06291956.8, Publication Number:1933266
   2006P00554US-Publication Number:US-2008-0167983-A1

3. **"A System and Method for Fully Decentralized Fine-grained Access Control on XML Data Structures".**
   Mohammad Ashiqur Rahaman, Andreas Schaad and Yves Roudier.
   2008P00330US-Application Number:12/339,026

4. **"A Memory Efficient and Secure Enterprise XML Processing."**
   Mohammad Ashiqur Rahaman, Yves Roudier, Andreas Schaad and Henrik Plate.
   2010P00022US-Application Number:12/757,477

5. **"A Comparing Method for Large Sanitized XML Trees."**
   Mohammad Ashiqur Rahaman, Yves Roudier, Andreas Schaad and Henrik Plate.
   2010P00022US-Application Number:12/757,477

6. **"Agile Workflow Modeling and Execution based on Document"**
   Mohammad Ashiqur Rahaman, Yves Roudier and Andreas Schaad.
   2010P00133US-Application Number:12/844,508

Chapter 1

Introduction

It is not the smartest nor the strongest that survive - but those most adaptive to change.
- Charles Darwin -

Workflows or business processes aim at automating routine business tasks that trigger exchanges of business documents amongst business actors. Such an actor can be a human or a machine within or outside of an organization which processes those documents when performing business tasks during an execution of a business process. Actors must comply with a defined set of business rules of their organizations during an execution to reach or to contribute to an over all business goal [Holc]. Routine tasks, for instance "request for a sales quote" and "check stock availability", their associated business rules in a "sales order workflow" and their execution sequence flow are known a priori for that workflow as both tasks are performed regularly in a day-to-day sales order process. However in constantly evolving business scenarios, termed as agile business processes, such as dealing with emergency patients, merging and acquisition of companies, cross-border crime and even completely new business scenarios one may not know suitable tasks and their corresponding sequence flow a priori due to individual peculiarities of the scenarios. In this context, agile business process modeling approaches [Fea, ABP, BPT, BAA, BPS, WW05] recently gained attention from a strategy level point of view. From a technical point of view, flexible workflows are argued to be appropriate to implement such scenarios as shown in [AB00, ADO00, Tag01, MPvdA07, vdAP06, vdAW05, vdAJ00]. The trend of agility demands more flexibility in business process modeling and its execution, among other needs [BBB06a, MSB08]. For instance, business process modeling can be made more loosely coupled with its runtime task enactment. In particular, integrating strategy level notions, such as, business goals and business rules during modeling of business processes as opposed to mod-

eling by a plain sequence of concrete business tasks [EAB+03, HMNS] and thus making the business process models independent of actual business tasks and their later executions. Determination of suitable business tasks and their corresponding binding to concrete services for later executions can also be performed at runtime based on contextual information as captured in business documents. This flexibility however comes at the expense of interoperability and security [DBL08, SYY+08, Dog97, SMLP05] and introduces new research challenges as opposed to usual workflow management systems in terms of supporting interoperability between actors and ensuring a loosely coupled and secure document distribution between them. Typical task-based workflows aim at automating routine business tasks that are modeled and executed in either centralized or distributed fashion but within trusted business boundaries. As a result enabling flexible and distributed workflow execution in an agile setting is not well understood or comes at the cost of significant disruptions, for instance, by loosing stable data exchange interfaces between actors of B2B for instance, and frequent back end actions [vdAW05]. The proposed document-based agile workflow system in this thesis is a flexible workflow system that addresses above mentioned issues.

The main contribution of the thesis is the design of a document-based agile workflow system by specifying design and run time principles [RRS09a] and appropriate solutions to enable interoperability [RRMS09, RRS09b], communication infrastructure [RRS09c, RRS] and security [RRMS09, RRS08] to implement agile business processes. The design and run time principles are captured in a special modeling approach for document-based agile workflows. Such a modeling approach relies on a declarative [MPvdA07, vdAP06] technique in order to organize and combine various strategic business concerns such as business goals, business rules associated to processing data in the business documents in a unified framework. Regarding interoperability, the document-based agile workflow system includes an ontology-based approach for data representation in documents and their semantic annotations to enable nondisruptive document exchanges between distributed actors during an execution. Regarding security, actors executing a document-based agile workflow are operating from distinct business boundaries having distinct business goals, associated rules and information structure. Information structure models of an actor need to be protected from unauthorized disclosure when exchanging documents as they may represent valuable information. This includes, for instance, business strategy, plan and financial status etc. and yet document exchanges amongst peer actors must continue non-disruptively. To facilitate such document exchanges an ontology driven Publish/Subscribe-based decentralized communication infrastructure is also developed as part of our proposed document-based agile workflow system. The Publish/Subscribe-based communication infrastructure supports an execution of a document-based agile workflow by distributing encrypted fine-grained document nodes in a selective fashion that are published by concerned actors. The introduction of such a communication infrastructure together with a priori unknown business actors from varying boundaries make the document-based agile workflow system distributed. Such a distributed system raises several document security challenges, for instance, protection of fine grained document integrity with respect to document semantics and structure, and protection of a trace of document updates (representing a series of editions on a document) from adversaries performing illegitimate inclusions and truncations. In addition, key management required for computing encryption/decryption keys with respect to document authorization policies of the actors in this distributed setting is indeed nontrivial. In this thesis, we also developed

appropriate security solutions addressing aforementioned security concerns to assure secure document exchanges amongst a priori unknown peers during an execution of a document-based agile workflow.

The remainder of this introductory chapter is organized as follows. We first provide an overview of the typical task-based workflows and business processes on which most business applications rely on before introducing a document-based agile workflow. The overview includes a comparison with computer supported cooperative work (CSCW), communication infrastructure solutions with respect to document-based agile workflows and XML-based document interoperability issues to position our contribution in that context. We further motivate the necessity of an ontology-driven decentralized communication infrastructure to support document-based agile workflow executions for offering adequate flexibility required by agile business scenarios. We then explain our choice of relying on Semantic Web and Service Oriented Architecture (SOA) paradigms to implement such agile applications. We finally outline requirements in terms of modeling, interoperability and security introduced by the unusual modeling and execution of such applications in a distributed setting, before highlighting the structure and contributions of this thesis.

1 Research Context

Organizational processes can be classified into three main types: material processes, information processes and business processes as categorized by Medina-Mora and Winograd [RMM93]. Material processes deal with assembly of physical components to products. Information processes involve business tasks performed by computers and humans to create, process and provide information. Business processes are market driven descriptions of activities of a company to fulfill a certain customer service or to reach a business goal. A business process can be realized either as a material process or as an information process and therefore focuses on a higher strategy level.

Our focus in this thesis will be on business processes that are realized as information processes within an agile scenario and are called document-based agile workflow (*DocWF*) applications. An agile business scenario has following basic features:

- **Collaborative:** The scenario is collaborative and goal driven. It involves frequent interactions amongst a priori unknown business actors to archive their business goals.
- **Peculiarities:** Each scenario has individual peculiarities and thus specific rules (not necessarily static) to be applied and therefore can not be generalized.
- **No Task-based Specification:** The scenario can not be described by a specific set of business tasks as suitable business tasks and their sequence flow are unknown or may not be determined a priori.

The workflow methodology is tightly associated to business processes, and is a standard methodology to implement processes. Among many use cases of information processes, a common use is to generate processed documents to report data to higher management. During an execution of a business process, information flows through systems. For instance, in a trade company, information about sales, stocks, purchases and audits are processed, and results in controlling data for management. Another type of system handles sets of related data. Examples prevail everywhere in the society today, ranging from application forms, for insurance, claims, obtaining visas, leave approval, opening bank accounts etc. In this context, a document is a collection of related data, often accompanied by some meta data about the document. This data is typically entered by business actors of the application. Examples of some meta data are: document content type, security level of the document, the author of the document, the date on which the document was created, a summary of the document, version of the document and so on. The complete set of information entered by all actors of an application can be large and composite. Often, multiple actors involved in a document, for example, document creators (who enters the initial values for the document attributes). Other actors might have to review or edit the data. All such actors or peers work on the document and each of them may have certain access rights for the document.

The class of *DocWF* applications we focus on therefore considers these various kinds of documents, involved actors and their processing. Such agile workflow applications thus share certain characteristics with respect to documents and actors as follows:

1. **Processing semantically related data:** They handle a set of semantically related data, captured in a document, which typically contains data entered and shared by multiple actors.

2. **Business rules for processing documents:** Any processing over documents is governed by some business rules which are applied on documents at run time to reach some business goals.

3. **Large documents:** Applications deal with large and composite documents where different portions of a document may originate from varying business boundaries.

4. **Interactions by document exchanges:** Actors operating from distinct business boundaries interact by exchanging documents among them and thus the application is distributed.

5. **Varying business boundaries:** Applications have a large user-base, that may span from intra to cross boundaries. Depending on the roles of actors in organizations actors have different authorizations and thus appropriate security mechanisms both for business actors and exchanged documents are applied.

6. **Varying business vocabularies:** Actors of an organization use and understand their own business vocabularies. They require some common interpretation when documents containing those vocabularies exchanged beyond boundaries.

When building a *DocWF* application, all these characteristics must be considered. Although theoretically this is not overly complex, in practice a straightforward design and solution is difficult.

2 Methodology and Modeling of Document-based Agile Workflows (*DocWF*)

Technology advancements of the internet makes it possible for rapid proliferation of web-based applications such as e-commerce, e-governance that support not only Business-to-Business but also Application-to-Application and Business-to-Consumer collaborations. These applications range from online booking systems to intelligent shopping assistance and health care platforms. The notion of workflow or business process has been the main enabler for orchestrating such intra and cross-organizational applications. One of the driving force of this advancement is the possibility of reusing and leveraging available IT components and developing applications rapidly for changing markets. The methodology and modeling of these collaborative and cross-organizational business processes and their later deployment approaches vary diversely. In the following, first we describe briefly the basic methodology of *DocWF*s, then the position of workflow technologies in the context of business processes and co-operative work. Finally, we motivate the needs for a new modeling approach for agile business processes.

2.1 Methodology of Document-based Agile Workflows

Our aim is to provide as much flexibility as possible to a *DocWF* actor during the design and execution time of a *DocWF*. Our approach combines a semantic enabled process modeling technique with a flexible binding approach to reusable components (e.g., Web Services, EJBs or human) as depicted in Figure 1.1. This is a two step process as described below:

1. **Determining business tasks:** Determine possible business tasks by performing a semantic matching of a concerned business goal with the corresponding annotation of business tasks of a BPMN model in an organizational repository (i.e., a knowledge-base *KB*).

2. **Binding to services:** Perform similar matching by, for instance, semantic service discovery to bind a task to concrete services.

Business domain experts (i.e., *DocWF* modelers) can specify organizational business goals of an agile workflow scenario in an abstract form. They can further specify business rules associated with the business goals. Such rules specify constraints over related documents and business goals and as such this approach of *DocWF* modeling is declarative (see Figure 1.1).

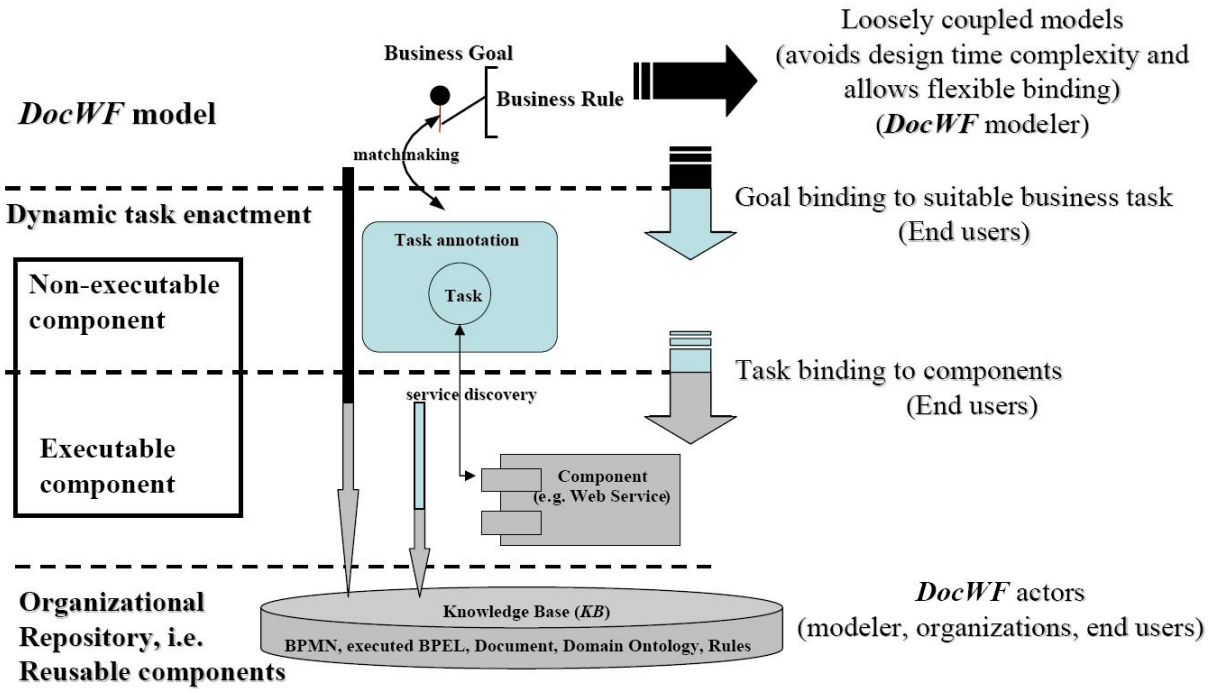


Figure 1.1: Flexible binding of document-based agile workflow (*DocWF*) models to design time and executable components.

Such a strategy level process models are then systematically grounded to executable components for their executions and thus called loose coupled models. To determine suitable business tasks and their later concrete binding to services (i.e., components), the dynamic task enactment of Figure 1.1 leverages the techniques of semantic match-making in the knowledge-base (*KB*) of a concerned peer. The knowledge-base contains among others semantic description of an organization's business processes (e.g., semantic BPMN) and executed business process description (e.g., executed BPEL) which enables them to be searchable by, for instance, string matching of a business goal with BPMN task annotation and of a BPMN task annotation with a semantic web service.

2.2 Workflows for Business Process Management (BPM)

Workflow technology has been used for a while to support key business processes and thus there is nothing new here. What has changed, though, is the realization that one of the easiest ways for organizations to be competitive, to reduce TCO (total cost of ownership), to be viable, to be flexible and to be responsive, is to understand and improve the structure and execution of their business processes. Through the deployment of workflow management software, organizations can meet those objectives [BPS].

While workflow may be manually organized, in practice most workflows are organized in the overall context of IT systems. What this means is that a workflow management system is

essentially a set of development tools that define, manage and execute workflows through the operation of software, which is driven by a computer representation of the workflow logic.

Business Process Management (BPM) has very similar dynamics at work but it works at a higher level of abstraction like information processing where one can bring together business processes from multiple organizations and systems to form one contiguous and manageable process. BPM creates an abstract process layer that takes the charge of control away from a workflow application layer by accessing repositories and applications, and letting knowledge workers to access those at appropriate points of a process. Therefore, BPM automates not only the flow of information in documents but also actions such as extracting customer information or adding new information about a customer transaction, and then generating transactions in the multiple systems involved in a business process. BPM technology effectively tracks and orchestrates business processes regardless of who or what performs the business tasks.

2.3 Document-based Agile Workflow Applications vs Computer Supported Co-operative Work

Workflow technologies, for instance, computer supported cooperative work (CSCW) are increasingly used for collaboration work in a group of peers. According to [Pra] any design of a computer supported collaborative application must support the following basic collaborative edition features among others:

- **A rich document structure:** A document structure may serve as a medium of collaboration for brainstorming, organizing ideas, or as a means of communication among peers.
- **Collaboration awareness:** A peer editor of data should provide sufficient context information so that users are aware of other active peers and their activities in the group.
- **Fault-tolerance:** A group of peers should continue data exchange despite machine crashes and actors join or leave a group. Fault-tolerance protocols should be designed to minimize the impact of network latency on response times experienced by actors, so that existing group collaboration can run smoothly.
- **Concurrency control:** Concurrency control is needed to ensure consistency of data being edited in parallel.

For *DocWF* applications assuring above mentioned features is a challenging task and sometimes infeasible or irrelevant as described below:

- Document content and their structure provide a better method for collaboration through information sharing when business actors possess common document schemas based on

which they generate and validate document content and structure. A shared static document schema amongst peers however decreases flexibility when information structures of interacting actors evolve.

- Knowledge of all peers in real time may not be feasible as increasing number of a priori unknown actors can join and leave a collaborative application autonomously.
- Maintaining a fault tolerant group communication in a pervasive environment akin to agile workflow application is challenging as opposed to a static group of peers.
- Finally, concurrency control is not an issue as an actor tends to work on her own copy rather than on a shared document. As such converging local document updates [VCFS00] is rather more important than controlling concurrency.

2.4 Modeling Approaches for Document-based Agile Workflow Applications

A business actor typically being a business domain expert can model a business process from various points of views as described below:

- **Task-based:** An exhaustive set of business tasks, their execution sequences and associated business rules are the drivers to model a task-based workflow where business rules typically govern execution paths of the business tasks. One feature of this task-based workflow modeling is the decision task nodes such as gateways acting as place holders to apply associated business rules possibly before and after each business task and thus makes a model cluttered. This modeling approach is suitable for routine business scenarios and thus the most prevailing one as productized by major industry players including IBM, SAP, Microsoft and Tibco [IBMa, SAPb, Hola, Holb, vdAW05].
- **Case-based:** In a case-based approach (known as "case handling workflows") [vdAW05] each case or one may call each instance is handled in isolation. In particular, for each business scenario there exists precise business tasks and their sequence flow definitions. Intuitively for each case the task-based modeling approach can be used.
- **Goal Driven:** It describes a goal oriented workflow modeling technique to generate alternative workflows whenever necessary [Rob96]. In this case, goals are statically linked with precise tasks and services and thus lacks of flexibility when it comes to realize a goal with alternate tasks or services.
- **Declarative:** In this approach a set of constraints, defined at design time, set boundary conditions associated to either any two business tasks or an individual task to control their execution sequences at runtime [MPvdA07, vdAP06]. This modeling approach aims at flexible workflow management as opposed to typical task-based workflows. In particular, declarative approach eliminates exhaustive decision task nodes of task-based modeling

approach. It also eliminates the specification of sequence flows of tasks at design time as these can be determined at runtime by checking the associated constraints. However, the complete set of business tasks must be known a priori so as to specify constraints over them.

- **Document-based:** It is sometimes called content-oriented workflow systems or knowledge oriented workflow systems that place the content object, i.e., document or data, in the center of a workflow process. Each pre-specified workflow task execution will alter the states of documents deciding further routing of the documents and as a result next tasks can be activated for execution. A document-based workflow can also be modeled by a task-based workflow assuming the operations performed on the documents have some dependency. Realizations of such dependencies may also vary, for instance, in [Ros04, DEL$^+$00] various document properties are defined and in [WK05, Sto] various events are defined to decide which tasks will be activated and need to be executed further. On the other hand, the approach of X-Doc-WFMS system [KMK02] specifies document structures as templates and also provides a mechanism for concurrent access that determines a task activation. However, all these techniques introduce task activation nodes similar to decision task nodes and thus make models more cluttered.

- **Scientific Workflows:** While scientific workflows are in the line of content oriented workflow modeling, their modeling approach differs in that this might not process structured data and also have very specific tasks to perform and to share depending on the targeted scientific domain [Ba05].

However, all the mentioned modeling approaches assumes various static elements related to workflows as illustrated below:

- **Prior knowledge of business tasks:** All business tasks of a business scenario are known a priori or can be determined before modeling the business process.

- **Prior specification of task sequence flow:** All possible execution paths of business tasks are known a priori for a business scenario.

- **Static business rules:** All business rules are either known or determined a priori of a process execution and thus support limited contextual business rules such as based on document content and current business goals.

- **Closed business boundaries:** All business peers and thus the business actors are operating either within the same business unit or from different business units having trust relationships.

Clearly, the mentioned assumptions are the characteristics of a routine business scenario such as in a "supply chain" where a requester, a supplier and a shipper know their precise tasks to be performed in a precise sequence, for instance, "request for a quote", "checking availability

of the requested product" and "deliver to a shipped address" and so on. All actors have their long standing business rules associated to tasks and possibly actors have trusted relationship in a closed business boundary. All these lead to major drawbacks in the context of modeling agile business processes as those assumptions may not hold simply due to the peculiarities of an agile scenario. Therefore, we need a more high level modeling approach closer to the business strategy level that can be executed later by leveraging reusable IT components as much as possible in a loose coupled fashion.

2.5 Business Rules for a Document-based Agile Workflow

A business rule is a statement that defines or constrains any aspect of a business that may influence the behavior of the business process [BR]. The statement can be about day to day business, access control, any business calculation or anything related to an application. A typical task-based workflow application has the following limitations with respect to specifications of business rules:

- **Task Association:** Business rules are statically associated with business tasks.
- **Enabling Static Execution Paths:** Business rules typically govern an execution of a workflow by enabling a sequence of predefined business tasks.
- **Isolated Rules:** Business rules are separated logically and physically from other components throughout a business process [Hua].
- **Rule Checks:** Business rules are checked only before or after a business task is performed.

As concrete business tasks and control sequences may not be known a priori in agile business processes, business rules can not be associated on the business task level. For a similar reason enabling static execution paths is not also feasible for a *DocWF* application. Moreover, as in exhaustive task-based modeling approach of previous section, exhaustive business rule specifications for each task and execution path may also clutter a model. Despite physical separation of rules from other IT components eases the rule management logical associations with business notions such business goals allows business actors to specify loosely coupled *DocWF* models. Checking of business rules anytime including a task execution enables actors to detect modeling errors and take necessary remedy for an immediate or a later execution.

3 Workflow-based Agile Business Process Scenarios

There are plenty of different use cases for agile business applications, as illustrated in the following scenarios:

- **Emergency patient care management:** In this scenario, actors i.e., administrative employees, diagnosis technicians, pathologists and doctors from distinct business units of a hospital collaborate to generate an electronic health care record (*EHR*) containing treatment information for a patient. Chapter 3 describes such an example scenario wherein we demonstrate how document-based agile workflow methods can be used to model the *EHR* workflow using high level business goals and associated business rules. To detect errors in models relevant model checking techniques based on finite automata representations of the rules are also described there.

- **Merger and acquisition:** Today's business organizations merge and acquire other organizations to cope with the constant changes in the market and to be competitive. In this example, a document-based agile workflow system enables non-disruptive document exchanges amongst workflow actors during an execution by using a stable interface. Such an interface is based on semantics of document content while allowing re-structuring of syntactical data models as opposed to using a syntactical structure-based interface, for instance, XML schemas.

- **Cross-border incidents:** Two elaborate examples of cross-border incidents are described in chapters 5 and 6 to describe an ontology driven communication infrastructure and security solutions respectively for document-based agile workflow applications. In the first scenario of Chapter 5, a citizen of one country fell into an accident while driving in another country. In this case, various organizations including polices of both countries, hospitals, news agencies and courts get involved in order to give a verdict or process an insurance claim if there is any. In the second scenario of Chapter 6, business actors such as the police, justice, customs from different countries collaborate after an occurrence of a cross-border crime where the criminal has been identified in a foreign country. Business actors exchange information in order to, for instance, decide dynamically the next business task or to generate a composite document called arrest warrant that will be used as a basis for further investigation.

  In both cases, each country has individual information structure and business vocabularies. Moreover, they may have confidential information for which their access control policies apply before document distribution. Depending on the received document content and organizational policies the recipient organization take further actions and update the document.

- **High volume business data exchange:** In this example, peer organizations in an ERP application such as a supply chain scenario, build a large and composite XML document by collaboration amongst peers according to some business rules. These documents are possibly encrypted in fine granular fashion to enable fine grained document access by legitimate *DocWF* actors or peers.

4 Communication Infrastructure for Document-based Agile Workflows

Today's SOA-based collaborative business applications such as ERP software (e.g., CRM, SCM) exchange huge volumes of data in documents containing various meta data in addition to the payload. The meta data is used in various stages of processing, for instance, message mediation, adding security headers to an outgoing message and content-based message routing. Such processes can only be done at runtime and thus are dynamic. Data providers may not be available indefinitely and thus can not do such processing. Such processing instead can be delegated to a communication infrastructure that enables loosely coupled data exchanges and the synchronization between a priori unknown peers in a workflow. Examples of such infrastructures are enterprise service buses (ESB) [SAPc, IBMb, PET, WSO] that will continue to flourish. Communication approaches of such an infrastructure can be mainly classified into two main types:

- **Centralized:** In this setting a dedicated communication infrastructure provides meta data processing services and thus act as a central point of coordinator between known peers.
- **Decentralized:** In the decentralized setting a set of coordination engines possibly installed in different business boundaries take over those tasks.

Existing communication infrastructures are based on either of these two approaches. However, they are limited when executing *DocWF* applications that could potentially leverage on the document-based agile workflows and related technologies as illustrated below.

- **Static Peer Information Structure:** Generally speaking, existing communication infrastructure solutions for workflow systems are quite static in that the peers or organizations involved in the execution of an agile process are often limited to a fixed organizational information structure (e.g., document structure) and thus restrict the evolving nature of information structure both at design and runtime. So any change in the information structure due to the organizational updates, for instance, after a merger breaks the existing data exchange interfaces with other peers and their collaborations are disrupted.
- **Limited Meta Data Processing:** Existing communication infrastructures largely deal with message routing related meta data, such as destination address, service endpoints etc. However, a *DocWF* application demands new kinds of meta data processing related to, for instance, fine-grained document access, filtering for selective document distribution to legitimate recipients according to original data providers policies and document integrity protection.
- **Closed Business Boundaries:** A centralized communication approach on the one hand constitutes a central point of coordination potentially liable for a performance bottleneck

for high volume data exchanges. On the other hand future peers may not collaborate through a workflow application unless they are from the same business boundary of the existing peers.

DocWF applications, for instance, the cross border incidents require the combination of a decentralized communication infrastructure that is capable of selective document distribution with enough meta data processing and of adequate security solutions. For sure, such incidents require data exchanges between a priori unknown law and enforcement authorities, news agencies, hospitals, and courts that may span multiple countries. Frequent merger and acquisition of organizations implies frequent information system architecture change which may affect existing data exchanges between peers. Current workflow management solutions do not seem to offer adequate execution support for such flexibilities to meet the requirements introduced by modern agile workflow applications such as the ones outlined in the previous section.

5 Interoperability for Document-based Agile Workflows

Information content, including that is present on the World Wide Web, is largely and primarily expressed and presented in natural languages. Consequently, a large gap has been emerging between the information available for tools aimed at addressing the problems above and the information maintained in human-readable form [FBD+02]. The "Semantic Web" has been the main enabler to address this gap by having machine-readable information and automated services that extend far beyond the current capabilities [TBLL01, DFKO02, W3C]. Software vendors like HP, IBM, Microsoft, Oracle, SAP, and Software AG have recognized these facts and have already begun to adopt Semantic Web technologies in their mainstream products [PS07, SWT, SWU]. In the context of a *DocWF* application, document exchanges are the only interface between a priori unknown actors. As such, documents need to be more interoperable so that content is more machine-readable in spite of varying document vocabularies. In the following, we outline the main enablers of interoperable documents which we detail in Chapter 3. Such enablers are semantic business process management, semantic enabled business documents, and policy specification as illustrated in the following.

5.1 Enabling Semantics in Document-based Agile Workflows

The fact that business drives IT and communication gap between business and IT people gave birth to BPM (Business Process Management) and BPMN (Business Process Modeling Notation) [BPM]. The goals of these approaches were the following:

- **Bridge Business and IT:** Reduce the gap of understanding of an organization wide business process between business and IT people and thus make a bridge between them.

- **Reuse:** Promote the reuse of existing services and capabilities as to enable quick service and product delivery into the market.
- **Agility:** Be flexible enough both in business process modeling and execution levels so as to be agile.

Despite leading software vendors like SAP, IBM have already productized their BPM enabled products, apparently the mentioned goals are still in infancy due to the following:

- **Peculiarities and varying business vocabularies:** Each scenario confronts with its specific features and thus can not be generalized by typical business tasks in a BPMN model. Moreover, such a scenario involves the interaction of peers from varying business boundaries with unavoidably variant business vocabularies.
- **Limited Machine Executable Processes:** The designed workflow models are far way from their executable models. This is due to the absence of standard transformation from design to execution model. For instance, BPMN models are mostly used for documentation purpose as opposed to immediately executable models and thus leading to developing proprietary transformations from BPMN to BPEL for instance.

For the former, the reason is pretty clear that every organization has its independent data models and vocabularies to understand the same business concept for instance. Regarding the executable models, for all BPMN notions there does not exist equivalent execution elements in BPEL [KHB00, Vig08] resulting in limited automation during an execution of a business process model, for instance, from a BPMN model to an executable BPEL [BPE]. Organizations thus interpret some BPMN models in their own way which results into proprietary execution BPEL engines and also spend considerable human interaction effort to make an executable business process.

Given the abovementioned facts, business actors in an agile workflow scenario can not rely only on typical business process management techniques. In particular, it calls for a looser coupling in modeling with its respective execution and content-based interoperability, e.g., ontology, amongst actors as opposed to structure-based interoperability, e.g., schemas. An EU research project called SUPER [SPM] has introduced the notion of semantic business process management in this direction. The idea is to make a BPMN model more machine readable by annotating the business tasks with, for instance, business goals, pre- and post-conditions along with the maintenance of an ontology representation of the business domain objects. These annotations significantly increase execution flexibility by providing the possibility of selecting suitable business tasks and executing them using a wide range of available IT components, for instance Web Services.

In our agile workflow setting semantic business process management is related in that, for instance, a business goal can be achieved by a business task whose goal annotations match with

that business goal. Further, that task can be realized by an execution of one or a composition of Web Services. A recent interest group called the "Semantic Business Process Management Working Group" [SBP] also states similar goals in their mission statement.

5.2 XML structure-based Document Interoperability for Document-based Agile Workflows

Since its introduction, the eXtensible Markup Language (XML) has become the de facto standard for data exchange and sharing. There are two main techniques for exchanging and sharing basic XML-based data based on XML standards:

- **Schema and DTD sharing:** There exists a variety of XML schemas and DTDs for interoperable data exchange for specific business domains such as the Darwin Information Typing Architecture (DITA) Pharmaceutical [DIT] for pharmaceutical industries, Open Document Format (ODF) for office applications [ODF] and *eXtensible Business Reporting Language* (XBRL) [XBR] for financial industries.
- **XML manipulation:** This consists in the manipulation of XML data structures using X-Path [CD], X-Query [BCF+], XSLT [XSL99] etc. in order to refer to a document portion, to query and to transform documents respectively.

However, these XML standards have not been designed with cross-standard interoperability in mind. There are some strong assumptions regarding current data exchange and sharing technology:

- **Sharing in a closed boundary:** Business actors that share data collaborate in a close environment and have trusted relationships. As such, they have a clear understanding of each other's XML vocabulary semantics.
- **Point-to-point sharing:** Each business actor is a terminus in the data sharing pipeline.
- **Sharing static business vocabularies:** Business actors share XML documents that contain static XML vocabulary.

Given these, several document interoperability initiatives for dedicated business domains have been formed recently. For instance, for news media (i.e., NewsML [IPT]) and browser applications (i.e., OpenXML [OPEb]). However, in the context of agile business scenarios such straightforward cross-standard interoperability solutions would not be feasible [SHO] as each scenario has its individual peculiarities and may end up with a specific XML schema for every scenario. Having said that and considering agile scenarios where one or more of these assumptions are not valid, a document interoperability solution beyond the XML structure is indeed required.

5.3 Enabling Semantics in Business Documents of *DocWF*s

The emerge of semantic web technologies makes it clear the necessity for interoperable descriptions of business documents for effective collaboration amongst disparate business actors [TBLL01]. This is more evident in B2B architecture use cases for semantic web services developed by DAML [SWS] where semantic mediation between different organizational vocabularies is required. Clearly, mechanized support is needed to deal with numerous and heterogeneous data formats of organizations. In this context, various standards [EBX, XBR, Ros] exist and are emerging on how to describe products and services, product catalogs using business documents. For instance, ebXML [EBX] focuses on building XML vocabularies and messaging formats for trading amongst trade partners and thus focuses on e-commerce solutions and XBRL [XBR] provides financial reporting related XML vocabularies. However, these lead to the following limitations among others.

- **Domain specific:** These are mostly suitable for core e-business applications.
- **Reinventing the wheel:** The proliferating number of such standards for variety of business domains might result in the need for yet another mediation between standards.

An organization performing businesses in different domains, for instance, in government services, trading businesses or in financial services needs to comply with heterogeneous standards. A semantic approach is required to define such standards better and to map between them whenever used in any agile workflow application involving cross business domains. Agile workflow applications being spanned through multiple business boundaries require very flexible and quick bridges between different terminologies as to ensure an open and loosely coupled communication.

5.4 Semantic Enabled Policy Specification and Checking for *DocWF*s

Agility includes the pervasive computing environment where actors can join and leave a collaboration and thus can be mobile and access services and devices in their vicinity. Such agility requires policy-based security due to an extremely dynamic environment [KFJ03] that tends to span several domains and be made up of varying security requirements. In this setting, policy providers and policy evaluators can be different entities where the latter check the policy on behalf of the providers. As a result data providers tend to have diverse policy specification languages and thus policy checking techniques also vary. Indeed, a policy language for such environments needs to be very expressive and easily extensible as shown in [KFJ03]. Semantic-based policy specification for semantic enabled entities such as web services is a solution in that direction [KPS$^+$04]. In this context, a Publish/Subscribe-based decentralized communication infrastructure supported execution environment that we develop in Chapter 5 features following two characteristics among others:

- **Ontology-based policy specification:** Independently of individual policy specification languages of content providers the decentralized infrastructure being the policy checking entity maintains a high level policy specification as can be represented by a policy ontology. Each provider either provides its policy as an instance of the policy ontology or the infrastructure transforms the provided policy specification into an instance of the policy ontology. As such,, the ontology-based policy specification is formed by all policy instances associated to the data providers.

- **SPARQL [SPQ]-based policy checking:** As policy is specified using ontology in the infrastructure level, the policy checking is entirely performed by semantic queries that typically need to be handled with a language such as SPARQL (SPARQL Protocol and RDF [RDF] Query Language).

6 New Requirements for a Document-based Agile Workflow System

The design of a document-based agile workflow management system should not only take into account functional requirements derived from agile workflow application scenarios but also meet interoperability and security requirements raised by such scenarios. Based on the modeling and decentralized communication requirements we identified above, a document-based agile worflow system needs to address various requirements in order to implement agile scenarios and introduces new research challenges as opposed to usual workflow management systems. While these new requirements are presented briefly in the following, later chapters elaborate more on the specific requirements accordingly.

6.1 Strategy Level and Loosely Coupled Models

Business actors must be able to model agile business processes using business vocabularies such as business goals and business rules. The corresponding realization of a business goal should be facilitated by contextual information as captured in the business documents and by the runtime enactment of business tasks that is realized by leveraging possibly reusable Web Services. Such a modeling and its realization approach are required to meet the following requirements brought up by agile execution environments:

1. **Loosely coupled models:** A *DocWF* model should not rely on existing services as opposed to typical task-based workflow modeling.

2. **Reuse:** Reuse is desirable not only during runtime task enactment but also during process modeling.

3. **Distributed control:** No centralized point of control is assumed to be present.

4. **Dynamic task enactment:** Enabling task enactment as dynamic as possible by for instance semantic-based service discovery at runtime.

5. **Modeling error detection:** Enabling early detection of design and execution errors due to faulty models and fixing errors without restarting a process.

6.2 Interoperable Documents

We require an interoperability solution making business documents interoperable when exchanged beyond business boundaries. Such documents must enable:

1. **Non-disruptive document exchanges:** The interface amongst organizations must be stable by limiting changes to the interface as much as possible.

2. **Distributed document handling:** Documents can be created, accessed and updated autonomously by actors and services hosted in different business boundaries. Therefore distributed modifications performed by actors need to reconciled methodically.

6.3 Loosely Coupled Document Exchanges

A decentralized communication infrastructure must support a loosely coupled interplay of distinct business actors of a document-based agile workflow application. One key requirement is to enable document exchanges between a priori unknown actors that relies an ontology as their only interface for data exchanges. There are mainly three high level requirements that are brought up by the design of a suitable decentralized communication infrastructure and associated coordination protocols between actors.

1. **Semantic-based policy specification and checking:** Policies must be specified at the document semantic level as opposed to document structure so as to enable flexible policies and automated policy checking by the communication infrastructure (on behalf of the document providers).

2. **Semantic-based document exchanges:** Document content must be routable to legitimate actors leveraging content semantics (i.e., ontology *concept*) as opposed to document structure-based data sharing. Moreover, the communication infrastructure must be able to deliver semantically related documents to the legitimate actors in a selective fashion independently of the original document providers.

3. **Fully distributed communication:** Communication amongst a priori unknown actors must be supported by the decentralized communication infrastructure in the absence of a dedicated coordinator. Such supports includes temporary storage of documents to handle asynchrony, coordination between decentralized communication infrastructure nodes, coordination between an actor and an infrastructure node. Coordination tasks include managing peer subscriptions, key management for instance.

6.4 Security of Document-based Agile Workflows

The loosely coupled *DocWF* process model, being a high level strategic style of agile workflow modeling, does not allow typical model-based testing and verification to check the correctness of modeling and its execution. As a result,basic correctness verifications may no longer be enforced, including the assurance that a model does not constitute a deadlock or a conflict according to some business rules. Then, a business domain ontology being the only interface and messaging tool for business actors the typical workflow security concerns such as task authorizations and document schema or structure level authorizations are no longer adequate. This is because in a *DocWF* application one simply may not know the suitable tasks a priori and no actor, including the infrastructure, may not know the document structures of the providers. Besides business actors are a priori unknown who may join and leave a *DocWF* application. They exchange documents with varying vocabularies for the same semantic business *concept* and possibly multiple versions of a document. Moreover, selective distribution of semantically related documents to legitimate actors is performed by a decentralized infrastructure who acts as a delegate of the original document providers.

Given all these, a plethora of security issues related to the workflow modeling need to be addressed: for instance, how to detect deadlocks and conflicts of process models. A large number of issues related to the document security also have to be solved: for instance, document access control and integrity need to be enforced. These issues are generalized in the following:

1. **Design time checking of models:** As organizational business rules associated to business goals can be introduced autonomously it is crucial to detect anomalies, for instance, deadlocks and conflicts (constituted by combination of business rules) early before actual *DocWF* execution when task enactment occurs.

2. **Document access control enforcement:** Ontology level document authorization policies must be enforced on the document structure level. In particular, for an ontology *concept* authorization, fine-grained document portions originating from different document providers may need to be delivered to authorized peers during a *DocWF* execution.

3. **Document integrity protection:** As documents will be annotated with semantic information, like for instance ontology *concepts*, and routed through insecure communication channels, integrity protection with respect to document semantics and structure is required.

Existing security mechanisms designed for protecting the execution of traditional workflow applications do not offer adequate solutions to solve these new security issues introduced by information intensive agile workflow applications. Existing access control mechanisms are mostly devoted to either access right management on business task level [LZS09, ACM01, CLW05, KPF01] or guaranteeing execution proofs against an a priori specified series of business tasks [MM07]. Instead, a *DocWF* execution requires enforcement mechanisms for fine-grained document access control in a distributed setting. While encryption over fine grained documents is the general answer to this problem, basic key management issues such as computation/recomputation of a same key associated to an ontology *concept* by a priori unknown actors are challenging tasks. Therefore, basic security issues in a document-based agile workflow setting are not addressed yet. In this thesis, we thus suggest the design of appropriate security mechanisms described in different chapters in order to meet the security requirements associated with the modeling (Chapter 3) and execution of document-based agile workflow applications (Chapter 5: semantic enabled policy checking and Chapter 6: enforcement of fine grained document access control, document integrity protection and key management).

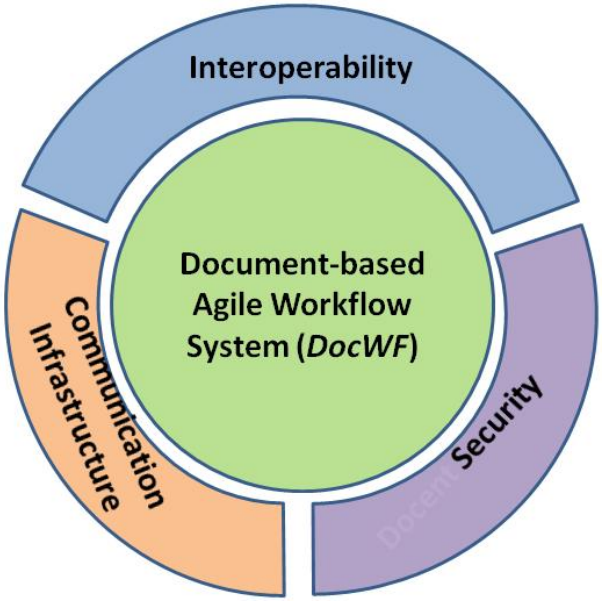


Figure 1.2: Thesis Organization in terms of core building blocks.

7 Thesis Structure and Contributions

Each chapter of this thesis stands for a building block in the design of a document-based agile workflow system to support executions of an agile workflow scenario. The four main components we mentioned in this section are depicted in Figure 1.2. The central component is the document-based agile workflow system that should meet the functional requirements (design and runtime principles captured in *DocWF* models) we identified in order to offer adequate support for the execution of document-based agile workflows. In order to implement interoperabil-

ity and loosely coupled document exchange capabilities of a document-based agile workflow application, the *DocWF* system is supported on the one hand, by an ontology-based document modeling technique that is in charge of ensuring adequate semantic annotations on documents. It also includes document comparison algorithms to implement document convergence. On the other hand, an ontology-driven distributed communication infrastructure supports the *DocWF* system by selective document routing and delivering to the legitimate peers on the basis of authorization policies of document providers. Finally, the building block of security enforces various security requirements. The remainder of this manuscript is outlined as follows.

Chapter 2 Preliminaries and Technical Background:

In this chapter, we provide some basic technical background material that will be used throughout this thesis. We especially introduce the basics of task-based workflows, document-based workflows, document-based agile workflows and their relation with business processes. We then shortly discuss communication infrastructures and specially Publish/Subscribe-based methodology with respect to the suggested decentralized infrastructure support for document-based agile workflow applications. We then move on to discussing high volume business document processing and their security issues akin to any document-based agile workflow applications before concluding.

Chapters 3, 4, 5, and 6 are the core of this thesis and describe the solutions we designed towards reaching the objectives we set in this introductory chapter. The reader is invited to read them linearly. However, the solutions presented in different chapters can be used independently in other context. Distinct example scenarios are introduced in different chapters to motivate the respective solution accordingly. Each chapter contains some cross references of previous chapters indicating a basic understanding of those references is required to proceed.

Chapter 3 Document-based Agile Workflows:

As opposed to traditional workflow-based collaborative applications, the modeling of document-based agile workflow applications may not be done using concrete business tasks and their sequence flow to provide an abstract business process view to the business actors. As such, the execution of workflows may not rely on predefined and dedicated services for task enactment. In this chapter, we focus on the loosely coupled modeling approach of document-based agile workflows and their execution principles for agile workflow applications termed as document-based agile workflows (*DocWF*). The modeling approach features the usage of business vocabularies such as business goals and associated business rules. Execution principles support dynamic task enactment by enabling, for instance, semantic match-making of goals and semantic service discovery techniques. The business rules, capturing various concerns such as business goal dependency and appropriate processing based on the document content, can be formally represented by finite automata expressions of the rules. These formal representation can then be used during execution time in order to detect some modeling errors.

Chapter 4 Interoperability of Document-based Agile Workflows:

In this chapter, we tackle the issue of interoperability within document-based agile workflow instances and propose an ontology-based business document interface amongst distributed business actors to assure a stable and flexible communication means amongst them. The stability comes from the facts that an agreed ontology representing business document content semantics hardly changes and it is independent from the underlying syntax level data models. Flexibility is increased as first individual data models of a business actor can evolve without affecting the stable ontology-based interface with others, second, policies can be specified on ontology level and yet fine grained document access control can be enforced on the XML syntax level and third, custom semantic annotations into the exchanged documents are possible allowing an eligible recipient to understand the document content even if it's local knowledge is based on a different vocabulary. Finally, as mentioned in Section 2.3 to support document handling due to autonomous document access, updates during a *DocWF* execution appropriate document comparison algorithms are developed in this chapter.

Chapter 5 Communication Infrastructure for Document-based Agile Workflows:

In this chapter, we present a decentralized communication infrastructure and associated communication protocols to enable loosely coupled document exchanges. The infrastructure is based on an ontology-driven Publish/Subscribe methodology to assure a selective document distribution during an execution of a document-based agile workflow application. The design of the suggested decentralized communication infrastructure is strongly coupled with the ontology representation of the business domain in order to ease their integration with the ontology driven data interfaces developed in Chapter 4.

Chapter 6 Security of Document-based Agile Workflows:

In this chapter, we present security solutions supporting the secure execution of document-based agile workflows. These solutions describe distributed access control enforcement mechanisms for fine grained XML documents. These mechanisms capitalize on a group-based cryptographic technique called tree-based group Diffie-Hellman (TGDH) [KPT00]. We adapt the TGDH technique to enable a group of peers to compute the group key independently where peers have the same authorized subscriptions for an ontology *concept*. In particular, it aims at restricting document access through encryption of document parts with keys shared by a group of peers with similar access rights. This independent key computation allows a legitimate peer to encrypt/decrypt documents autonomously in a fine grained manner and to prevent malicious business actors from performing unauthorized access and actions in documents, for instance, illegitimate updating, deleting, inserting and even moving of nodes. Traceability for an posteriori verification is also enabled

based on specialized assembly of security meta data. The adapted technique limits the scope of rekeying when peers dynamically join or leave in an agile environment so as to enable non-disruptive document exchanges for an extended period of time. The design of the suggested mechanisms is strongly coupled with the ontology-based policy specification of Chapter 4 and selective document distribution of the infrastructure of Chapter 5 in order to ease their integration with semantic enabled document description by the document providers and semantic-based routing by the infrastructure respectively.

Appendices Appendix A provides details of an implementation of the communication infrastructure and appendix B describes some implemented security libraries for secure document-based workflow executions. Finally, appendix C illustrates the formal document-based agile workflow process execution of Chapter 3 with the transition rules. The reader is also invited to read appendices in a linear fashion.

The research work performed by the author in the scope of this thesis resulted in a number of scientific publications [RRS, RRS09a, RRS09c, RRS09b, RRMS09, RRS08, RS07a, RSR06, RMS06] and industrial patents that contain the main ideas presented in this manuscript.

Chapter 2

Preliminaries and Technical Background

All progress is precarious, and the solution of one problem brings us face to face with another problem.
- Martin Luther King, Jr. -

In this chapter we introduce some preliminaries related to the solutions described in this thesis. Modeling basics associated to task-based workflows, document-based workflows and our proposed document-based agile workflows are first presented using some illustrative examples before illustrating their differences. We then review some technical background related to communication infrastructures and large XML document processing such as document parsing and document comparing. We also briefly discuss associated security issues with respect to documents (e.g., document access control and integrity) and the communication infrastructure for implementing document-based agile workflow applications.

1 Task-based Workflows

The task-based workflows are defined by the workflow management coalition [Holc] as follows: "A Workflow is concerned with the automation of procedures where documents, information or tasks are passed between participants according to a defined set of rules to achieve, or contribute to, an overall business goal". In the context of business processes a task-based workflow specifies a sequence of routine procedures in the form of business tasks that can be performed

by distinct business peers in order to achieve their business goals. This is illustrated in the following example.

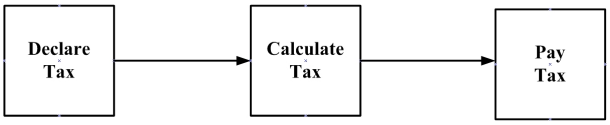


Figure 2.1: A simplified task-based tax handling process.

Example 1. A tax consultant, being a business expert of a "tax handling process" domain, may define a sequence of business tasks and associated business rules. Figure 2.1 depicts a simplified tax handling process that consists of three routine business tasks: "Declare Tax", "Calculate Tax" and "Pay Tax" in that sequence. Each business task can be associated with one or more business rules, for instance, "tax declaration must be completed before summer" that can be associated to the "Declare Tax" business task.□

As mentioned in Chapter 1, this style of business process modeling is relatively static as routine business tasks and their possible sequences of a regular tax handling process can be determined a priori. This approach of modeling is widely adopted in industry and thus adequate tool support exists. However, the execution of a business process leveraging IT components has always been a challenge. Not to mention that in an agile business scenario this static style of modeling and its execution is not feasible.

2 Document-based Workflows

A workflow of this type borrows from information processing to implement a content-oriented business process as mentioned in Chapter 1. More precisely, workflow activities can be described by some *operations* on defined documents performed by actors in a business scenario as illustrated in the following example.

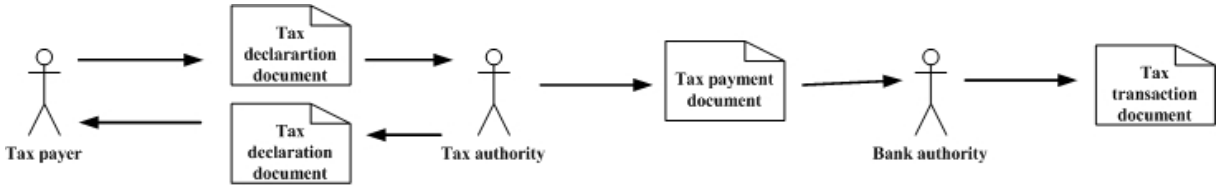


Figure 2.2: A simplified document-based tax handling process.

Example 2. With respect to the tax handling process several documents can be defined: 'tax declaration document', 'tax payment document' and 'tax payment transaction document' (see Figure 2.2). A tax payer first *declares* her income over the last year in the 'tax declaration document'. This document can be *edited* several times before *passing* it to the tax authority who can also *send* the document back to the tax payer for correction if it contains any error.

Based on this declaration the tax authority *calculates* the payable tax amount and *records* it into the 'tax payment document' which is sent to the corresponding bank authority for the actual money *transfer*. Upon a successful payment the bank authority *sends* a 'tax payment transaction document' to the tax payer for her future reference.□

As mentioned in Chapter 1 a document-based workflow can be simulated by a task-based workflow given the *operations* performed on the documents have some dependency. Such dependencies can be associated to tasks. For instance, the creation and according edition in the 'tax declaration document' must be performed before creating the 'tax payment document'. Similar dependencies exist for the 'tax payment document' and the 'tax payment transaction document'. Such a typical document-based workflow realization by a business actor is depicted in Figure 2.3.

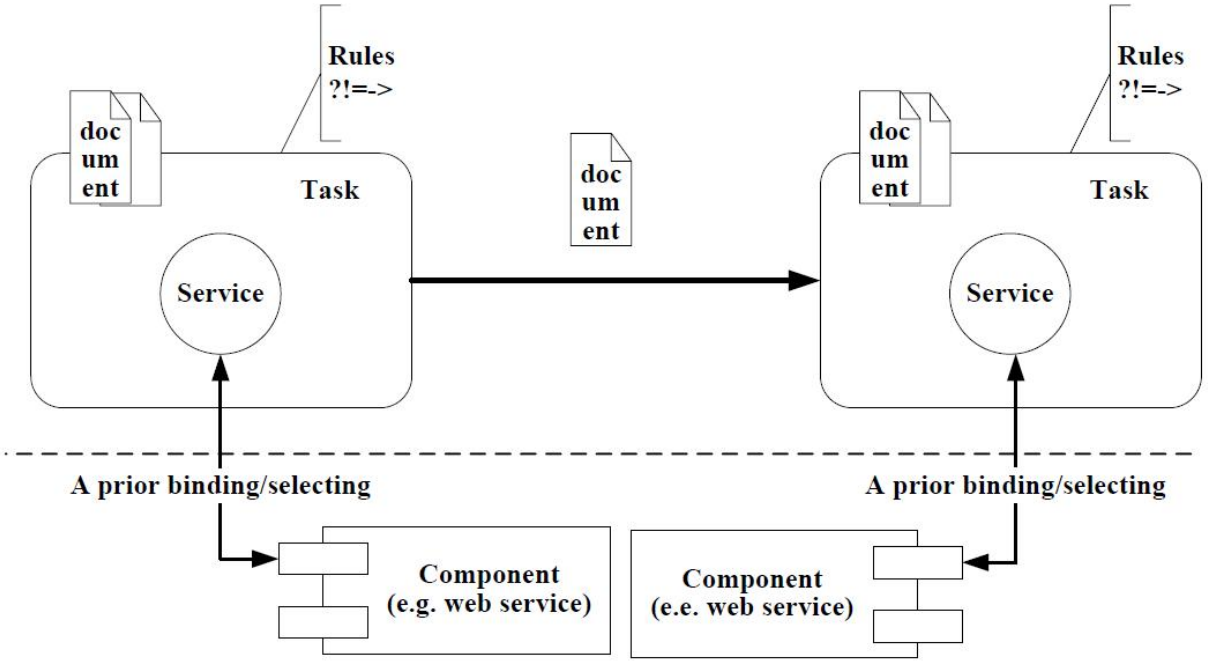


Figure 2.3: Typical document-based workflow realization by a business actor.

Here, a business task performs operations on documents based on some associated business rules and updated documents are considered for the next business task. Each business task is realized by one or more IT components (e.g., Web Services) that are a priori bound to or selected for the task.

3 Document-based Agile Workflows (*DocWF*)

A document-based agile workflow is an information intensive collaborative workflow application that deals with agile scenarios as defined in Chapter 1. Organizations involved in such

applications may confront such scenarios, for instance, merging and acquisition of companies, emergency patient health care, cross border crime etc. briefed in Chapter 1 where suitable business tasks may not be defined in design time as those can only be determined at runtime. Surely, such scenarios are information intensive, for instance, in a cross border crime, police, customs and courts of different countries have to exchange information regarding the crime, victims, spot, criminal and existing records to proceed further in the investigation.

A comparison of basic principles of document-based agile workflows with traditional workflows (e.g., task-based, declarative and case handling [vdAW05]) is reported in Figure 2.4. We discuss the principles of document-based agile workflows in detail in Chapter 3.

	Task-based Workflow	Declarative Workflow	Case Handling	Document-based Agile Workflow
Focus	Task	Task	Whole case	Document content, goal
Modeling artifact	Task	Task, constraints	Task	Goal, business rules
Task and sequence flow definition	A priori	Only task definition	A priori	Dynamic
Primary perspective	Control flow	Constraints over tasks	Case / instance	Constraints over goals, document content

Figure 2.4: Comparisons of document-based agile workflows with task-based, declarative and case handling workflows.

Considering the lack of knowledge of suitable business tasks at design time (see Figure 2.4), to model and execute a document-based agile workflow a loosely coupled process modeling approach can be followed. To design loosely coupled process models, a declarative technique is illustrated in the following section and will be detailed in Chapter 3.

3.1 Declarative Modeling of *DocWFs*

Business domain experts can specify organizational business goals of an agile workflow scenario in abstract form. They can further specify business rules associated with the business goals. Such rules specify constraints over related documents and business goals and as such this process is declarative. This is illustrated in Figure 2.5 and detailed in Chapter 3.

Compared to Figure 2.3, here clearly an agile business process is modeled leveraging the business notions, business goals and associated business rules as opposed to a sequence of pre-defined business tasks. To achieve a goal, suitable business tasks and their realizations by

concrete services can be determined in a loosely coupled fashion. This is a two step process as follows:

- **Determining suitable business tasks:** Determining suitable tasks to achieve a goal possibly by semantic matchmaking of a goal with a task annotation.
- **Binding to services:** Binding the tasks to concrete services possibly by service discovery capability of a business actor.

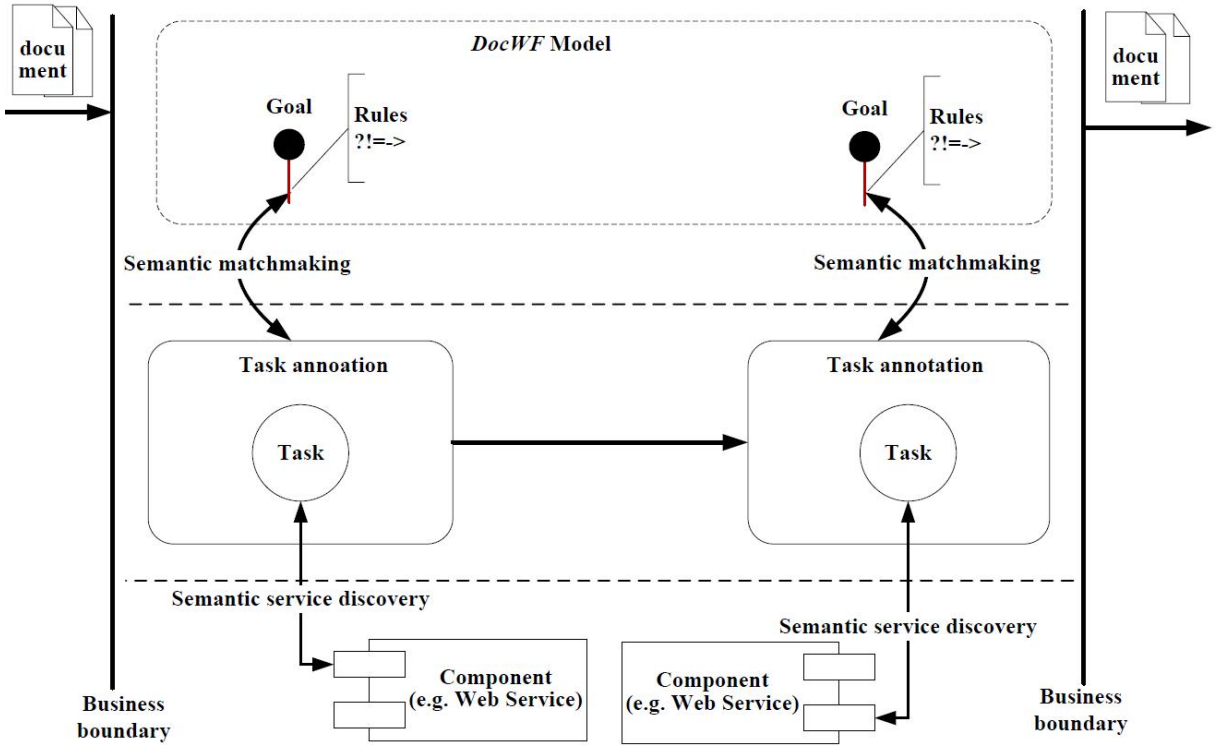


Figure 2.5: A document-based agile workflow realization by a business actor.

These two steps are further illustrated in the following based on the following example.

Example 3. Consider an event management company X specialized in organizing a conference proceeding process (CPP) for which X follows a sequence of two routine business tasks "submit papers" and "review papers". The "submit papers" task is realized by an internal submission portal service where all the interested authors submit their papers. For the "review papers" task company X relies on several third party content management systems. Depending on the availability of the third party services or requirements for a particular conference one or another service is used by the reviewers for their review work. Now, X is exploring a new business opportunity for offering services to manage a research proposals granting process (RPGP). □

Since the RPGP business domain is completely new for X it is not sure about the suitable business tasks and thus not sure about required executable components. However, after a quick

market and requirement analysis by RPGP experts they identified two kinds of documents that can be handled in a RPGP process. These are "proposal documents" and "review documents". After further analysis they come up with their business goals and the associated business rules upfront as shown in Figure 2.6. Now, the tasks "Submit" and "Review" of a BPMN model are annotated with the goals "Minimum 5 proposals" and "Selection of top 5%" respectively which they found after an extensive research or provided by a repository either locally or remotely. As the first business goal of the high level agile workflow model matches (i.e., semantic matching) with the goal annotation of the business task "Submit", X can safely determine that "Submit" is the suitable business task. Suitable IT components (e.g., Web Services) can be found for the binding of the task in a similar fashion using semantic service discovery for instance. However, the service must comply with the associated business rules. For example, for the "Submit" task the potential web service must ensure that "the call for proposal is published before summer".

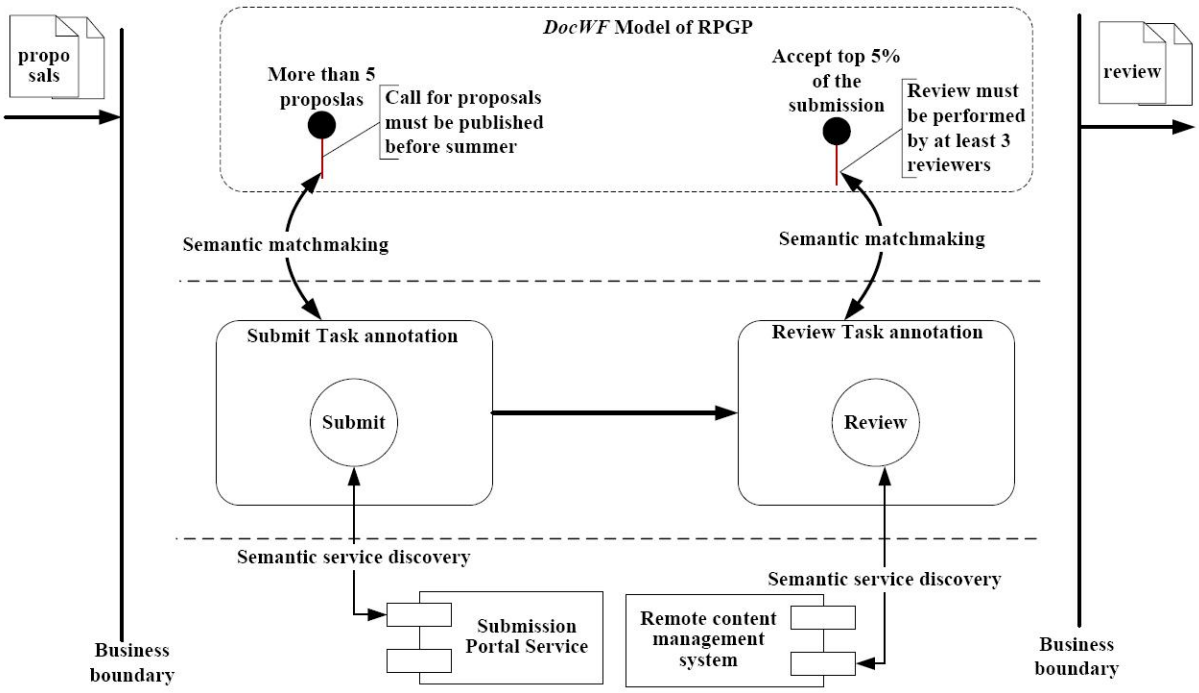


Figure 2.6: Document-based agile workflow for research proposal granting process (RPGP) realization by event management company X that is an expert for a conference proceeding process (CPP).

In order to determine the best possible "review papers" service, the company X needs the dynamic knowledge of available third party services. In Chapter 3, we developed a task state models in this regard. In particular, the task state model, a state machine model that enables a suitable binding of tasks to services based on the status of the available services.

3.2 Declarative Business Rules for *DocWFs*

As mentioned in a *DocWF* model, relationships between business goals and potential execution paths can be specified using business rules that set constraints over goals and document content for instance. As the models are loosely coupled they may introduce some inadvertent errors, for instance, conflicting rules. As mentioned in Chapter 1, business rules may not be associated to business tasks at design time. In the context of document-based agile workflow modeling a declarative approach seems to be more appropriate. We chose linear temporal logic (LTL) as an underlying formalism to represent declarative business rules associated to each business goal. Over the past three decades LTL has been the primary tool for specifying constraints of any distributed reactive system as first introduced by Pnueli [Pnu77] in 1977.

Using LTL-based business rules desirable properties of a *DocWF* execution and constraints to control potential execution paths can be formally specified by a set of well defined operators such as "Always" $\Box$, "Eventually" $\Diamond$, "Until" $\cup$ etc. Then with the help of transformation of LTL to finite automata such as Büchi automata [LTL] it is possible to verify properties such as absence/presence of deadlocks in *DocWF* models. In particular, it is possible to check whether the business rules associated to one goal or combination of several goals constitute a deadlock or a conflicting situation. This is a rigorous formal approach of *DocWF* modeling that allows business actors to detect such anomalies early, i.e., even before determining suitable business tasks. Such a declarative business rule specification based on LTL is illustrated in the following example.

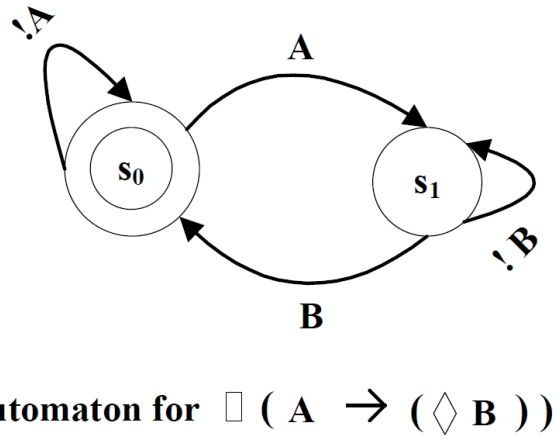


Figure 2.7: An automaton representation of a LTL-based business rule.

Example 4. Figure 2.7 shows a simple automaton of the LTL formula, $\Box(A \rightarrow (\Diamond B))$ for a rule "Whenever A occurs B will eventually occur". The automaton is straightforward. It consists of two states S_0 and S_1 where S_0 is the final state. For an occurrence of A, the automaton moves from S_0 to S_1 where it stays indefinitely until B occurs and moves to the final state. $\Box$

4 Communication Infrastructures for Executing *DocWF*s

There are two general types of communication approaches to assure data exchanges and coordination between actors during an execution of workflows, the centralized approach and the decentralized approach.

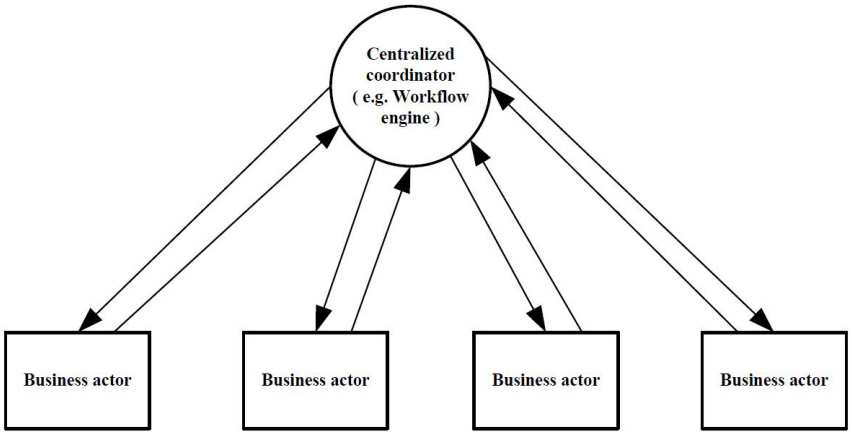


Figure 2.8: Centralized workflow execution infrastructure relying on a dedicated coordinator.

4.1 Centralized Communication Infrastructure

In a centralized setting, a dedicated entity hosts the coordinating workflow engine and is in charge of routing messages and distributing documents between business actors based on a predefined sequence of business tasks (i.e., a priori defined) [AMAA97, AS96], as depicted in Figure 2.8.

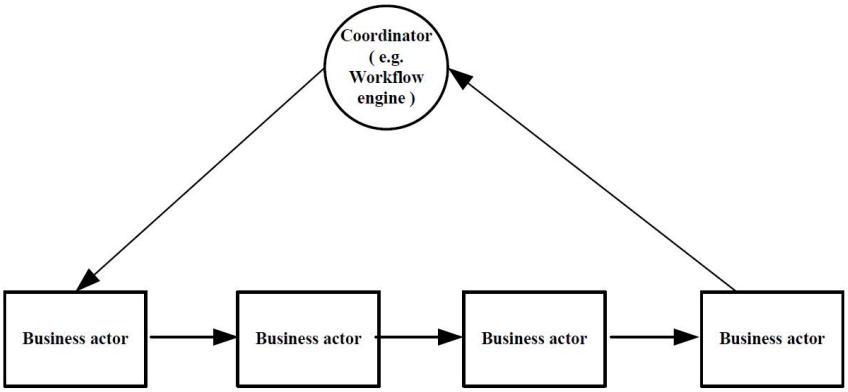


Figure 2.9: A decentralized workflow execution infrastructure of a priori known workflow (in a sequence of tasks) and business actors.

4.2 Decentralized Communication Infrastructure

The responsibility of data and message exchanges and coordination tasks of a dedicated coordinator can be delegated to other entities, for instance, to workflow actors (i.e., business actors) [MM05] and to a distributed set of coordinators in the decentralized approach [AAM+95, GAC+97]. For the former, the business actors must know each other a priori at least for defining precise business tasks and initiating the workflow afterwards [MM05]. This is depicted in Figure 2.9 (arrows between actors). Regarding the latter, on the contrary, for agile workflow scenarios of Chapter 1, business actors may not have prior trust relationships yet they want to collaborate by exchanging data and messages. One way to address this is to enable a loose coupled communication among the business actors by leveraging Publish/Subscribe methodologies. In this decentralized communication context, a Publish/Subscribe based document exchange relying on a distributed set of coordinators is developed in Chapter 5 and is depicted in Figure 2.10.

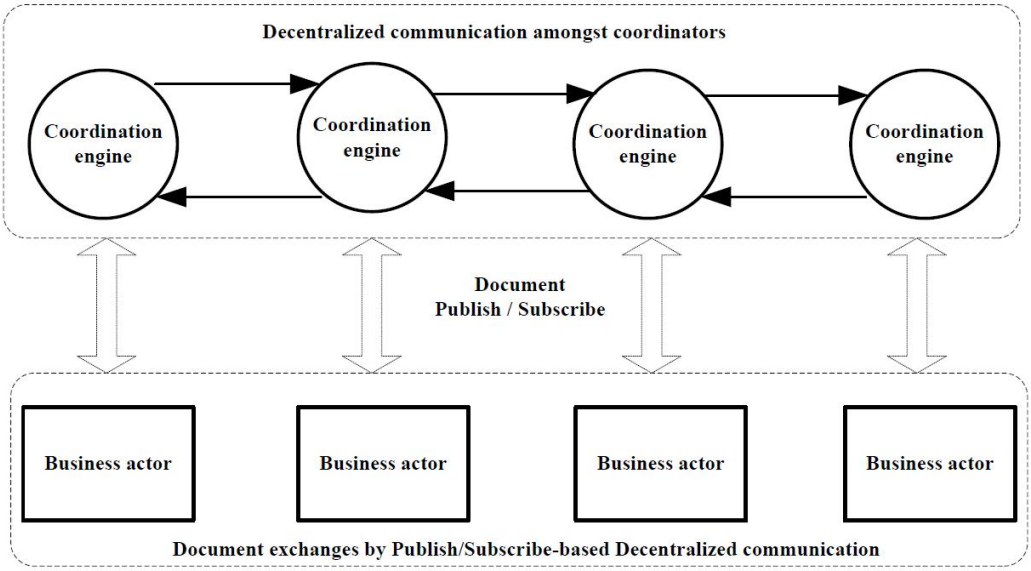


Figure 2.10: A Publish/Subscribe-based decentralized communication infrastructure for *DocWF*s.

The basic idea of the Figure 2.10 is that a priori unknown business actors, i.e., peers, (bottom part) exchange documents through a dissemination layer (upper part) where peers publish documents. The dissemination layer then takes charge of document distribution and coordination among actors. In particular, depending on the subscriptions of peers one coordinator can route documents to other coordinators which then deliver documents to the legitimate recipients. Using an SOA terminology, the dissemination layer can be compared with a set of enterprise service buses (ESB) that route documents among them. In Figure 2.10 the dissemination layer, however, differs from the typical ESB setup mainly in that it not only delivers documents to legitimate peers but also routes documents amongst coordinators and acts as a delegate of the document providers for fine grained document access control enforcement (details in Chapters 5 and 6). Given these, a Publish/Subscribe-based decentralized communication infrastructure is

developed in this thesis for loosely coupled and distributed document exchanges between peers. We further clarify the position of the decentralized communication infrastructure for *DocWF*s with respect to the Publish/Subscribe methodology in the following section.

4.3 Publish/Subscribe-based Decentralized Communication Infrastructure for *DocWFs*

The Publish/Subscribe methodology is one convincing paradigm for large scale distributed systems with many to many interaction and loose coupling among the interacting entities [EFGK03, BESP08]. There are mainly three kinds of Publish/Subscribe methodologies [PE] that can be used for decentralized communication between peers:

- **Topic-based:** This is the classic Publish/Subscribe interaction model (for instance JMS [JMS]) that basically resembles a group-based collaboration [Pow96]. Subscribing to a topic, T, can be viewed as becoming a member of a group T. However, a subscriber might be interested in messages with more fine granular properties of T but may end up receiving more coarse-grained messages and vice versa [PE]. This is known as a limited expressivity problem leading to an inefficient use of bandwidth.
- **Content-based:** Content-based Publish/Subscribe variants (e.g., SIENA [SIE], WS-Notification framework [WSN]) enable subscribers to describe runtime properties of messages they wish to receive as opposed to static topics. This gives more expressiveness and flexibility by removing the limitations of statically defined distinct topics. Messages are therefore not classified according to arbitrarily fixed criteria. However, it introduces other inefficiencies such as event matching with subscription patterns that may need to be performed as close to the source as possible [PE].
- **Type-based:** In this approach, producers publish message objects on a communication bus, and subscribers subscribe to the bus by specifying the types of the objects they are interested in rather than specifying any runtime properties [PE].

As we describe in Chapter 5, the developed Publish/Subscribe-based decentralized communication infrastructure to support loosely coupled document exchanges during an execution of *DocWF* applications is similar to the type-based paradigm. In particular, an ontology *concept* representing the semantics of the document content is therefore similar to a type of a message object and mapped XML document portions associated to the *concept* are the message objects. Publishing a document results into delivery of these document portions like an event notification to a recipient peer. Therefore, such an ontology-driven communication infrastructure decouples document providers and consumers by providing increased stability and agility:

- **Stable data exchange interface:** Actors can subscribe ontology *concept*s to receive semantically equivalent documents that trigger further processing on part of subscribers in

a *DocWF* execution. Therefore, *DocWF* actors can dynamically join or leave without requiring any modifications to the existing infrastructure.

- **Agility of peer information structure:** Actors can publish documents with varying vocabularies associated to an ontology *concept* where the ontology represents business domain *concept*s and their relationships. Thus, an actor can independently modify its data model without affecting the common data exchange interface with other peers.

5 High Volume Business Document Processing for *DocWFs*

For enterprise scale workflow-based collaborative applications such as enterprise resource planning (ERP), supply chain management (SCM) and customer relationship management (CRM), very often peers deal with large documents. These documents need to be processed (i.e., created, accessed, compared, stored) before advancing in a *DocWF* execution. We deal with this set of operations which can be termed distributed document handling (Chapter 4). Business processes being increasingly collaborative, they require document exchanges in intra- and inter-organizational domains. One example of such a collaboration is the outsourcing of business tasks such as the maintenance of recruitment process, the provision of HR services, or conducting vendor certification process of large organizations (e.g., HP, VOLVO, SAP, IBM) to third parties (e.g., www.prometric.com[1]), all tasks that require regular information dissemination to the outsourcing company.

In this section, we discuss the fundamental techniques related to this processing such as representation of these documents into memory (i.e., their parsing) and versioning of documents (i.e., comparing). To this end, we motivate the requirements of a special parsing and of a comparison technique that will be elaborated in Chapter 4.

5.1 Enterprise XML Documents for *DocWF*s

The advent of cross-organizational communication based on XML processing standards such as XML schema, XSL [XSL99], SOAP [GHM+], WSDL [EC01], or BPEL leads to an increasing number of business related XML document exchanges through communication infrastructures such as internet. As can be verified for enterprise applications, these documents may have a considerable size, complex structure, and rich semantics. We term such documents as "Enterprise XML". Structurally, it has following two main characteristics:

- **Large:** One such "Enterprise XML" document of a SAP purchase order can be found in [SAPa]. This schema consists of 442 element definitions, of which 36 may occur

[1]Prometric maintains all HP and IBM certification exam materials, Netmedia maintains all staff recruitment information of IBM.

unboundedly.

- **Composite:** An Enterprise XML document can be composite where different portions of the document may originate from varying business boundaries and thus consolidates data content of different semantics in a single "Enterprise XML" document.

Given these, a *DocWF* application very often deals with "Enterprise XML" and in the course of this thesis we take such documents into consideration for developing solutions for interoperability, communication infrastructure and security. As document exchange is the sole communication means in a *DocWF* application, "Enterprise XML" processing in terms of parsing and comparing different versions has direct effects for non-disruptive collaboration between peers.

5.2 Enterprise XML Processing in a *DocWF* Application

For any "Enterprise XML" processing, the related document part must be in memory for useful processing. There are two general ways to process XML documents: tree and event based parsing. Tree-based API includes the document object model (DOM) [HWW], the Java optimized DOM (JDOM) [JDO], and event-based API includes the simple API for XML (SAX) [SAX] and the streaming API for XML (StAX) [STA] that are the de facto standard APIs for XML processing.

5.2.1 Tree-based processing of Enterprise XML

DOM and JDOM require the whole document to be in memory whereas the amount of memory required for concerned document portions might be smaller. For example, an empty element <e/> (4 bytes for the source file) could easily take 200 bytes of tree storage for these 4 bytes of source with empty information in Java [Kaya]. JDOM optimizes the representation of XML nodes in memory by avoiding unrelated nodes yet it needs to parse the whole document before the application can do any useful processing. A typical usage scenario of tree based parsers is the random access to XML documents. However, in real world usage, the random access to the XML documents is hardly ever utilized. Most importantly, it is a major performance inhibitor as shown by the event based parsers described next.

5.2.2 Event-based processing of Enterprise XML

SAX and StAX require only the current document node[2] to be in memory. An associated event is raised which any application specific event listener can then process. With respect

[2] XML elements, attributes, comments, spaces.

to memory space and processing time, event based parsing (e.g., SAX, StAX) outperforms DOM [SAX]. However, if document updates (adding elements, attributes, changing them) are required frequently, DOM is better than event based parsing [Sidb] as tree based API (e.g., DOM) preserves the hierarchical structure of the XML documents.

5.2.3 *DocWF* Approach for processing Enterprise XML

As a *DocWF* application is collaborative, thus update-intensive and deals with "Enterprise XML", we face contradicting requirements of consuming less memory and low processing time vs maintaining a hierarchy of document nodes to do the updates. To address these contradicting requirements we propose a hybrid approach. In particular, purely view based application scenarios where updates are not required should follow the event based technique. For update intensive applications akin to any agile workflow collaboration, event based parsing can be used to get an event for each node for which tree based parsing (e.g., DOM) will then be used to store the children nodes of the parsed node in a temporary FIFO queue. Taking the advantage of the preserved natural order of a FIFO any extra memory representation for preserving, for instance, parent-child and sibling relationships are not required any more in this approach. This approach is used during semantic annotation of "Enterprise XML" to have a common interpretation of documents (i.e., interoperability) between actors and elaborated in Chapter 4.

To ensure the convergence of documents as mentioned in Chapter 1, a comparison technique between multiple versions of an "Enterprise XML" is also developed in Chapter 4 as motivated next.

5.3 Comparing Enterprise XML in *DocWFs*

As mentioned in Chapter 1, despite individual updates of a document by multiple peers, the legitimate peers always should access a consistent document. The pre-requisite to have a consistent document in a *DocWF* application is to compare different versions of a document produced independently by various interacting peers. The solutions described in [VCFS00] exactly suit this need where the authors showed how to keep a group of collaborating peers up-to date in real time with respect to a shared data object, for instance, a document or document portion. However, the basic criteria of distinguishing that one document is different from the other and thus comparing them specially when considering "Enterprise XML" and very often confidential documents is not well understood. To compare "Enterprise XML", we developed algorithmic solutions that are equally applicable for confidential documents (Chapter 4).

6 Secure Document Exchanges in a *DocWF* Application

When exchanging "Enterprise XML" documents in a *DocWF* execution, various *DocWF* specific processing can be triggered in a recipient side depending on the received document content. This processing includes, for instance, interpreting the semantic annotations, determining suitable business tasks and their sequence complying with some business rules and binding to actual services as shown before. As such in a *DocWF* application security issues are shifted from typical task authorization concerns to documents, for instance, access control for documents and their integrity protection. Moreover, the communication infrastructure being the actual document exchange medium between a priori unknown peers raises several other concerns such as key management which we describe in Chapter 6. In the following, we summarize XML-based document security mechanisms from the literature in six different classes and which we discuss in relation to *DocWF*s:

1. Structure-based Document Access Control,
2. Semantic-based Document Access Control,
3. Document Integrity Protection,
4. Filtering-based XML Access Control Enforcement,
5. Cryptography-based XML Security Enforcement and
6. SOAP Message Level Security.

6.1 Structure-based Document Access Control

In this category, document providers specify their policies over the document schemas (i.e., XML schemas) or document instances. Given a request or interest for a particular document, the provider checks its policy over the schemas associated to the document resulting into either releasing the document or denying the request. An XML schema is annotated with pre-defined attributes that basically set the authorization level for an XML source. The *eXtensible Access Control Markup Language* (XACML) [GM] plays a vital role in this context by providing an XML-based language for specifying the policies and requests. Both the policy specification and requests can be fine-grained down to an XML element or an attribute level. X-Path [CD] and X-Query [BCF$^+$] like expressions are leveraged for specifying such a fine grained source as an XACML object to be protected or requested for instance. Structure-based document access control approaches can be further categorized into the two following sub-categories depending on evaluation techniques of a request:

- **Node filtering:** In this approach separate security views of each XML document are computed for each policy by filtering the unauthorized nodes [DDdV$^+$00, DdVPS02,

Gab05, KH00]. User queries are then evaluated on those views. Although views can be prepared offline, these schemes suffer from high maintenance and storage costs specially for large XML repositories of "Enterprise XML" for instance.

- **Query rewriting:** XML access control by query rewriting [BP06, DCdVMS05, FCG04, LLLL04, MTKH03] removes the shortcomings of node filtering. In this approach, access control policies are not directly applied to the XML sources to be protected; instead they are used to convert potentially unsafe user queries into safe ones which are then evaluated against the original XML source.

It is evident that both of these approaches apply over XML document structures and are suitable when XML schemas and document instances are static and assumed to be shared by both the providers and requesters. Moreover, the policy engine needs to understand the meaning of annotated attributes for all schemas leading to an unmanageable system soon when considering a large number of documents to be evaluated for a large number of users. As discussed in Chapter 1 such a sharing of static XML schemas by a priori unknown peers is not feasible. Moreover, in an environment of varying business boundaries akin to a *DocWF* application assuming a policy engine that is aware of all vocabularies is a far cry.

6.2 Semantic-based Document Access Control

Recognizing the necessity of loose coupling communication of peers and flexibility for XML-based document access control, the authors in [YdmGM05, JWST, PSC03, FJK+08, ASTK06] introduce another level of indirection based on an ontology of *concept*s. This work can be further categorized based on the usage of ontology in general as follows:

- **Using an ontology for representing document content:** In this category [JWST, PSC03], an ontology simply represents the document content in isolation (i.e., irrespective of business domain).
- **Using an ontology for policy representation:** In this category such as in [YdmGM05, FJK+08] policies associated to the document sources are also specified using an ontology.

Both of these techniques aim at distributing documents in a request/response manner and hence first, none of them considers issues related to dissemination of semantically related data or document integrity and confidentiality being identified as key security requirements of the envisioned *DocWF* system. Second, no work yet considers using ontology to represent document content semantics of a business domain and thus combining the agreed business domain vocabularies with the potential business document content. Finally, a combination of ontology representations of both a business domain and policies of document providers at the communication infrastructure level is yet to be designed. Such a combination provides more agility

by enabling loosely coupled document exchanges between peers, peer information structures and peer policy specifications to evolve. The solutions for interoperability and decentralized communication infrastructure described in Chapters 4 and 5 respectively are based on these principles.

6.3 Document Integrity Protection

Integrity protection for an "Enterprise XML" document can be applied in various granularities (from coarse grain (i.e., whole document) to the finest grain (i.e., node) and several schemes can be found in the literature:

- **XML signature:** An XML signature [BBF+] is a special digital signature for XML documents. In this approach, the recipient needs to receive the complete document in addition to the signature value in order to verify the integrity of a received document and thus increases the required processing time and bandwidth for messages.
- **Merkle signature:** The Merkle signature capitalizes upon the natural tree structure of an XML document to recursively compute an accumulated signature from the leaf nodes to the root node [BCF04] of a source XML. Using this technique, the integrity of a document can be verified even if the document is not received in its complete form (i.e., pruned, portion) and thus mitigates the problem of integrity verification in the case of XML node pruning and selective document portion distribution as can be performed by filtering techniques of the envisioned communication infrastructure (detail in Chapter 6).
- **Watermarking-based signature:** This signature scheme is based on watermarking algorithms and initially developed for protecting images [Kun], video [MK] and audio [WCX]. However, it is also gaining momentum for other data sources as shown in [YZL06, NlL05]. It has similar advantages like a Merkle signature scheme has over a typical digital signature scheme. The difference with the Merkle signature is that it allows the signed value to be embedded within an XML document as opposed to attaching it and thus removes the problem of dropping the signature value by a malicious peer.

When dealing with enterprise applications (e.g., SCM, CRM etc.) in agile scenarios, business actors tend to exchange "Enterprise XML" documents. For that purpose Merkle signature seems to be appropriate as discussed in Chapter 6. As ontology driven fine-grained business document distribution is actually realized by sending XML nodes in a selective fashion document integrity protection in a *DocWF* execution becomes more challenging. These challenges are described below:

- **Semantic Integrity:** Peers must verify the semantics of the received content semantics so as to ensure that received document portions represent the expected ontology *concept*s.

- **Structural Integrity:** Peers should be able to verify the structural integrity of a received pruned document with respect to the original one whenever required. Adversaries including malicious peers and communication infrastructure hosts may exchange invalid documents by performing some unauthorized editions in the documents, for instance, updating, deleting, moving and swapping document nodes and thus may violate document integrity. In that context, a peer should also be able to verify that a filtered/pruned document portion that it receives originated from a composite document, a stringent document integrity property that we call "document containment".

The integrity protection solutions span through different chapters of this thesis. In particular, the Chapter 4 lays the foundation techniques for annotating "Enterprise XML" documents with semantic information and comparing them. This annotated documents are then used as inputs in later chapters where Chapter 5 extends annotations by allowing security meta data which are altogether used for later document security verification as described in Chapter 6.

6.4 Filtering-based XML Access Control Enforcement

Filtering illegitimate XML nodes from a source XML document is a basic access control enforcement technique. The XML nodes to be filtered are determined based on an authorization policy associated to the XML document. Depending on the hosting of the XML source and filtering entity the enforcement can be either of the following:

- **Client/Server:** XML document sources are hosted and policies over those are typically provided by the same entity (i.e., servers) in this approach [BLL04, DdVPS01, DdVPS02, FCG04, KMR05, MS03, MTKH03]. This is suitable in closed business boundaries where the filtering is also performed by the same entity.
- **Distributed:** XML document sources may be hosted by trusted third parties as opposed to original providers. Hosted document nodes can already be filtered by the providers [DDdV$^+$00, BCF04] or the third parties may filter on behalf of the providers [DdVPS02, MFBK06, KB08, KE06] and thus a third party can act as a delegate. In all cases third parties deliver filtered XML documents on behalf of the providers. This is mostly done due to the scalability reasons.

Both of these approaches are targeted for structure-based document access control and thus also suffer from similar limitations as described before. Surely, the distributed approach removes the classic vulnerability of the central point of failure of a Client/Server based approach. The solutions for decentralized communication infrastructure and security described in Chapters 5 and 6 respectively are based on the distributed paradigm. In the case of filtering XML nodes by a third party as in [DdVPS02, MFBK06, KB08, KE06], the third party either needs to know the authorization policy specific to the hosted document or should be able to access the

plain text XML nodes in order to determine the potential nodes to be filtered out. In a *DocWF* scenario, however, neither the specific policy is available a priori nor the documents are readable to third parties. As such, communication infrastructure nodes rely on meta data attached with the encrypted XML nodes. In particular, document nodes annotated with semantic and security meta data are hosted by infrastructure nodes which then filter those in a selective fashion based on ontology-based policies of the original document providers. The infrastructure manages peer subscriptions for documents in a *DocWF* execution. However, the communication infrastructure are not trusted for document integrity protection meaning an infrastructure node can or can be used by an attacker to modify the documents maliciously before delivering to peers.

6.5 Cryptography-based XML Security Enforcement

Encryption techniques have been the main choice for years to enforce access control, confidentiality and integrity protection in a fine granular fashion on XML documents either in Client/Server [BLL04, DdVPS01, DdVPS02, FCG04, KMR05, MS03, MTKH03] or in distributed paradigms [BCF04, MFBK06, KB08, KE06]. In a Client/Server environment, the server being the document provider encrypts a document in stipulated granularity before releasing it to the recipients possibly for each single request. In a distributed environment, a trusted third party hosts the encrypted documents and releases those on behalf of a provider. A document provider may encrypt the documents before publishing or let a third party encrypt in case it is fully trusted to the provider. Encryption schemes can be classified in two major types: Public key encryption and secret key encryption, each of them can address one or more such scenarios:

- **Public Key Encryption:** In this scheme, the document provider encrypts a document or document portions using the public key of the legitimate consumer. Only the recipient possessing the associated private key can decrypt the document. For signing a document, the provider encrypts using her private key and associated public key is used for integrity verification.
- **Secret Key Encryption:** In this scheme, two a priori and mutually trusted peers share a pre-computed secret key. Once a document is encrypted with that secret key by one actor, only the other can decrypt the document with the same key.
- **Group-based Cryptography:** In this setting, a group of peers share a common secret which they use to encrypt and decrypt documents. A peer encrypting a document is sure that only the peers in the same group can decrypt the document. The common secret can be computed by a nominated peer which then needs to distribute the key to other peers. Alternatively, the secret key can be computed autonomously by each peer in a distributed fashion and thus no key distribution is required.

As mentioned before, for a priori unknown peers and decentralized hosting of communication infrastructure nodes, key management (i.e., key generation, key distribution, rekeying

and key revocation) is a challenging task in a *DocWF*. For instance, when sending a sensitive document, a document provider can not simply encrypt it with the public keys of potentially legitimate peers simply because they are unknown to her. Moreover, peers can not rely on secret key computations as they do not know each other. For similar reasons, distributing a secret key by a nominated peer is also infeasible. As we elaborate in Chapter 5, a group of interacting peers having the authorizations for the same business *concept* may receive semantically equivalent XML document portions originating from multiple providers. This implies that an authorization for a business *concept* may result into delivering multiple documents to a group of legitimate peers who may not know each other a priori. To address this, we design a distributed key computation technique based on a tree-based group diffie hellman (TGDH) protocol [LLY02] in Chapter 6 that allows interacting peers to compute the common secret autonomously with a partial knowledge of other peers. To be precise, a peer only needs to know some transient cryptographic values associated with intermediate key-tree nodes as opposed to knowing other peers who thus remain anonymous.

6.6 SOAP Message Level Security

In a Web Service-based implementation of a *DocWF* application, peers (i.e., service providers and consumers) exchange SOAP [GHM$^+$] messages that include documents that are published or subscribed and subscription requests. A message may be routed through several intermediaries before reaching its ultimate recipient. In this setting, typical point-to-point security is not adequate and requires an end-to-end multi-hop message security [RSR06]. A document sending peer puts the payload message into the body of a SOAP message and attaches control information associated to the payload such as semantic and security meta data as headers of the SOAP message that can be processed by intermediaries (e.g., communication infrastructure nodes) and the ultimate recipient peer (see WS-Security [NKMHB] for details). The header information remains valid until it reaches the legitimate recipient which can then verify those headers and therefore called SOAP message level security (i.e., end-to-end) as introduced by WS-Security. However, an incorrect usage of headers may introduce some security holes against SOAP messages, for instance, XML rewriting attacks as shown in [BFG04]. In [BFG04], a formal solution by rigorous policy checks is proposed to reduce the possibility of these attacks.

In the context of selective document distribution by communication infrastructure nodes, "Enterprise XML" documents or their portions can be the payload and different meta data can be attached into the headers of a SOAP message for a later verification. To protect such SOAP messages from XML rewriting attacks we developed an efficient solution in [RMS06, RSR06, RS07b] that complements the usage of WS-Security [NKMHB], WS-Policy [WSP]. This solution keeps accounting of legitimate SOAP headers and bundles this accounting information into a special SOAP header we term a "SOAP Account". Thanks to this data structure, the recipient is able to detect XML rewriting attacks before processing the messages containing "Enterprise XML" and thus can save an expensive computation and avoid destructive effects.

7 Conclusion

We introduced most of the basic technical background information that is required for a thorough understanding of the remainder of this thesis. We briefly introduced the basics of task-based workflows, document-based workflows and document-based agile workflows. Document-based agile workflows (*DocWF*) is the underpinning concept to the work presented in this manuscript. We then discussed centralized and decentralized communication infrastructures including Publish/Subscribe methodologies in relation to *DocWF*s. Going further, we discussed "Enterprise XML" processing issues in general and suggested an approach for *DocWF* applications. We also provided an overview of different mechanisms for document security and key management and their pros and cons with respect to a *DocWF* application.

Chapter 3

Document-based Agile Workflows

You can never solve a problem on the level on which it was created.
- Albert Einstein -

1 Introduction

Business organizations collaborate by exchanging business documents in constantly changing business processes. These processes range from typical day-to-day static processes such as purchase order handling to extremely dynamic processes such as merger and acquisition procedures [Fea, ABP, BPT, BAA]. In existing examples of agile workflow business processes, the involved organizations only execute some strictly defined business tasks in a predefined sequence of operations realized by pre-selected services, merely as command followers [MM05, MM07]. None of the examples allows a business actor to determine suitable business tasks and their underlying services dynamically. In this thesis, we propose to go beyond this simple command or service executors by leveraging loosely coupled agile workflow models. Such models are designed by business actors and can be systematically bound to IT components by a loosely coupled execution. Such an execution, first, determines suitable business tasks associated with business goals and then binds those tasks to appropriate IT components, such as Web Services. We suggest agile workflow modeling principles that describe the specifications of such models and runtime principles that describe a dynamic task enactment paradigm for an execution of such models wherein actors do not merely follow a strict execution plan but also actively

participate in the execution. Based on contextual information as captured in business documents, business goals and associated business rules of an agile workflow business process can drive a document-based agile workflow modeling and its execution as opposed to a typical task-based workflow modeling and its predefined execution plan respectively. An actor leverages its domain expertise and organizational capabilities to determine business goals and associated business rules aiming at building viable and value-added processes. This allows business actors to be autonomous in terms of suitable process modeling and their execution and can therefore be called a loosely coupled process modeling approach. Capitalizing on information processing, such an autonomous and distributed execution environment relies on fairly complex document exchanges among business actors and can therefore be regarded as an extension of the typical workflows and be called a document-based agile workflow system (*DocWF*). Therefore, the role of *DocWF* is to assist an actor in her pro-active task enactment to reach goals rather than to instruct her. As such *DocWF* focuses on what can be the tasks to achieve a goal rather than what should be the tasks and their precise execution paths. In this chapter, we focus on such *DocWF* modeling and its execution methodologies.

The reminder of this chapter is organized as follows. In Section 2, we introduce the general problem by eliciting some requirements and an emergency patient care scenario to discuss methodologies of a *DocWF* system. Section 3 gives an overview of a *DocWF* system. Section 4 illustrates the terminologies and introduces the basic principles of a *DocWF* by describing a *DocWF* model and elaborating the relationships of different design and run time entities. In Section 5, we demonstrate *DocWF* modeling principles for the emergency patient care scenario. Section 6 describes *DocWF* model checking and remedy techniques for detected modeling errors. Section 7 describes dynamic task enactment principles for an execution of a *DocWF* model and Section 8 illustrates a dynamic task enactment by showing a *DocWF* execution of an electronic health care record (*EHR*) generation workflow derived from the scenario. Section 9 positions our work with related literature and finally Section 10 concludes the chapter.

2 Problem Statement and Example Scenario

For a routine business scenario like the "tax handling process" a business actor can model specific business tasks and their precise execution sequence during design time using typical task-based workflow principles. Moreover, there exists pre-selected services bound to these tasks for runtime task enactment as shown in Chapter 2. However, in agile business scenarios like "emergency patient care" and "merger and acquisition" business actors lack the knowledge of suitable business tasks and their execution sequences during design time. Business actors can instead determine the business goals and associated business rules at design time. Then concrete business tasks and executable services can be determined dynamically based on the contextual information at runtime.

We proceed further by introducing an agile workflow scenario of emergency patient care in a hospital to explain various principles throughout this chapter.

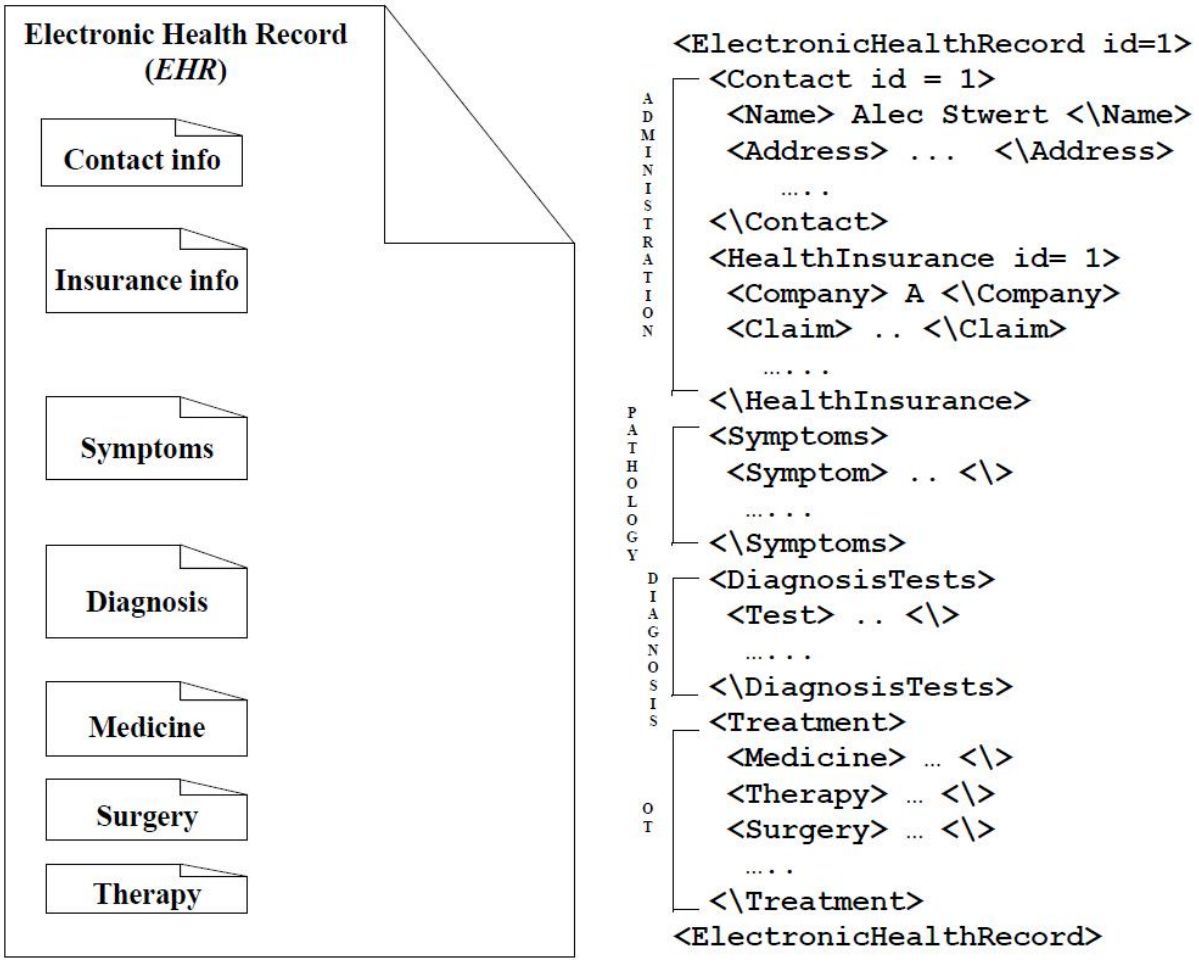


Figure 3.1: A composite electronic health record (*EHR*) document of an emergency patient care scenario.

2.1 Emergency Patient Care Scenario

Let us assume a simplified emergency patient care service of a hospital that comprises four different independent business units, i.e., Administration, Pathology, Diagnosis and Operation Theater (OT). Business units are independent in the sense that each of them have their own goals, rules and procedures to deal with a patient. They rely on individual software systems which exchange documents containing patient information in order to generate a composite document called electronic health record (*EHR*) for each patient as shown in Figure 3.1. For instance, an inventory system of the administration department maintains patients contacts and insurance records. A diagnosis system maintaining all diagnosed results of the patients is used by the diagnosis department. Similarly, other departments have their own dedicated systems to work with. Some of the units possibly are run by third parties and are therefore completely autonomous in determining their course of tasks. As such even though they might have some very general guided procedures they may go through completely different treatment processes depending on the patient's symptoms, diagnosis results as captured in the documents produced by

their software systems. Each emergency patient may have his or her individual peculiarities that need to taken care of differently. Moreover, each software system maintains its own information structure suitable for its purpose and thus business units have their own vocabularies to understand respective patient related records. Given all these, all the involved business actors agree upon an ontology describing the content semantics of the *EHR* document. Such a document includes information about a patient's contact, symptoms etc. originating from the administration department and the pathology department respectively. Any document containing sender specific vocabularies has associated semantic information that allows a recipient to interpret based on the agreed ontology. As such, based on the goals and associated rules a recipient can determine its further tasks and suitable services to execute. The *DocWF* modeling features a loosely coupled high level business process modeling approach using business notions such as business goals and associated business rules and dynamic business task enactment and service discovery so as to meet the following requirements brought up by such agile execution environments:

2.2 Requirements

1. **Loosely coupled models:** A *DocWF* model should not rely on existing services to be bound to business tasks as opposed to typical task-based workflow modeling.

2. **Reuse:** Reuse is desirable not only during runtime task enactment but also during process modeling.

3. **Distributed control:** No centralized point of control is assumed to be present.

4. **Dynamic task enactment:** Enabling task enactment as dynamic as possible by for instance semantic-based service discovery at runtime.

5. **Modeling error detection:** Enabling early detection of design and execution errors due to faulty models and fixing errors without restarting a process.

Loosely coupled models mean that a business expert may model a *DocWF* using her domain knowledge such as business goals, associated business rules without even considering business tasks and any concrete services for instance. As such *DocWF* models not necessarily rely on existing services for actual executions. Regarding reuse, the recent industry adoption of SOA-based business process tools and frameworks [IBMa, SAPb] emphasizes the reuse of existing capabilities. In this context, we introduce the notion of an organizational *knowledge-base (KB)* that contains existing business process models (e.g., BPMN), and their execution history (e.g., executed BPEL) among others. This knowledge is used to determine whether existing tasks and services can be chosen for systematic binding as shown in Figure 1.1 of Chapter 1. For this, a *KB* must be searchable using, for instance, semantic tagging of BPMN [BPM] with goals and documents with semantical *concept*s (detail in Chapter 4) as in [BDW07] and [RRMS09] respectively.

Each actor determines suitable tasks and services to achieve its goals independently and thus control is distributed. Dynamic task enactment requires two important enabling mechanisms: semantic based discovery of available services matching business goals for instance and algorithms for composing these services dynamically for executing a process. As we focus on the basic principles of a *DocWF* modeling and its execution, the main features of these mechanisms and algorithms for dynamic composition are briefly described but their specifications are out of the scope of this thesis. The solutions specified in [MBW07] and [WMD07] featuring a semantic matching techniques of business tasks and a dynamic composition algorithm for services can be used to implement those functionalities respectively. The detection of modeling errors basically identifies inconsistencies in a model and such an error detection is performed because of the two following facts. First, as a *DocWF* is modeled using only high level notions (i.e., business goals and rules) and when the model is executed documents are produced and exchanged between peers. So, first error detection and possible remedy should also be on the modeling level so as to be meaningful to business actors. The other fact is: a high level error detection allows an actor to identify inconsistency even before a *DocWF* deployment and to take necessary remedy steps to make a *DocWF* model consistent.

3 An Overview of a *DocWF*

Similarly to a task-based workflow, a *DocWF* process is modeled and executed by business actors. To make it easier for business actors to proceed with design time modeling the notions of business goals and associated rules are leveraged. Despite being derived from long term strategic business goals, business process models are increasingly executed in dynamic, uncertain, and data centric distributed environment like the emergency patient care scenario. This involves diverse aspects of workflows and their application domains including changes in business goals and rules, confronting exceptional or new scenarios etc. To address this dynamicity during design time a declarative approach is followed to realize a loosely coupled modeling requirement which is described below. To this end, a functional overview of the envisioned components of a *DocWF* run-time system that need to be deployed in a peer site is provided.

3.1 Declarative Approach for *DocWF* Process Modeling

Being an imperative approach, a task-based workflow models a series of all possible sequences of tasks (if known) that are executed at runtime by business actors. As shown in [MPvdA07, vdAP06], such an exhaustive task specification makes a model over specified with limited flexibility. Similarly, specifying all the guard conditions for all the business tasks may also clutter the models and thus be over specified. As mentioned in Chapter 2, one way to reduce this over specification is to use constraint-based declarative workflows [MPvdA07, vdAP06], an approach where constraints simply specify the boundary conditions of the allowed execution paths of a set of business tasks. Clearly, business actors require a priori knowledge of all the business

tasks at design time. However in agile workflow scenarios like treating critical patients, cross-border crime, merger and acquisition, all business tasks and the mentioned constraints may not be known a priori due to their individual peculiarities.

On the contrary, business actors of a *DocWF* simply use the notions of business goals and associated rules to model a *DocWF* process at design time. A business goal is an abstract milestone set by business actors and has to be achieved by an execution of a *DocWF*. A business goal also serves as an abstract definition of business tasks that need to be made concrete at runtime. At runtime, actors of different business boundaries primarily deal with business documents or portions thereof from their creation to their destruction in order to achieve goals, all such processing which we collectively term as document handling. To determine appropriate business tasks and their corresponding realization by concrete services, such document manipulation needs to be performed in a controlled manner by, for instance, enforcing business rules.

3.2 Declarative Business Rules for *DocWF*s

A business rule is a statement that defines or constrains any aspect of a business that may influence the behavior of the business [BR]. The statement can be about day to day things, access control, any business calculation or anything related to the business. Business rules are usually expressed as constraints or in the form "if condition then action". Business rules provide a means to express and specify high-level constraints in the form of policies, which are separated logically and physically from other components throughout business processes [Hua]. Various constraints can be raised from inter-organizational business processes related to business goals dependency, document content and regulations, such as Six Sigma [SS] and Sarbanes-Oxley [SOX] legislation compliance.

As mentioned before, in a *DocWF* business rules are associated to goals as opposed to tasks. Rules are specified as declarative constraints that may not only set conditions for access control or calculation and for restricting execution paths but also set conditions for contextual information, for instance, expected document content, current goals etc. In particular, these rules set boundary constraints over goals and document content and can be expressed using LTL as in [MPvdA07]. These constraints specify restrictions on goals, documents and application specific requirements, for instance, credentials to access documents. We categorize business rules for a *DocWF* process into two main types: functional business rule (FBR) and organizational business rule (OBR).

1. **Functional business rule (FBR):** FBRs capture business domain expertise such as goal precedences. At design time, a *DocWF* actor models a *DocWF* process over goals and associated functional business rules (FBR) independently of any underlying tasks.

2. **Organizational business rules (OBR):** OBRs are constraints over document content and process models that further support in determining suitable tasks/services and as well as

in their dynamic enactment. As such an OBR associated to a goal can be specified also at runtime.

When an actor receives a document from another actor, the recipient's rules are enabled. FBR and OBR being merely constraints allow a runtime task enactment without specifying precise tasks and of their sequence flow at design time.

3.3 A Functional Walk-through

Figure 3.2 depicts the envisioned *DocWF* components in a peer site. It does not consider communication infrastructure between peers and security issues that will be detailed in chapters 5 and 6 respectively. At design time, a business domain expert specifies business goals, their precedences and associated FBRs and OBRs in a *DocWF* model as opposed to modeling a sequence of tasks. This model may be shared with all actors. To leverage previously designed models, their execution history when determining suitable business tasks and their execution sequence a knowledge base is defined as follows:

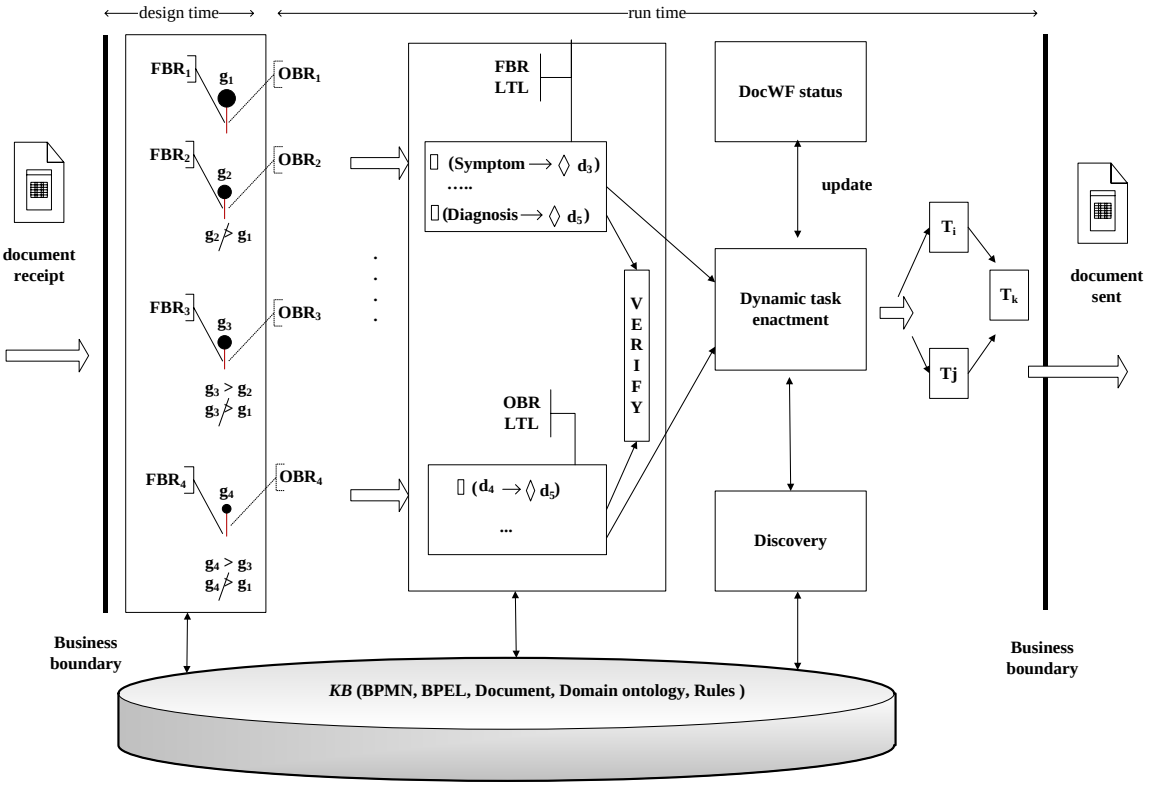


Figure 3.2: Overview of *DocWF* components in a peer site to implement agile workflow business processes.

Knowledge-base (KB): The *KB* of an organization is a collection of business document semantics (e.g., shared ontology), existing business process models (e.g., BPMN), process execution history (e.g., executed BPEL processes). It can be browsed and/or updated during a *DocWF* modeling and its execution time.

As Figure 3.2 shows, at runtime it is possible to represent the rules (FBR and OBR) as LTL formulas which is then transformed into corresponding automata. Based on the automaton of the rules a *DocWF* model and its execution can be verified by detecting modeling errors such as deadlocks and conflicts (c.f. Section 6) that are potentially constituted by the combination of the rules. The dynamic task enactment component can first find possible tasks/services with the help of a discovery component which performs semantic search in the *KB* to match existing business tasks and to bind tasks to services. In case no such tasks/services are found the actor may define/implement the required services (c.f. Section 4). Then the state of existing or invoked tasks/services is modeled in the *DocWF* status component (c.f. Section 7) in order to check their suitability to be bound to business tasks. To do this the *DocWF* status component realizes the task state modeling mentioned in Chapter 2 and that will be detailed in Section 7. Based on the states of the tasks/services, dynamic task enactment is performed, which in turn handles associated documents/document portions before sending those to another actor.

4 Document-based Agile Workflow (*DocWF*) Principles

In the following, we first introduce different *DocWF* terminologies and then discuss a *DocWF* model showing the relationships between those terms.

4.1 *DocWF* Terminologies

Figure 3.3 illustrates the document-based agile workflow terminologies. A *DocWF* system has to achieve a business goal from which a set of subgoals can be derived. Goals are abstract definitions of tasks (called *potential tasks*) that will be executed by actors.

Documents are the main entity of a *DocWF* system where data are instantiated during a *DocWF* execution. A goal achievement instantiates such data flow in documents. This data flow is realized as document handling by performing operations on documents. Based on the achieved and remaining goals and consultation of the *KB*, *potential tasks* are defined and their executions trigger operations on documents which result in either creation of new documents or updates to existing documents by legitimate actors who possess credentials (proving the roles).

Actors may choose tasks from a pool of defined tasks called *recipe tasks* in a predefined sequence which we term as *recipe flow*, if the goals can be achieved by those *recipe tasks* in that *recipe flow*. In this setting, the research proposal granting process (RPGP) of Chapter 2

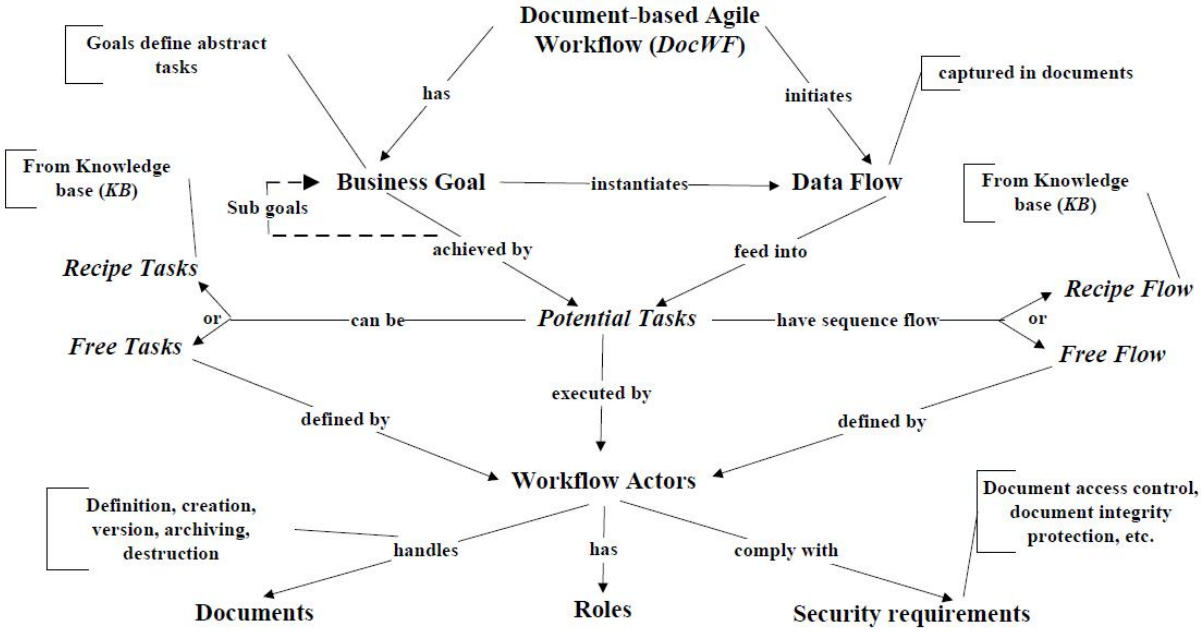


Figure 3.3: Document-based Agile Workflow (*DocWF*) terminologies.

is developed as a *DocWF* system that determines the business tasks and reuses services of a conference proceeding process (CPP) found in the *KB* (illustrated in the following example).

Example 1. A research proposal granting process (RPGP) can follow the conference proceeding process (CPP) as a recipe process from a *KB*. Several actors (i.e., proposal writers, reviewers, granting authority) can handle research proposals in a sequence of *recipe tasks*: publishing call for research proposals by the organizers → submitting several proposals by the authors → reviewing the proposals by reviewers → selecting accepted proposals by the authorities → sending notifications with the reviews to the authors → submitting the final versions by the authors and finally → granting funds for the accepted research projects.

However, there might be exceptional situations as indicated before that can not be handled by any *recipe process* (from *KB*). In this context, potential tasks (i.e., *free tasks*) and their sequence flow (i.e., *free flow*) need to be defined at runtime to achieve the business goal (explained in the next section).

Workflow actors are stakeholders with precise roles in a business process who may also need to comply with security requirements. An actor possesses domain knowledge but may require assistance in the form of meaningful business documents (semantically annotated documents for instance) during a *DocWF* execution to decide upon *potential tasks*. While we consider the security aspects should be integrated for a secure *DocWF*, we do not discuss further these issues in this chapter.

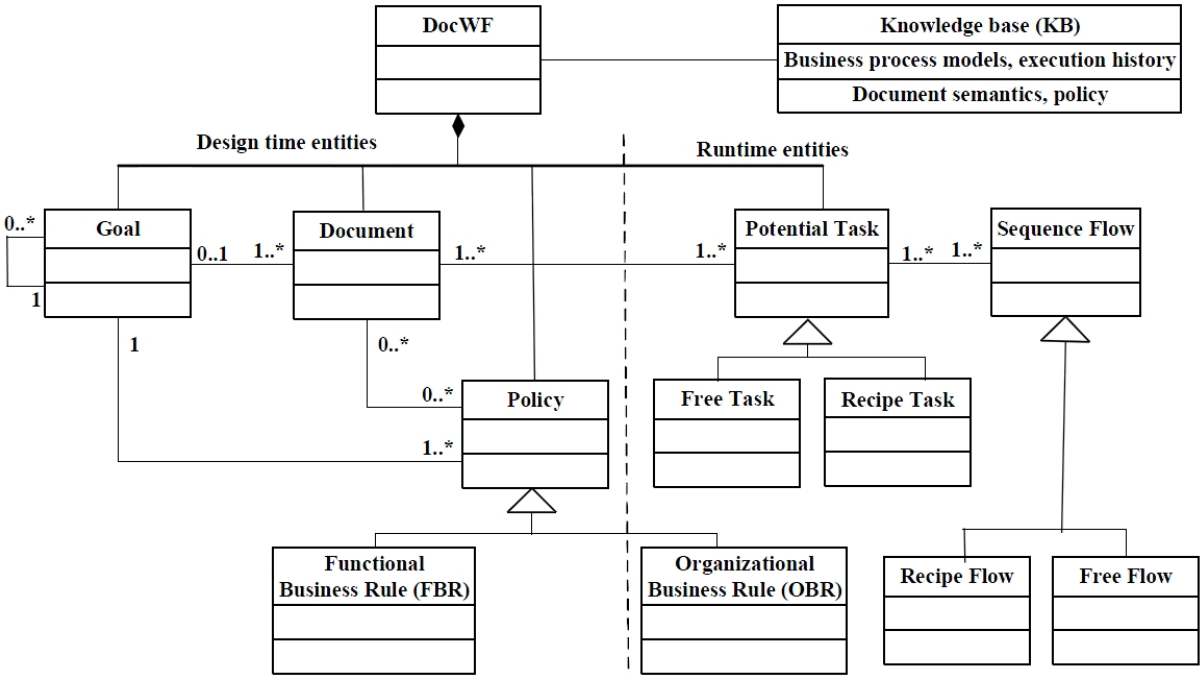


Figure 3.4: Design and run time entities (separated by a vertical dotted line) of the Document-based Agile Workflow (*DocWF*) model.

4.2 *DocWF* Model

An object oriented approach in Figure 3.4 shows the design and runtime entities and their relationships of a *DocWF* model.

4.2.1 Business Goals

Goal: A *DocWF* has to achieve a business goal $\mathcal{G}$ from which a set of sub goals can be derived (i.e., $\mathcal{G} = \{g_1, g_2, ...g_m\}$, for $m \geq 1$).

Goals may have causal relationships, meaning one goal may not be achieved before another goal. For two goals g_i and g_j of $\mathcal{G}$ if g_j can not be achieved unless g_i is achieved then g_j succeeds g_i, denoted by $g_j > g_i$.

The derivation of sub goals and data instantiation upon a goal achievement are represented respectively by a recursive association of goals forming a goal precedence and by another association between goal and document. One goal achievement implicitly enables subsequent goals and documents to be handled (e.g., updated) until the business goal is achieved. No such precedence between two goals and two documents/document portions are denoted by: $g_j \not> g_i$ and $d_j \not> d_i$ respectively.

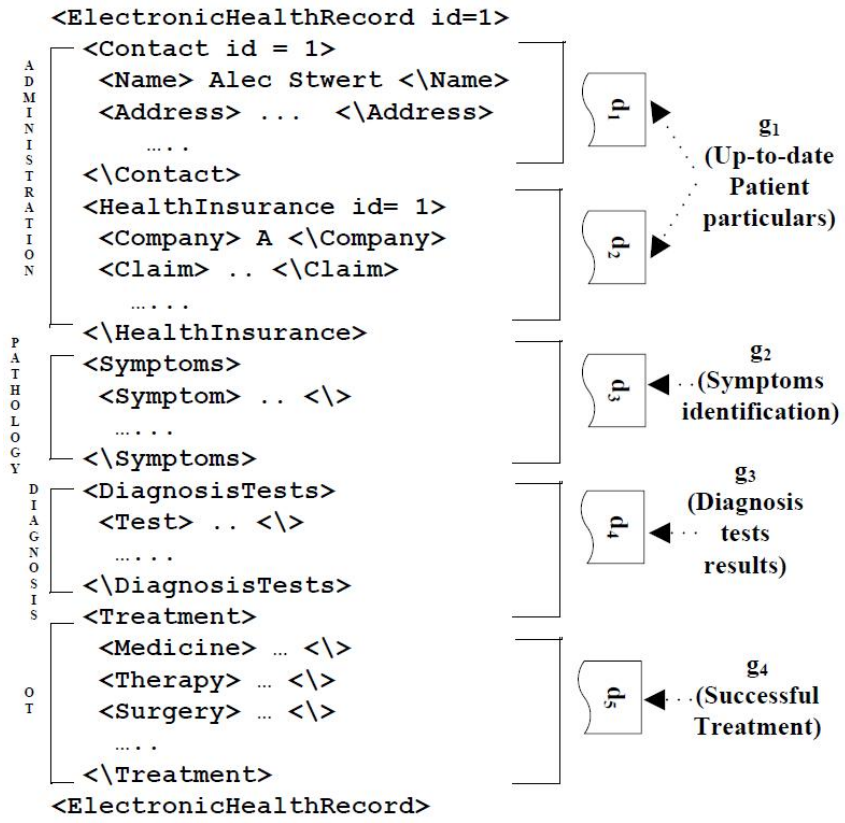


Figure 3.5: Different portions of the *EHR* (d_1, d_2, d_3, d_4, d_5) are updated as *potential tasks* for different goals (g_1, g_2, g_3, g_4,) are executed in a *DocWF* execution.

Example 2. In Figure 3.5, the business goal $\mathcal{G}$ of the *EHR* workflow is to generate a complete *EHR* document D containing patient `<Contact>` d_1, `<HealthInsurance>` d_2, `<Symptoms>` d_3, possible `<DiagnosisTests>` d_4 and `<Treatment>` d_5 records maintained by departments of the hospital(e.g., administration, pathology, diagnosis, operation theater etc.). The derived set of goals is, $\mathcal{G} = \{g_1, g_2, g_3, g_4\}$; with goal precedence $g_4 > g_3 > g_2$ but $g_{[2-4]} \not> g_1$ (see Figure 3.2). A diagnosis result can not be recorded without knowing a symptom, i.e., $g_3 > g_2$. Similarly for a successful treatment record a diagnosis result is required i.e., $g_4 > g_3$. However, patient's contact and insurance data can be recorded at any time i.e., $g_2 \not> g_1, g_3 \not> g_1, g_4 \not> g_1$. □

4.2.2 Documents and their Semantics

Documents: Since a *DocWF* is document centric, an associated document meta model showing the relationships of involved entities in document handling, i.e., domain *concept*s, actors, document portion, and version is depicted in Figure 3.6.

Let D be a set of documents/document portions d_i denoted by $D = \{d_1, d_2, ..., d_n\}$, for $n \geq 1$, that need to be handled in a *DocWF* scenario. If d_j can not be handled before d_i then d_i

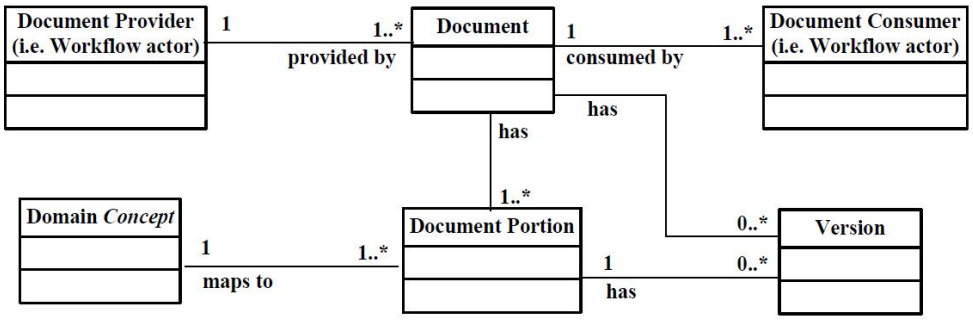


Figure 3.6: *DocWF* Document Meta Model.

precedes d_j, denoted by $d_j > d_i$. A simple task/service or a composition of tasks/services can be bound to achieve a goal that implicitly handles documents/document portions.

Document semantics: Before exchanging document portion instances, an actor can define its individual document data model (e.g., schema) and still be interoperable by associating its document instances to shared business domain ontology *concept*s represented as OWL [OWL]. In the *EHR* generation workflow, individual departments can share a patient ontology *concept*s describing relationships of patient related data as *concept*s and yet can map those to department's documents/document portions. For instance, a business *concept* 'treatment' can be mapped to `<Medicine>`, `<Therapy>` and `<Surgery>` related XML fragments of the operation theater department (Figure 3.1). Such agreed document semantics are a pre-requisite for a *DocWF* system and will be detailed in Chapter 4.

4.2.3 Potential Tasks and their Sequence Flow

Potential Task: As mentioned before, an actor can define *potential tasks* at runtime. The execution of a *potential task* handles documents which is reflected either by creating new documents or updating existing documents. Such a definition of a task can be based on an existing description or can be a brand new description. In both cases such tasks are arranged in a sequence as described below.

Recipe Tasks and ***Recipe Flow***: Previously defined business tasks of a BPMN model or invoked services of a BPEL [BPE] model in a *KB* can be reused as returned by the discovery component of Figure 3.2. The discovery component can, for instance, match current goals with the semantically annotated BPMN tasks [BDW07] in the *KB*. Such tasks/services and their sequence flows are termed as *recipe tasks* and *recipe flow* respectively.

Free Tasks and ***Free Flow***: However, in exceptional situations where no suitable process models or execution history exist in the *KB*, new tasks (i.e., *free tasks*) need to be defined and implemented. Their sequence flow (i.e., *free flow*) can be defined utilizing rules (OBR) over the current goals and document content.

Example 3. For the *EHR* generation workflow (Figure 3.5), consider a doctor orders diagnostic tests for a surgery patient according to a FBR, but cannot wait for the results as patient condition is getting critical. She may start treatment by providing medicine or therapy as satisfying some OBR. As soon as the test results arrive, she might need to achieve a new goal requiring a completely different treatment (i.e., *free tasks*) depending on the result. □

The *recipe* and *free* tasks (and *recipe/free*) are important entities for agile workflow business processes. By finding *recipe tasks* and/or their *recipe flow* of existing business processes of *KB* allows an actor to reuse existing services. If no such tasks/services are found, the *free tasks* and *free flow* elements allow her to define tasks and determine their flow at run time by applying business rules.

4.2.4 Policy for Document Handling and its Evaluation

Policy: The policy base of an organization is formed by functional business rules (FBR) and organizational business rules (OBR). FBRs are defined at design time and describe functional boundaries and relationships of goals. On the other hand, OBRs may be defined at runtime (Figure 3.4) describing organizational concerns such as expected data in documents, their further processing and desired credentials of an actor etc. Effectively, one or more policies can be concerned with a goal. However, a document can be affected by zero or more policies and a policy can affect zero or more documents (see Figure 3.4). Note that the goal precedence mentioned before can also be a rule.

Rules can be represented as LTL constraints internally as shown in Figure 3.2. LTL rules are grounded using operators between operands. Goals and document content represent two basic LTL operands as rules specify constraints about goals and document content. However, any other things related to a business process can also form LTL operands as illustrated below.

- Only goals that effectively represent goal precedences.
- Only document portions associated with business ontology *concept*s that effectively represents document precedences.
- Combination of goals and documents.
- Entities that affect a business process, for instance, required credentials to get access to documents.

The LTL formula is transformed to a Büchi automaton internally by the system as in [MPvdA07]. Transitions to an accepting state of such an automaton satisfies a rule (i.e., *true*) or *false* otherwise. This value may change during an execution meaning a rule that is *false* for the time being may be *true* eventually. In terms of later deployment of a *DocWF* system, UI templates may be used to abstract such LTL formulas from business actors as shown in Figure 3.7. These are illustrated in Section 5.

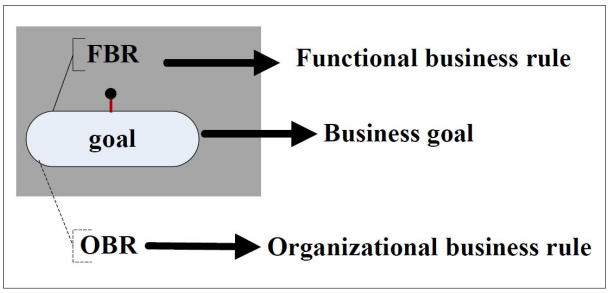


Figure 3.7: A *DocWF* model UI template.

4.2.5 Knowledge-base (KB) Update

A *DocWF* execution trace (i.e., sequence flow, executed BPEL), business document's semantics (i.e., domain ontology) and associated rules can be added to the *KB* by the actors and thus the *KB* gets continuously enriched for later consultation. To enable an actor to consult its *KB* including updating it whenever needed, the *KB* is not associated to any design or run time entities (see Figure 3.6).

5 Modeling an *EHR* Process using *DocWF*

Now that basic principles of a *DocWF* are described, we illustrate the loose coupled modeling for the *EHR* process of Section 2. It also includes business rule representation using LTL formulas and transformation into Büchi automata [LTL] for later modeling error detection.

5.1 Determining Goals and Business Rules of *EHR* Process

An emergency patient care management expert (i.e., actor) models an *EHR* process using the business goals of the pathology ("Symptoms identification"), diagnosis ("Diagnosis test results") and operation theater ("Successful treatment") departments as shown in Figure 3.8. Each business goal is also associated with corresponding FBRs which are "Symptoms must be identified if exists", "Symptom identification must be preceded by a diagnosis result" and "Treatment needs to be performed for each diagnosis result" respectively. As mentioned before this model can be shared amongst the actors and in this case amongst pathology, diagnosis and OT departments. Note that, for simplicity, Figure 3.8 does not show any goal and rule of the administration department.

As mentioned in Section 3, OBRs specific to a department may also be specified upon a document receipt at runtime. For instance, when a new patient gets admitted, the administration department records the patient's contact (i.e., `<Contact>`) and insurance information

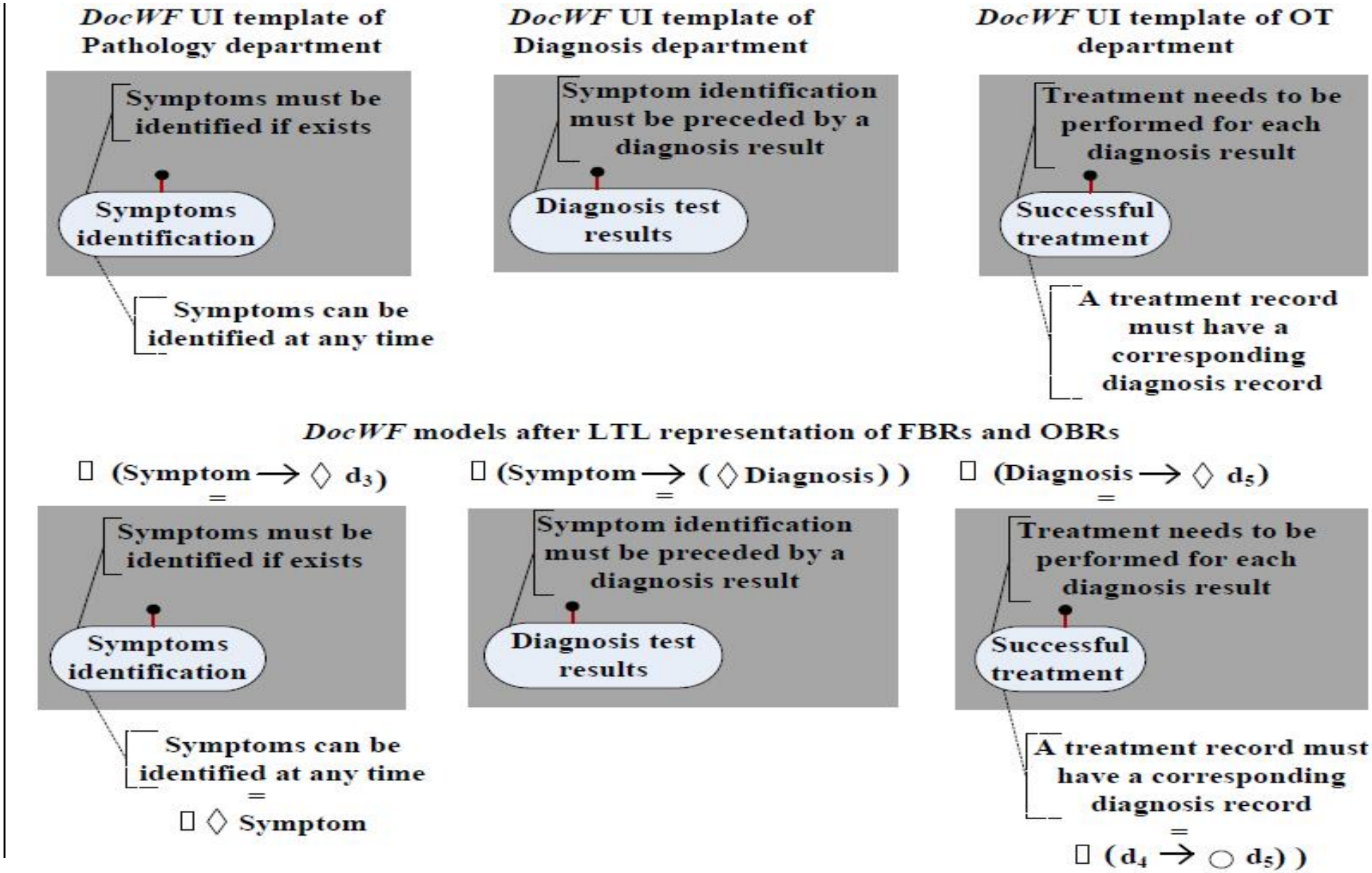
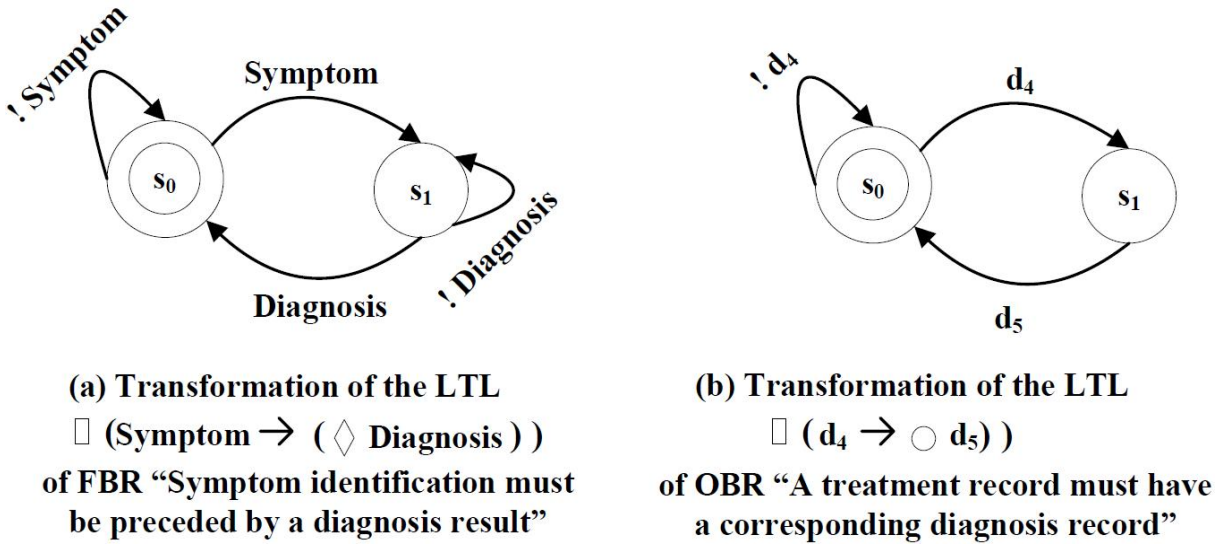


Figure 3.8: *DocWF* models of *EHR* process for the Pathology, Diagnosis and OT departments.

(i.e., <HealthInsurance>) and sends those to pathology department for symptom identification. An OBR, "Symptoms can be identified at anytime", associated to the goal of pathology department is enabled which allows further processing in this case. On the other hand, for the goal "Successful treatment" the associated OBR "A treatment record must have a corresponding diagnosis record" is specified at design time realizing the fact that treatment can be done without a diagnosis as captured as a goal dependence, i.e., $g_4 > g_3$ in Figure 3.2.

5.2 Rule Representation and Evaluation for the *EHR* Process

In this section, we illustrate the Büchi transformation of a LTL representation of a rule and the evaluation of the rule.



(a) Transformation of the LTL □ (Symptom → (◇ Diagnosis)) of FBR "Symptom identification must be preceded by a diagnosis result"

(b) Transformation of the LTL □ (d4 → ○ d5)) of OBR "A treatment record must have a corresponding diagnosis record"

Figure 3.9: Transformed automata of a FBR and an OBR for the goals g_3 and g_4 of Figure 3.8 are (a) and (b) respectively.

5.2.1 Büchi Transformation of a LTL Representation

The FBR, "Symptom identification must be preceded by a diagnosis result", of the goal, "Diagnosis test results", of Figure 3.8 can be interpreted as the rule, "A diagnosis eventually occurs after a symptom identification". Then the LTL operands for the FBR are two goals, "Symptoms identification" and "Diagnosis test results", as represented by "Symptom" and "Diagnosis" in the LTL formula: $\Box$(Symptoms$\rightarrow\Diamond$Diagnosis). The OBR, "A treatment record must have a corresponding diagnosis record" of the goal, "Successful treatment", can be interpreted as the rule "A diagnosis record must be followed by a treatment record". Intuitively the LTL operands for this OBR are two document portions d_4 and d_5 of the *EHR* document representing <Test> and <Treatment> records respectively as represented by d_4 and d_5 in the LTL formula: $\Box(d_4 \rightarrow \bigcirc d_5)$.

The transformed Büchi automata of the FBR and the OBR are illustrated in Figure 3.9 (a) and (b) respectively. Each automata consists of two states s_0 and s_1 where s_0 is the accepting state. In Figure 3.9 (a), for an identification of a symptom, the automaton moves from s_0 to s_1 where it stays indefinitely until a diagnosis occurs and moves to the final state. In Figure 3.9 (b), for a record of diagnosis in d_4, the automaton moves from s_0 to s_1 from where it moves immediately to the final state after recording a treatment information into d_5.

5.2.2 Rule Evaluation

In Figure 3.8, for the FBR of the goal "Symptom Identification", the operands are Symptom and *EHR* document portion d_3 of Figure 3.5 and represented as $\Box$(Symptom$\rightarrow\Diamond d_3$). To illustrate a LTL rule evaluation in Figure 3.8, consider the goal of the operation theater (OT) department, the associated FBR is "Treatment needs to be performed for each diagnosis result". The corresponding LTL formula is $\Box$(Diagnosis$\rightarrow\Diamond d_5$) where "Diagnosis" and d_5 refer to the associated goal, "Diagnosis test results" and document portion containing treatment records, i.e., `<Treatment>`, respectively. This implies that each diagnosis result, i.e., `<Test>` record will eventually have a corresponding `<Treatment>` record in Figure 3.5. In particular, for some diagnosis results, corresponding treatment data may not be filled in d_5 until the corresponding treatment is successful. In Figure 3.8, an OBR of the pathology department is "Symptoms can be identified at any time", i.e., $\Box\Diamond$ Symptom which can be *true* after a patient's arrival.

6 Detecting Modeling Errors of a *DocWF*

For a correct execution of a *DocWF* model, it is important that the model does not contain errors and its current execution is correct. In the verify component of Figure 3.2, an actor can check the current model and execution so far before determining suitable tasks and then concrete binding of tasks to services. We define a consistent *DocWF* model with respect to its execution as follows:

- **During an execution:** The execution of a *DocWF* is correct at some point of time if all the constraints representing the rules evaluate to *true*.
- **After an execution:** Similarly, at the end of an execution if all the constraints evaluate to *true*, the execution has completed successfully.

The model checking detects inconsistencies in a *DocWF* model and captures them as either deadlocks or conflicts so as to be meaningful to business actors. Inconsistencies can be detected by checking the transitions and states of an automaton. The absence of any transition in the automaton of the rules for some goals is considered to be a deadlock. Also if the automaton has

Modeling Error	Deadlock	Conflict
Design time error	***No transition*** exists of the combined automaton of the ***FBRs***	***No state & transition*** exists of the ***FBRs***
Run time error	***No transition*** of the combined automaton of the (***FBR & OBR***) or ***OBRs*** exists	***No state & transition*** of the combined automaton of the (***FBR & OBR***) or ***OBRs*** exists

Figure 3.10: Detecting deadlocks and conflicts in a *DocWF* model at design and runtime.

no state nor any transition, then there is a conflict. A deadlock or a conflict may occur at both design and run time and can be reported to actors accordingly. These are depicted in Figure 3.10 and illustrated below.

- **Design time deadlock and conflict:** These may occur between FBRs. In particular, if an automation for any two FBRs results in no further transition then these two FBRs constitute a deadlock. In addition to no transition if there exists no state then there is a conflict formed by the FBRs.
- **Run time deadlock and conflict:** These may occur in a combination of FBRs and OBRs or only OBRs. In particular, if an automation for a FBR and an OBR or two OBRs results in no further transition then the combination constitutes a deadlock. In addition to no transition if the combination does not correspond to any state then there is a conflict formed by the combination.

A deadlock and a conflict detection in a *DocWF* model is illustrated in Figure 3.11 (a) and (b) respectively and described in the following. Compare to Figure 3.8, Figure 3.11 shows another *DocWF* model (with different FBR and OBRs) associated with another patient for instance. It implies the fact that depending on patients or diseases, *DocWF* models may also vary and thus a *DocWF* system can be extremely agile.

6.1 Deadlock Detection

In Figure 3.11 (a), consider the FBRs "Symptom identification must be followed by a diagnosis" and "Symptom identification must be preceded by a diagnosis record" of the goals "Symptoms identification" (g_2) and "Diagnosis test results" (g_3) respectively. If g_2 is achieved then g_3 can never be achieved as the FBR of g_3 states a completely opposite goal to the corresponding FBR of g_2. In particular, the first FBR says that every symptom record must be followed by an associated diagnosis record, i.e., $d_4 > d_3$, whereas the latter FBR says that there should not be a symptom record before a diagnosis result is recorded, i.e., $d_4 \not> d_3$. So according to Figure 3.11, after achieving the goal g_2 the automaton will have no transition and thus the system would be in a deadlock for a design time error.

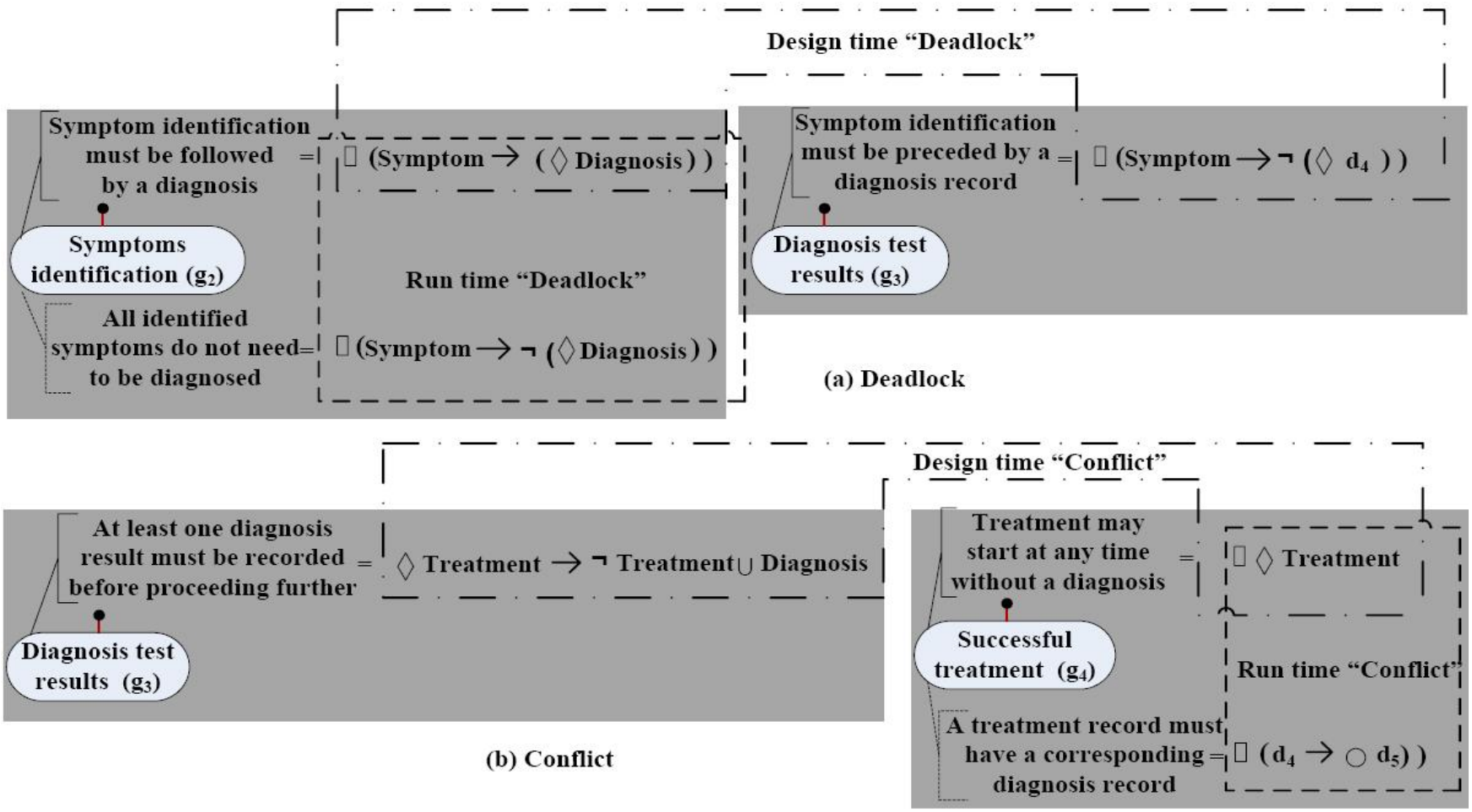


Figure 3.11: (a) Design time deadlock of two FBRs of goals "Symptom identification" and "Diagnosis test results"; and Runtime deadlock of FBR and OBR of the goal "Symptom identification". (b) Design time conflict of two goals "Diagnosis test results" and "Successful treatment"; Runtime conflict of FBR and OBR of the goal "Successful treatment".

Similarly, a runtime deadlock may occur when the actor specifies the OBR "All identified symptoms do not need to be diagnosed" that contradicts with the corresponding FBR "Symptom identification must be followed by a diagnosis" of g_2. In particular, the combination of the FBR and OBR constitutes an automaton that does not have any transition once the FBR evaluates to true.

6.2 Conflict Detection

In Figure 3.11 (b), a design time conflict occurs for the FBRs "At least one diagnosis result must be recorded before proceeding further" and "Treatment may start at anytime without a diagnosis" of the goals "Diagnosis test results" (g_3) and "Successful treatment" (g_4) respectively. This is because the combination of both rules constitutes no states and transitions in the automaton. A similar situation arises for the FBR and OBR of the goal g_4 at runtime leading to a runtime conflict (see Figure 3.11 (b)).

6.3 Remedy

The first step of a remedy is to notify concerning business actors as soon as any modeling error is detected. Then actors can change the rules so as to avoid further errors. Depending on the type of errors, actors may change FBRs or OBRs as shown in Figure 3.12. The updated *DocWF* model must be checked again in order to be sure that changes do not introduce any new deadlock or conflict.

	Deadlock Remedy	Conflict Remedy
Design time remedy	***Change*** the immediate FBR and ***check*** again	***Change*** any of the FBRs and ***check*** again
Run time remedy	***Change*** either the FBR or the OBR and ***check*** again	***Change*** any of the FBRs or OBRs and ***check*** again

Figure 3.12: Remedy after detecting deadlocks and conflicts of *DocWF* models at design and runtime.

To illustrate a design time remedy, consider the design time deadlock in Figure 3.11 (a), by changing the FBR "Symptom identification must be preceded by a diagnosis record" into, for instance, "A diagnosis record should have a corresponding symptom record" would avoid the deadlock. In particular, the changed FBR would not contradict with the FBR, "Symptom identification must be followed by a diagnosis" of the goal "Symptom identification", g_2.

As shown in Figure 3.12, to resolve a design time conflict any FBR can be changed. To illustrate, for instance, consider the conflict of Figure 3.11 (b), changing the FBR "Treatment

may start at anytime without a diagnosis" of the goal g_4 into "Treatment can start only after a diagnosis" would avoid the conflict. In particular, the new FBR of the goal g_4 would coincide with the FBR of the goal g_3 and as such the automaton after the goal achievement of g_3 still evaluates to *true* for the new FBR.

7 Dynamic Task Enactment

Once an agile workflow scenario is modeled and shared with business actors, the model is ready to be executed by concerned actors. This section describes the execution semantics of a *DocWF* model.

7.1 Determining Suitable Business Tasks and Service Binding

To find suitable business tasks and their concrete binding to services, the dynamic task enactment component of Figure 3.2 performs searching in the knowledge-base (*KB*) of the peer. This searching is a two step process:

1. **Determining business tasks:** Determine possible tasks by performing a semantic matching of current goals with the corresponding annotation of business tasks in the BPMN model of a *KB*.
2. **Binding to services:** Perform similar matching by, for instance, semantic service discovery to bind a task to concrete services.

This process is illustrated in Figure 3.13 for pathology and diagnosis departments of emergency patient care scenario. For the goal "Symptoms identification", the task "Identify symptoms" of a BPMN model in the *KB* is chosen as its annotation "Symptom identification" matches with the goal. Similarly, a semantic web service "Pathology Service" from a previously executed BPEL of the *KB* matches with the task "Identify symptoms".

As also mentioned in Chapter 2, several services can be found after performing the above-mentioned steps. However, not all services may be reachable due to various reasons, for instance, communication or system failure or an overload of requests etc. To bind to suitable services, the dynamic task enactment component of Figure 3.2 captures these uncertainty issues in a formal task state model. In particular, a task or service (both terms being used indifferently now onwards) suitability and previously performed task execution information are modeled as task states and as a matrix representation respectively so as to provide a global view to a peer with respect to achieved goals and further goals to be achieved. This is described in the following sections.

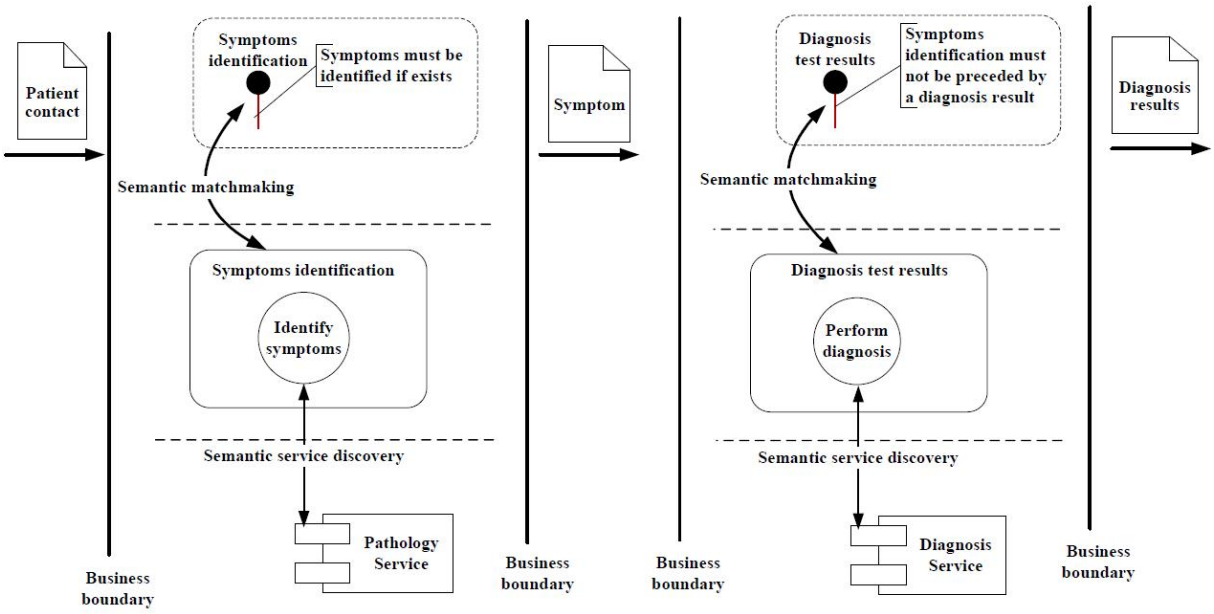


Figure 3.13: Determining suitable business tasks and binding to services by pathology and diagnosis departments.

7.2 Task State Modeling

In order to capture the state of a task or service, we model the possible states of a task using simple integer values. Depending on the state value one task or service can be determined to be suitable for later task enactment.

Let T_i be an executed task for a goal g_i for which the associated rule is r_i and r_j be an enabled rule after T_i's execution. Now, we formally define a *potential task* as follows:

Potential Task: A task T_j is a *potential task* after an executed task T_i, denoted by $T_j > T_i$ if either (1) both tasks are chosen from the *KB*, i.e., are *recipe tasks*, and have the same precedence, i.e., $T_j > T_i$ or (2) both rules r_i and r_j as represented by automata evaluate to *true* (if they don't, they may possibly become *true* later).

Similarly, two *potential tasks*, T_l and T_m, can be executed in parallel if both tasks are chosen from the *KB* and/or both r_i and r_j evaluate to *true* and denoted by $r_{lm} = 0$ otherwise $r_{lm} = 1$. A directed graph $G = (T, >)$ represents these precedence relationships, where each *potential task* is a node in the graph and a directed edge from a node T_i to node T_j represents a sequence flow of tasks, denoted as $f_{ij} = 1$. Given this, we specify below the task state model.

Task state: The state of a task (i.e., executed or potential) T_i, denoted by $S(T_i)$, in a *DocWF* execution, is an integer value in $\{0, 1, 2, 3\}$ such that:

- $S(T_i) = 0$, i.e., T_i is *not* executed before and is *not* a *potential task*.

- $S(T_i) = 1$, i.e., T_i is *not* executed before and is a *potential task*.
- $S(T_i) = 2$, i.e., T_i was successfully executed before and is *not* a *potential task*.
- $S(T_i) = 3$, i.e., T_i was successfully executed before and is a *potential task*.

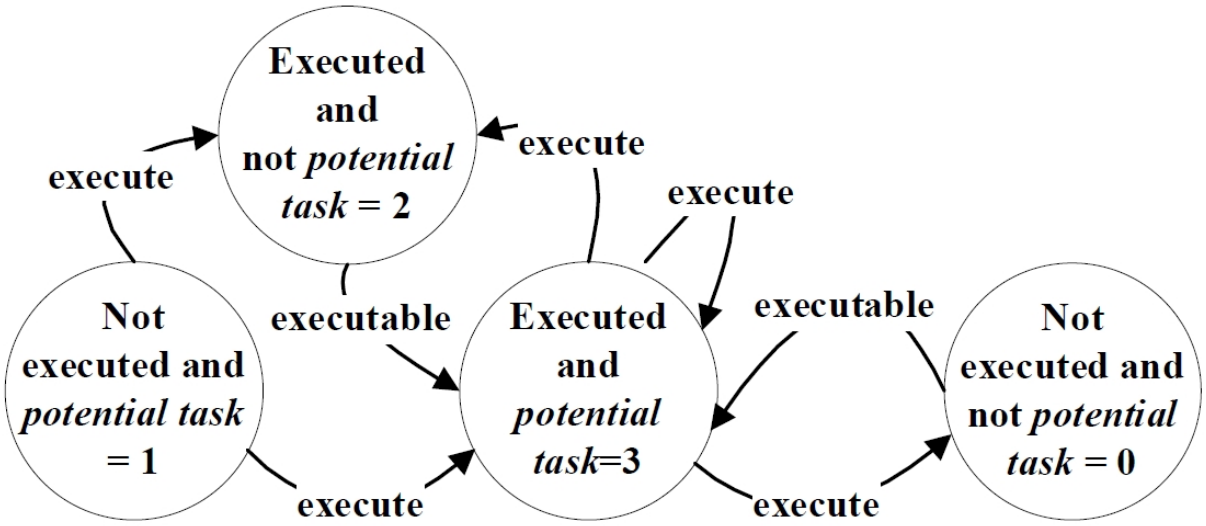


Figure 3.14: Task state model of tasks/services for dynamic enactment.

By the above definition, only those *potential tasks* having a value either 1 or 3 can be executed further (see Figure 3.14). As such, the task state model allows a binding to a suitable service from multiple possible services. In order to provide a comprehensive picture of the execution to an actor, we define a *DocWF status* that represents the current execution status of a *DocWF* system.

DocWF status: A *DocWF status* of a *DocWF* execution is described as an array of the state values of executed tasks. Let S be a *DocWF status* of p number of executed tasks then $S = \{S(T_1), S(T_2), ..., S(T_p)\}$.

Now, we formally define an execution of a *DocWF*.

7.3 A Formal *DocWF* Process Execution Snap Model

An execution snap of a document-based agile workflow is a tuple $DocWFexec$ = ($\mathcal{G}, D, T, F, R$,-S_I, S_F), where

1. $\mathcal{G} = \{g_1, g_2, ...g_m\}$, $m \geq 1$, is a set of goals derived from a business goal $\mathcal{G}$.
2. $D = \{d_1, d_2, ..., d_r\}$, $r \geq 1$, is a set of documents/document portions that need to be handled.

3. $T = \{T_1, T_2, ...T_n\}$, $n \geq 1$, is an incremental set of executed tasks where a task is defined as either a *free* or a *recipe* task.

4. $F = [f_{ij}]_{|T|\times|T|}$ is a sequence flow matrix of the tasks of T. i.e., $f_{ij} = 1$ if $T_j > T_i$ otherwise $f_{ij} = 0$ for $i, j = 1, 2, ...|T|$.

5. $R = [r_{ij}]_{|T|\times|T|}$ is a rule violation matrix of the tasks of T, i.e., $r_{ij} \in \{0, 1\}$ (as defined previously) for $i, j = 1, 2, ...|T|$.

6. $S_I \in \{0, 1, 2, 3\}^{|T|}$ is the initial status of the *DocWF*.

7. $S_F \in \{2, 3\}^{|T|}$ is the final status of the *DocWF*.

As an execution goes on, T is incremented by adding executed tasks. Similarly, the sequence flow matrix F and rule violation matrix R are incremented as executed sequence flows and enforced rules are added. As such T, F and R altogether show the status of executed tasks, their sequence flows and rules applied. Given an initial status S_I, for a task $T_j \in T$, if there is no T_i such that $T_j > T_i$, i.e., $f_{ij=0}$ then the state of T_j is 1, i.e., $S_I(T_j) = 1$; otherwise $S_I(T_j) = 0$. Note that initial status may also consist of state values of executed tasks (i.e., $\{2, 3\}$). As such a *DocWF* execution may start from an unfinished execution as opposed to a task-based workflow which is typically in a blocking situation in case of an exception for instance. On the other hand, in the final status S_F, all the documents/document portions are handled and all the goals and thus the business goal is achieved (implies a *potential task* is executed at least once as denoted by $\{2, 3\}$).

7.4 *DocWF Status* Transition Rules

The dynamic execution of the tasks is governed by a set of transition rules based on which the incremental set of executed tasks T is built. We define general transition rules that take uncertainty and re-usability into account during a *DocWF* execution. While a complete specification of the transition rules are depicted in an Appendix C, we illustrate in the following a dynamic task enactment of the *EHR* workflow relying on the task state model of Section 7.2.

8 A *DocWF* Task Enactment for the *EHR* Generation Process

Assuming no occurrence of modeling errors, we now illustrate a dynamic task enactment of the *EHR* generation workflow by applying dynamic task enactment approach of Section 7 in Figures 3.15, 3.16, 3.17, and 3.18. The actors are hospital administrative employees, pathologists, diagnosis technicians and doctors. Some tasks are bound to human oriented services and some

others to automated services. In the following, the execution of the determined tasks for the *EHR* generation process is illustrated with respect to goals. As shown in Figure 3.5, the business goal is to 'Generate a complete *EHR*' after a treatment of an emergency patient. The sub-goals are to record "Up-to-date patient particulars" (g_1), "Symptoms identification" (g_2), "Diagnosis test results" (g_3), and "Successful treatment" (g_4) with goal precedence $g_4 > g_3 > g_2$ but $g_{[2-4]} \not> g_1$ (see Figure 3.2).

8.1 Up-to-date Patient Particulars

An execution of *EHR DocWF* starts after the arrival of an emergency patient, i.e., task T_0 (Figure 3.15 (a)). Note that the sate of the task T_0 is 3. According to the task state model, it means that this task (i.e., ambulance service) can be executed again allowing future emergency patients to arrive. The initial goal g_1 is 'up-to-date patient particulars' and the associated FBR is 'Separate patient's contact and health insurance data'. This implies the document portion d_1 and d_2 of the *EHR* document (Figure 3.5) can be updated in parallel by administrative employees.

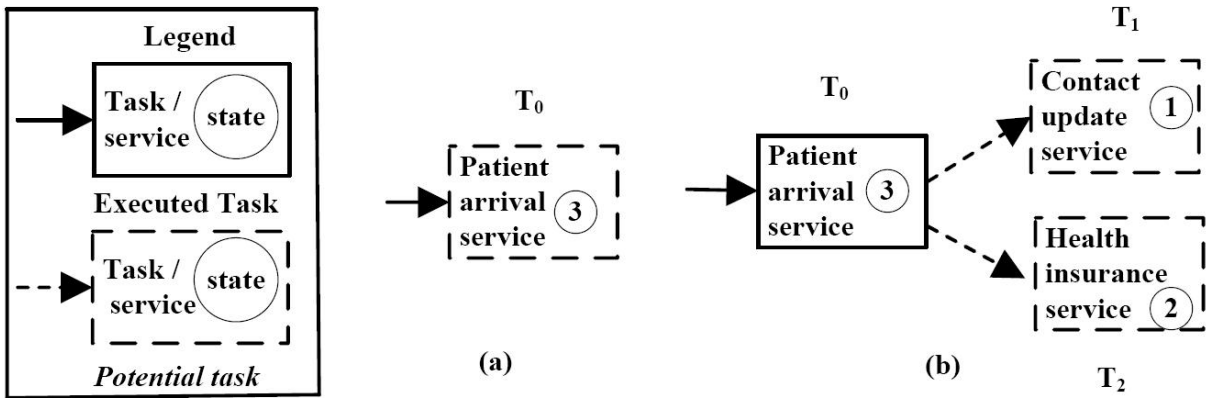


Figure 3.15: A dynamic task/service enactment of *EHR* generation process for the goal "Up-to-date patient particulars".

The determined human oriented service, i.e., 'Contact update service' (T_1) and an automated service, i.e., 'Health insurance service' (T_2) from the *KB* (i.e., *recipe tasks*) for the goal g_1 are to be executed. As the administration employee is available the task T_1 is immediately executable resulting into a task state value 1 for the task T_1. However, the 'Health insurance service' is currently unavailable (indicated by state 2) but can be executable as soon as it is up and running (Figure 3.15 (b)). The tasks T_1 and T_2 would not need to be executed if the patient would have been admitted previously in the hospital implying patient's particulars would have had up-to-date already. This can be determined, for instance, from a BPEL execution history of the generating *EHR* process in the *KB*.

8.2 Symptoms Identification

As the goal g_2 'Symptoms identification' can be achieved independently of the goal g_1, i.e., $g_2 \not> g_1$, a human oriented 'Pathology service', i.e., pathologists can already record the problem symptoms of the patient, (i.e., task T_3) into d_3, i.e., <Symptom>, (Figure 3.16 (c)) as suggested from the *KB* of the pathology department.

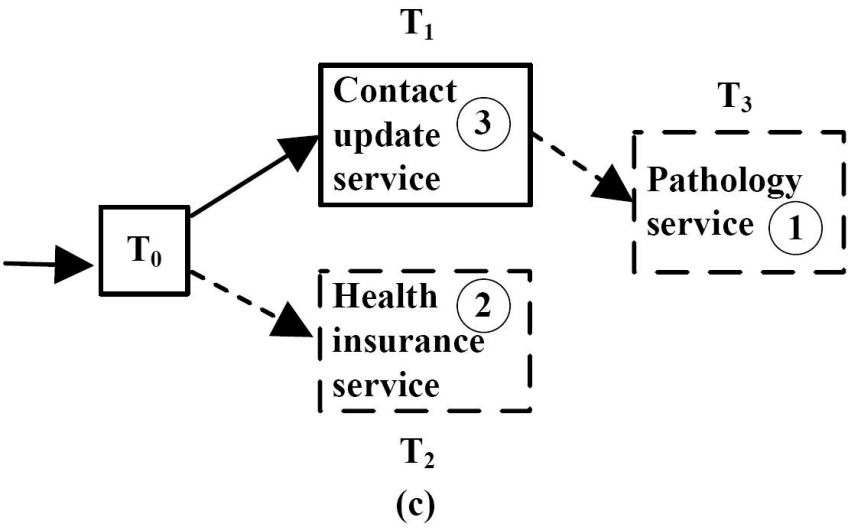


Figure 3.16: A dynamic task/service enactment of *EHR* generation process for the goal "Symptoms identification".

Note that the current state of the executed task T_1 is 3, that is, patient's contact <Contact> can be updated further if required.

8.3 Diagnosis Test Results

Now, for the goal g_3 'Diagnosis tests results', an associated symptom must be identified in d_3, i.e., <Symptom>, according to its FBR (of Figure 3.8). So, after d_3 is filled in with a symptom data by the task T_3, a diagnosis technician can record the test result by another human oriented service 'Diagnosis service', i.e., T_4 (Figure 3.17 (d)). Note that the states of task T_3 and T_2 are now 3 meaning symptom identification can occur again and 'Health insurance service' is also executable now.

As soon as the diagnosis test results are recorded in d_4, <DiagnosisTests>, the task T_4's state is changed to 3 that allows more diagnosis test results to be recorded if required (Figure 3.17 (e)). Note that, the state of the task T_2 is changed to 2 indicating that service is unavailable now. On the other hand, the task T_5, i.e., Medication service and the task T_6, i.e., Pathology service are enabled again. In particular, T_1 and T_6 are executable, i.e., states are 1 and 3 respectively.

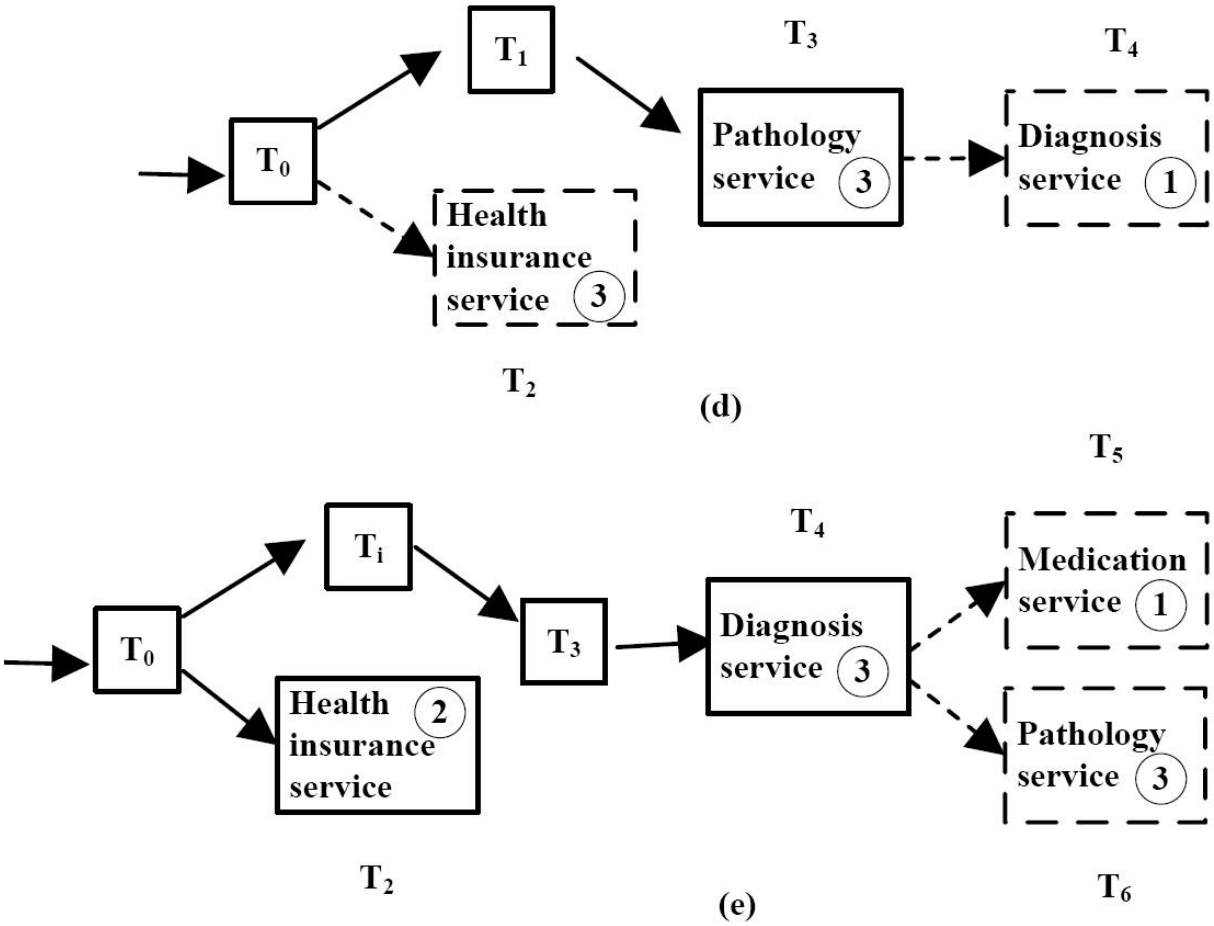


Figure 3.17: A dynamic task/service enactment of *EHR* generation process for the goal "Diagnosis test results".

8.4 Successful Treatment

Now for the goal g_4 'Successful treatment', operation theater has OBRs enabling suitable treatment that results in recording treatment data into `<Medication>, <Therapy>, <Surgery>` etc. The doctor may start the treatment, i.e., T_5, and thus may proceed with recording data.

As the rules of the pathology, diagnosis and the operation theater allow further tests and ongoing treatment (i.e., no deadlock and conflict), before treating further doctors may ask for new pathology diagnosis test records, i.e., T_6. It indicates that new test data may need to be recorded in d_4, i.e., `<Test>` while the doctors are medicating i.e., T_5. Then depending on the test records doctors may perform 'Surgery', i.e., T_7, and/or provide 'Therapy', i.e., T_8, services for a successful treatment (Figure 3.18 (f) and (g)) resulting updates into d_4 and d_5.

Now, when the treatment is successfully completed, the *DoCWF* execution will reach a final status and will generate a complete *EHR* document and thus the business goal is achieved (Figure 3.18 (g)).

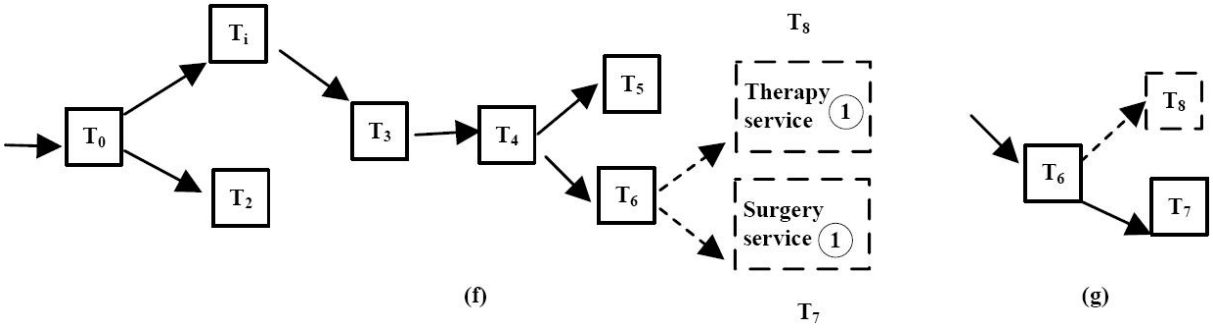


Figure 3.18: A dynamic task/service enactment of *EHR* generation process for the goal "Successful treatment".

9 Related Work

Workflow modeling and its execution have been studied for long in the research community and can be categorized into three main types:

1. Flexible Workflows for Agile Scenarios.
2. Typical Task-based Workflows.
3. Semantic Service Composition.

This section describes these three approaches and positions our work in that respect.

9.1 Flexible Workflows for Agile Scenarios

Many flexible workflows have been proposed [vdAW05, vdAJ00, MPvdA07, vdAP06, CCPP96, EK00, Wes01, vdAW05] to handle agile workflow business process scenarios and many communities [AB00, ADO00, Tag01] have developed techniques for flexible workflow management. These work vary diversely and can be divided into two main subcategories: declarative workflows and case-based workflow handling.

9.1.1 Declarative Workflows

In declarative workflow models, a set of business tasks are determined in design time and their execution sequence is governed by constraints-based rules associated to tasks (c.f. Chapter 2). As such, in these models constraints are applied on a set of pre-defined tasks for their flexible executions. With a plethora of works, this idea can differ into the three following ways considering the result of applying those rules to reach the following objectives.

1. **Avoiding changes:** To avoid changes by providing alternative paths [vdAW05, vdAJ00].

2. **Restricting execution paths:** To control execution paths by declarative constraints expressed in LTL formulas [MPvdA07, vdAP06].

3. **Allowing changes:** To allow changes in one execution and/or to change a process model while migrating all current executions [CCPP96, EK00, Wes01].

All these require significant disruption in a running business process and workflow executions [vdAW05, AB00, ADO00, Tag01, CCPP96, EK00, vdAJ00, Wes01] as frequent exceptions may occur leading to frequent backend changes. Like task-based workflow modeling, declarative modeling also tends to be exhaustive as it requires the complete enumeration of all business tasks and all the guarding conditions in the gateways, which in turn leads to an over specification of tasks, therefore the workflow [MPvdA07, vdAP06]. Most importantly, all these mechanisms still require an a priori definition of business tasks and their sequence flow which are infeasible in agile workflow business scenarios.

	Task-based Workflow	Declarative Workflow	Case Handling	Document-based Agile Workflow
Focus	Task	Task	Whole case	Document content, goal
Modeling artifact	Task	Task, constraints	Task	Goal, business rules
Task and sequence flow definition	A priori	Only task definition	A priori	Dynamic
Primary perspective	Control flow	Constraints over tasks	Case / instance	Constraints over goals, document content

Figure 3.19: Comparisons of document-based agile workflows with task-based, declarative and case handling workflows.

9.1.2 Case-based Workflow Handling

The authors in [vdAW05] proposed a case handling approach to support business processes where each case is handled in isolation (i.e., for each instance). Apparently the problem area of case handling and declarative workflows are close to our *DocWF* approach for agile business processes. However, case handling still considers that tasks and their sequence flow of a case are specified a priori.

Our approach is fundamentally different from declarative workflows and case-based handling as shown in Figure 3.19. We allow dynamic task definition, its enactment and business rules set constraints on goals and document content. Moreover, our work enables a loose coupling for suitable business task determination and its binding as opposed to their work.

9.2 Task-based Workflows

As we described the limitations of task-based workflows in Chapter 1, we only point out the differences of *DocWF*s with existing approaches of document-oriented workflows and goal-driven workflows to position our work.

9.2.1 Document-oriented Approach

Document oriented workflows proposed in [MRW96, Pod08, Kayb, WW98, Ros04, Sto] largely follow the task-based approach where authors in [MRW96] focus on a manufacturing process. In [Pod08] and [Kayb] the authors demonstrate the usage of XML technology to realize a document and workflow based collaborative system. The authors of [Kayb] defined the structure and content of documents based on a given business process whereas in our approach of *DocWF*, depending on the content, peers can determine which workflow tasks can be performed (i.e., "can be" rather than "should be"). X-folders described in [Ros04], trigger a task from a predefined orchestrated set of tasks depending on the state of the documents inside a folder. Such a state correspond to a complete document. In our approach, states are defined for the previously executed or executable services so as to determine an appropriate service for a given business task. As such peers in our approach are independent of any predefined task meaning they may define new tasks depending on the document content. Another difference with our approach is that the authors of [Ros04] trigger a pre-specified task which has a static binding to concrete services as opposed to ours. In line with [Pod08], we developed a secure XML document-based collaboration technique [RRS08], that allows exchanges of fine grained documents among anonymous actors. Upon receipt of such documents, our approach of a *DocWF* can be applied to reach a business goal.

9.2.2 Goal-driven Approach

The authors in [Rob96, BBB06b] describe goal oriented workflow modeling techniques to generate alternative workflows whenever necessary. Our proposed *DocWF* system differs from that approach in two aspects: (1) unlike the goal of [Rob96] which depends on stakeholders goal and the goal hierarchy of [BBB06b] which are associated with pre-planed execution paths, our model supports the derivation of subgoals including security goals from a business goal independently of actors and execution plans; (2) a goal achievement is recorded by data instantiation in the documents making a document a stateful representation of a workflow which is not considered at all in [Rob96].

9.3 Semantic Service Composition

With the recent research initiative for semantic business task description of BPMN [SPM] service composition can be of two main types described below.

9.3.1 Semantic-based Business Task Determination

In [MBW07], the authors have developed a semantic matching technique of business tasks with the operations performed over business domain objects. The resulting business tasks then can be composed using algorithms, for instance, as in [WMD07]. As mentioned before in Section 2, their semantic match making techniques can be used to determine suitable business tasks. However, the main differentiating factor of this technique with our approach is their composeable services are pre-selected in that a business task has its direct binding with respect to a low level service implementation or a task itself represents a service.

9.3.2 Services Composition for Dynamic Task Enactment

There has been extensive work (surveys can be found in [MM04, RS04]) about the semantic composition of services that can be categorized into two general approaches: determining Web services at process design time [ADK+05] or at runtime [JMM03]. Our dynamic task enactment approach generally falls into the runtime category but differs mainly by two aspects:

1. The states of available services are modeled formally by a task/service state model allowing a rigorous approach of determining a suitable service or substituting an existing one at runtime.
2. Integration of a new service implementation by *recipe tasks*, for instance, when available services are not found to be suitable for a goal.

However, we assume that alternative services are semantically equivalent to a substitutable service which may not always be true in a pervasive setting as shown in [FGIZ08, FGIZ]. We discuss this issue in the concluding Chapter 7.

10 Conclusion

In this chapter, we presented a document-based agile workflow (*DocWF*) system that is particularly suitable for agile workflow scenarios where required tasks and their sequence flow may need to be determined dynamically. A business actor can model a *DocWF* process utilizing her domain expertise without a priori knowledge of actual tasks/services which can then be determined by matching of semantic descriptions of *DocWF* models and BPMN models for instance. Enactment of these tasks is performed possibly by binding to semantic enabled services or defining new tasks or implementing new services. The employed business rules set constraints over goals and documents and are expressed as LTL formulas. Such rules treat a *DocWF* execution as goal achievements while a goal can be grounded to one or more task enactments. Utilizing the automaton of these formulas actors can detect modeling errors caused by inconsistent models. By implementing a modeling platform, for instance, actors might even be notified of the modeling errors before actual task determination and later executions. A problem with such a rule-based system is possible contradiction, in particular when rules are introduced by different actors. However, associating priorities with rules may resolve such contradiction.

The work of this chapter has been published in the proceedings of IEEE International Conference on Services Computing, SCC 2009, September 21-25, 2009, Bangalore, India [RRS09a].

Chapter 4

Interoperability of Document-based Agile Workflows

No great discovery was ever made without a bold guess.
- Isaac Newton -

1 Introduction

In Chapter 3 we presented the basic modeling and dynamic task enactment principles of a document-based agile workflow to support the determination of suitable business tasks and their later binding with services. These principles are particularly appropriate for executions of agile business scenarios that may lack prior knowledge regarding suitable business tasks and services. In particular, using a semantic matchmaking of business goals with, for instance, corresponding goals and/or pre/post conditions of annotated business tasks of a semantic BPMN model, *DocWF* actors can determine suitable business tasks to achieve their goals. Furthermore, binding tasks to concrete IT components, e.g., Web Services, can be done by means of semantic based service discovery mechanisms. Then the proposed task state models enable appropriate service determination in case of unreachable services for instance. The *DocWF* model utilizes business notions such as business goals and business rules and makes it possible to check models rigorously by detecting deadlocks and conflicts at design and runtime that may occur due to faulty models. However, *DocWF* models assume that business actors from varying business boundaries have not only common understanding of the content carried by diverse vocabularies in the documents that are exchanged amongst them but also have appropriate data structure modeling techniques that enable their information structures to evolve. One way to understand the diverse vocabularies in the documents is to annotate the documents with adequate semantic tagging and thus making documents interoperable. Considering this required interoperability of a *DocWF* system, the document meta model of Chapter 3 needs

to be implemented. In this chapter, we elaborate on this by assuring a stable interface amongst business actors and semantic annotations of "Enterprise XML" documents exchanged during an execution of a document-based agile workflow application.

The remainder of this chapter is organized as follows. In Section 2 we introduce a document oriented workflow scenario based on which interoperability requirements are elicited and basics of interoperable documents in a *DocWF* are discussed. To implement the document meta model, Section 3 describes a semantic enabled document modeling approach using which a semantic annotation technique for document nodes is developed in Section 4. In Section 5 we introduce comparison mechanisms for "Enterprise XML" documents as identified below as an important technique for interoperable documents which is further elaborated from Section 6 to Section 9. Section 10 positions our work with related literature and finally Section 11 concludes the chapter.

2 Problem Statement and Example Scenario

The increasing standardization of XML processing (e.g., XML Schema, DTD, XSL) makes it possible for peer organizations to cooperate and to integrate their information systems through XML document production and exchanges. Documents are typically structured and modeled through XML schemas in peer organizations. These models may evolve due to changes in business organizations. For instance, frequent mergers and acquisitions of companies lead them to repeated changes in the existing common data exchange format. Moreover, schemas may contain valuable and sensitive information about resources, strategies, services, organizational structures, marketing plans, or information system structures closely tied to business processes, and which organizations do not want to expose to competitors. When executed, a production business process scenario, derived from [Sida], generates such a hypothetical "Enterprise XML" document as described below.

2.1 A Production Business Process Scenario

Activities in a production are known as *work orders* which are reflected in a business document by production activities. Activities are triggered by document exchanges and performed by either internal business units or third party organizations. For example: 1) The sales unit receives a "purchase order" from a customer containing a "list of items" to be produced. 2) The sales unit evaluates the "purchase order". 3) After evaluation, the sales unit sends the "purchase order" to the production unit which then determines activities (e.g., production, packaging, labeling, quality inspection) to be performed in the production hall depending on the requested items. 4) For each activity a *work order* is then issued by the production department. 5) The production department plans the execution of *work orders*.

Conceptually a work order in a production environment contains the following information:

- *Work order* meta data: such as type (production order, maintenance order, quality-inspection order), id, priority level (urgent, normal, escalated), issue date.

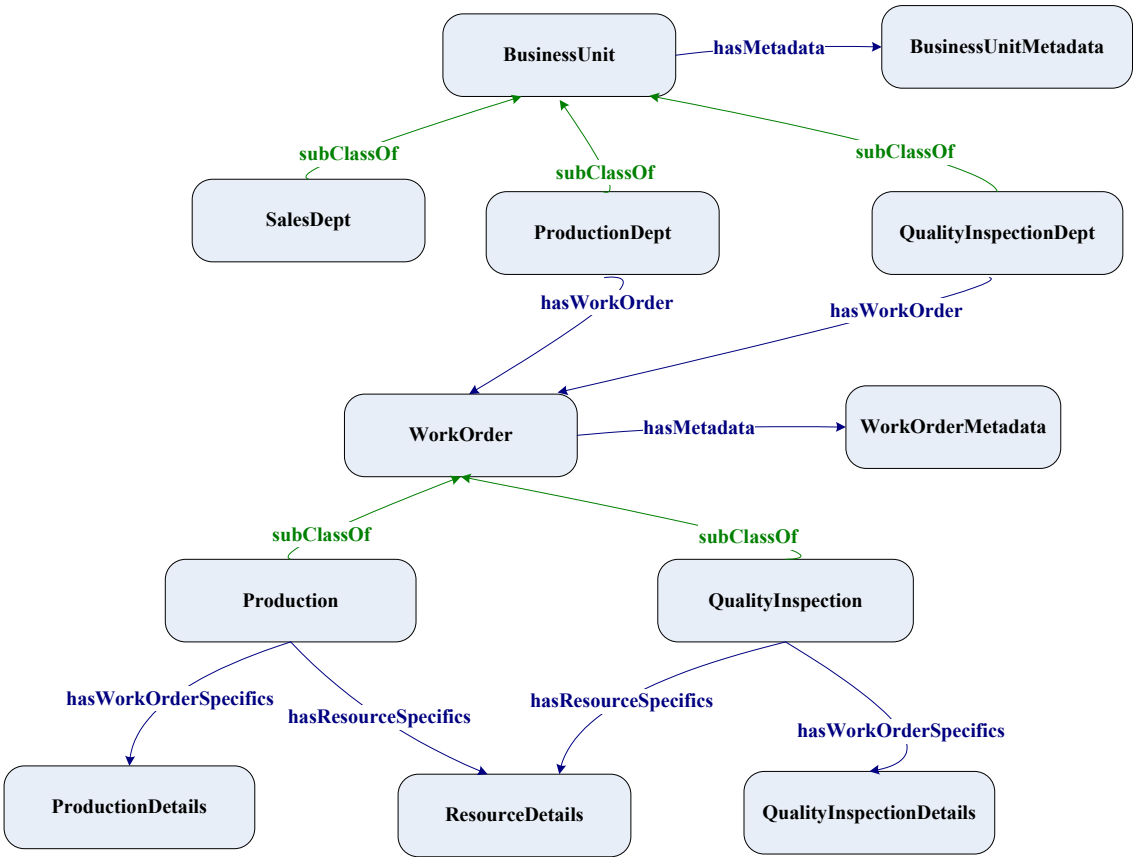


Figure 4.1: A semantic graph representing production business process *concept*s.

- *Work order* details: for instance, for the production order, specification of the products and quantities.
- *Work order* resource details: for instance, for the production order, allocation of employees, machines, assembly lines etc.

All *work orders* require the abovementioned information which are conceptually same (i.e., same type of data with different structures). In particular, meta data structure would be the same for all *work orders*. However, since each *work order* is performed by independent business units they have individual and specific details in the way they are encoded and handled. For example, the *work order* details for a "quality-inspection order", executed by a third party quality inspection company, could be: specifying the product quality and the metrics to measure them. Moreover, some activities can not be pre-specified until some other activities are finished. For example, for the "production order", the allocated employees, used machines and assembly lines can not be specified during the resource allocation planning time unless the order is evaluated.

Finally, product related specifications need to be sent to the quality inspection company which can then use the specification to define its *work order* details. Figure 4.1 shows a simplified ontology graph representing production business process *concept*s (e.g., production, quality-inspection etc.). Production and quality inspection units map related *concept*s to their individual information structure of XML documents (shown in Figures 4.2 and 4.3 respectively). For simplification the corresponding OWL XML

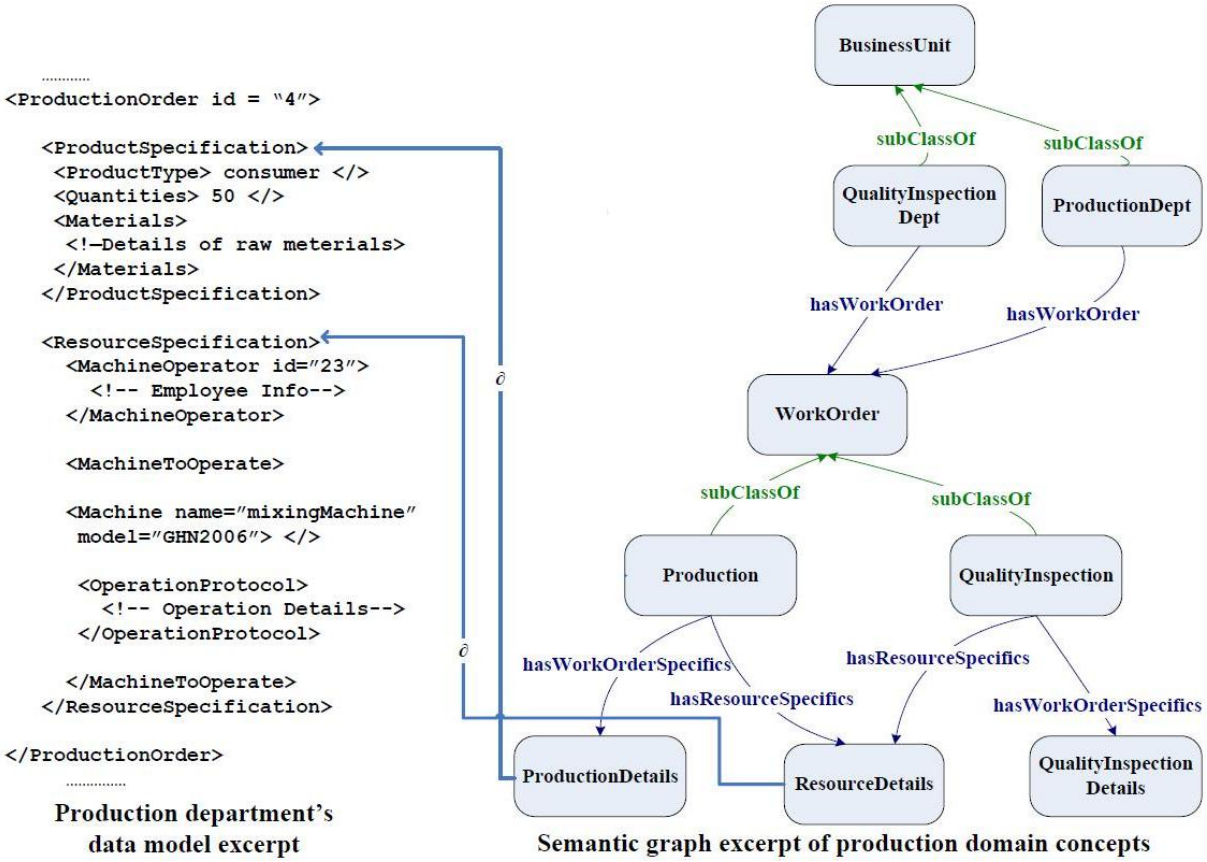


Figure 4.2: "ProductionDetails" work order and "ResourceDetails" *concept*s are mapped to the corresponding XML data model excerpts of production department using a mapping relation, ∂.

excerpt is not presented as an ontology graph can equivalently represent the ontology *concept*s in a domain [UDRS05].

2.2 Requirements

Document exchanges in such a document oriented workflows must support the following two features:

1. **Non-disruptive document exchanges:** The interface amongst organizations must be stable by limiting changes of the interface as much as possible.
2. **Distributed document handling:** Documents can be created, accessed and updated autonomously by actors and services hosted in different business boundaries. Therefore distributed modifications performed by actors need to reconciled methodically.

Regarding non-disruptive document exchanges, we claim that, although data models may differ from one organization to another or vary with time, the semantics of XML document content or data units like

subtrees or nodes might constitute a more stable and interoperable interface between organizations. Semantic Web languages like RDF [RDF] and OWL [OWL] make it possible to share an ontology describing the semantics of the document content (a conceptual data model), independently from any XML data structure. In our case, the ontology specifies data that are carried by documents and their relationships in a business domain. This allows interacting organizations to independently design and evolve their own information models whenever required without affecting their shared interfaces and yet map the ontology *concept*s to their desired document instances and portions thereof. We also claim that access control can be defined at the semantic level notably to achieve a simpler expression of policies with complex rules (detailed in Chapter 5). To enable distributed document handling we need to address various issues as described and motivated below:

1. **Semantic enabled Enterprise XML:** Exchanged documents should be annotated with adequate semantic information so that diverse vocabularies of "Enterprise XML" are understood beyond business boundaries.
2. **Fine-grained document access:** Access rights must be specified on the semantic level which then needs to be enforced on the document instance level that may span from several document instances to a simple document node.
3. **Document convergence:** All authorized business actors should access a consistent document.
4. **Enabling document comparison:** Actors must be able to distinguish between different versions of a document and their content (i.e., both structurally and semantically).

Regarding semantic enabled "Enterprise XML", in the production business process business units have their specific vocabularies for an agreed business *concept*. For instance, the *concept* "$ResourceDetails$" of Figure 4.2 is represented as `<MachineOperator>` etc. and `<QualityTester>` etc. by the production unit and quality assurance unit respectively. With respect to fine-grained document access, first, expressing access rights over a single *concept* might result into granting authorizations to multiple XML documents or portions thereof. Second, and more importantly, authorizations on *concept*s might be automatically inferred from the expression of the right to access a related *concept*. Third and finally, as shown in related work like Rei [KPS+04], ontologies can formally describe an access control model by representing policy *concept*s as first-class objects. We contend that this feature is particularly suitable to inter-organizational document exchange systems like the production business process, by making it possible to specify expressive policies which may also evolve in the course of time.

Regarding document convergence, as mentioned in Chapter 2, any data exchange oriented collaboration requires mechanisms to control concurrent or individual updates of a document by several peers so that eligible peers always access a consistent document throughout an execution of a *DocWF*. The solutions described in [VCFS00] exactly suit this need that showed how to keep a group of collaborative peers up-to date in real time with respect to an editable shared data object, for instance, a document or document portion. However, mechanisms to distinguish the fact that one document is different from the other specially when considering "Enterprise XML" and documents containing sensitive data are yet to be developed. In this context, the solution for interoperable documents in a *DocWF* also includes document comparison techniques. Existing solutions [CGM97, CRGMW96, NJ02] assume documents to be compared are in memory, thus requiring a huge amount of space, and that those documents do not contain sensitive data. Therefore existing solutions proved to be limited. In this chapter, we have developed document comparison techniques that are elaborated from Section 5.

2.3 Current XML-based Data Modeling Approaches and Limitations

Currently the abovementioned issues are partly addressed. In particular, by sharing XML schemas and DTDs as common data exchange interfaces and by using XSL style sheets for transforming one XML format to another [BLL04, DdVPS01, DdVPS02, FCG04, KMR05, MTKH03]. Substitution mechanisms of XML schemas may be used also for transformation. However, these techniques are limited in terms of non-disruptive document exchanges between peers as described below.

- **Frequent changes of peer data model:** Any change of a local data model may require changes in the shared schemas and thus disrupts document exchanges.
- **Inadvertent data disclosure:** Shared XML schemas or DTDs require peers to exchange XML documents in a specific format which might expose individual data models inadvertently as the shared format might represent the original data structure partially if not completely.
- **Knowledge of peer data models:** Performing XML vocabulary translation using XSL style sheets does not require a common schema, however, it requires the knowledge of interacting peers' XML data models for the translation. Similarly, substituting a source XML vocabulary with a target XML vocabulary still requires the knowledge of the interacting organization's XML data model.

3 Semantic Enabled Document Modeling

In order to support non-disruptive document exchanges amongst actors a stable interface is required. Although existing approaches may implement interoperability in a static scenario it turns out to be limited for *DocWF* applications as mentioned before.

This section describes an ontology-based data model that needs to be implemented for a stable interface in the production business process scenario. Such a model specifies business *concept*s and their relationships. Appropriate ontologies can be sought depending on business scenarios.

3.1 Ontology-based Document Model

An ontology-based document model describing domain semantics can be agreed upon amongst peers instead of sharing schemas. This serves two purposes:

1. **Stable data exchange interface:** A stable interface amongst actors overcomes the limitations mentioned before, in particular it allows the evolution of a local data model without affecting the interface and protects any inadvertent data disclosure.
2. **Flexible policy specification:** This allows a document provider to change its policy whenever required and thus enables flexible authorization policies (detailed in Chapter 5).

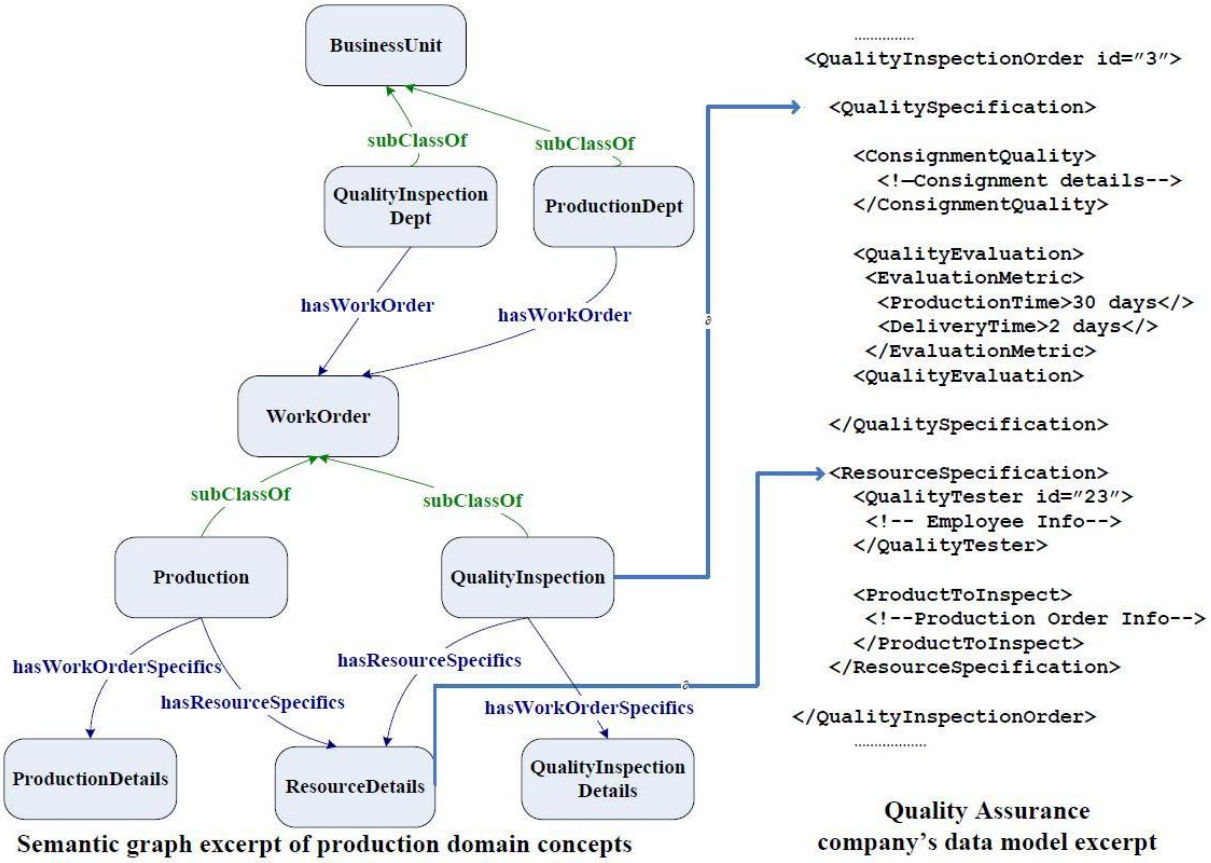


Figure 4.3: The "QualityInspection" work order and "ResourceDetails" *concept*s are mapped to the corresponding XML data model excerpts of a third party business unit, quality assurance, using a mapping relation, ∂.

A business domain semantics that typically consists of vocabularies of the domain of interest can be captured by ontology *concept*s where a *concept* is defined as follows.

Definition 1. A *concept* denoted as C_i is an abstraction of any physical or logical entity of a business domain that can be shared and communicated amongst actors of varying business boundaries.

A collection of *concept*s is commonly called an ontology [NM][1]. The semantic representation techniques of an ontology can be, for instance, RDF [RDF], OWL [OWL] and DAML[DAM]. Semantical representation being agnostic of the data representation and associated behavior, are focused on the meaning of the data itself as opposed to syntactical representations. In effect, a semantical representation can capture not only multiple syntactical representations but also model relationships between several semantical *concept*s. For instance, using OWL, an ontology *concept*, denoted by a *class* can be represented by multiple vocabularies while relationships between *concept*s are captured using *subclass* and *properties* of classes.

In a *DocWF*, a domain-specific ontology provides the common language to communicate the contents of an XML document. Figure 4.1 shows a semantic graph of *concept*s, shared by production

[1] An ontology *concept* has its properties and associated data types.

department and quality inspection company, defines an ontology (e.g., work order, production, quality-inspection etc.). It also illustrates how these *concept*s may be mapped to XML documents in a production business process scenario. For example the 'ResourceDetails' *concept* can be modeled in two different XML document structures as shown in Figures 4.2 and 4.3. Relationships amongst ontology *concept*s are further defined by extending the idea of a *subclass* and called as *concept containment*.

Definition 2. *Concept Containment:* Let $\mathcal{C}$ be the collection of all domain *concept*s and $C_i, C_j \in \mathcal{C}$. If there is a *subclass* hierarchy from C_i to C_j or a property relationship exists between C_i to C_j, denoted by $C_i \Rightarrow, ..., \Rightarrow C_j$ then C_i contains C_j and denoted by $C_i \preceq C_j$.

Example 1. Figure 4.1 shows a collection of *concept*s $\mathcal{C} = \{BusinessUnit, BusinessUnitMetadata, ...QualityInspectionDetails$ etc.$\}$ for the production business process. $WorkOrder$ contains $QualityInspection$ and $Production$, i.e., $WorkOrder \preceq QualityInspection$, $WorkOrder \preceq Production$. □

Since the work order related *concept*s (e.g., work order meta data, work order details, resource details etc.) for all interacting peers are the same, peers can agree on a collection of *concept*s. Each peer can then define its own XML data model independently which can then be associated (i.e., mapped) to a *concept* as described in the next section.

3.2 *Concept* Mapping to a Document

An "Enterprise XML" document denoted as d and identified by doc_{id} (e.g., URI, RDF) is a collection of parsed XML nodes and a document portion d_i is a subtree of d rooted at node i. We assume that documents do not have any mixed content meaning that any content of an XML node is inside a tag pair. A mapping, ∂, specifies relations from a *concept* to document portions d_i which is used by a peer to specify its XML content associated to *concept*s.

Definition 3. Mapping Relation, ∂: Let D be a set of XML documents having corresponding identifiers $\{doc_1, ..., doc_i\}$. $\partial(f, metric)$ is a mapping relation where f is a function from a *concept*, C_i, to a set of document portions, $\{d_i | d_i \in doc_i\}$. $metric$ is a measurement value that is used only for resolving a desired mapping in case several mappings exist from different *concept*s to the same document portion and thus specification of this $metric$ is out of scope of this thesis. The $metric$ ranges typically from 0 to 1, i.e., [0,1]. The mapping with the highest $metric$ value is applied eventually. The function f is defined as follows:

1. $O.C_i \longmapsto_f d_i$: a *concept* of a collection, $\mathcal{C}$, identified by $O.C_i$, where O is a path expression that can be formed by the *concept containment* relation, relating to a XML document portion, d_i.
2. $O.P_N(SC_N, RC_N) \longmapsto_f d_i$: a property P_N identified by $O.P_N(SC_N, RC_N)$, where O is a path expression having the superclass name SC_N and range class names RC_N, relating to a set of XML nodes of d_i.

Multiple document portions, each originating from a distinct peer, may be associated with the same *concept* and thus they are semantically related. A peer only maps *concept*s to its appropriate data model

(i.e., document portions) and as such some *concept*s may remain unmapped for that peer. Similarly, there might be some document portions (i.e., XML nodes) that do not have any corresponding association to *concept*s. Consequently, mapped document portions of a document are disjoint: $\{d_i |\ d_i \subseteq doc_i$. Let d_i, d_j be two mapped document portions of a document d, then $(d_i \cap d_j) = null$. Such a mapping is illustrated by the following example.

Example 2. In Figure 4.2, the *concept*s, $ProductionDetails$ and $ResourceDetails$, identified by the paths over the semantic graph $BusinessUnit.ProductionDept.hasWorkOrder.Workorder.Production.hasWorkOrderSpecifics.ProductionDetails$ and $...Production.hasResourceSpecifics.ResourceDetails$ are mapped to the document portions rooted at `<ProductSpecification>` and `<ResourceSpecification>` of the production department's XML data model. In Figure 4.3, the quality assurance company's data model, the *concept*s $ResourceDetails$ and $QualityInspection$ identified by the path expressions $...QualityInspection.hasResourceSpecifics.ResourceDetails$ and $...QualityInspection$ are mapped to the document portions rooted at `<ResourceSpecification>` and `<QualitySpecification>` respectively. □

Now that basic document modeling is described, we move on to semantic enabled document parsing.

4 Semantic Enabled Document Parsing

To annotate an "Enterprise XML" document according to a mapping (possibly down to a node level) the followings should be considered with respect to parsing:

1. **Annotation while parsing:** Annotation should be performed while parsing a document.
2. **Annotating semantic, document structure and security information:** Annotation should enable not only attaching semantic information but also document structure and security information.

Regarding annotation while parsing, a one pass encoding algorithm that allows annotating structural information with associated *concept* to "Enterprise XML" nodes is developed in this chapter. For annotating security meta data, this algorithm is extended and described in the next chapter. For any new document this encoding should be performed before sending the encoded document to other actors of a *DocWF* application.

An XML document typically consists of multiple levels of tree nodes where nodes having the same depth are positioned in the same level. We consider XML documents of ordered nodes where nodes can be elements, attributes and text. Attributes of an element can be represented as the first set of children before its sub-elements and text. Node order is important, for instance, in a BPEL document `<invoke>` elements in a different order implies different orchestrations. Annotation techniques utilize this natural tree structure of an XML and can be described by the two following steps which are detailed in the following sections.

1. **Traversing XML documents:** XML nodes of an XML document are traversed in breadth-first order (level by level) as opposed to typical XML node order (i.e., pre-order) or reverse node order

(i.e., post-order). Pre-order and post-order are suitable for view intensive accesses while breadth-first order is more suitable for *DocWF* applications as discussed in Chapter 2.

2. **Structural encoding of XML documents**: The actual annotation of XML documents involves attaching XML structural and conceptual information as meta data (Figure 4.4) for a traversed XML node.

4.1 Breadth-First Order Labeling (BOL) for XML Documents

Once document modeling is done, a document provider parses an XML document as follows: XML nodes are traversed level by level from root to leaves. While traversing a node, its children nodes are stored in a FIFO (First In First Out) queue for their later traversal. For each traversed node, an integer pair called BOL as defined below, capturing various structural information of that node, for instance, parent-child, siblings, left/right child, with a minimized memory footprint (detailed later) is computed.

Definition 4. Breadth First Order Labels (BOL): A *BOL* is a pair of integers associated with an XML node as it is parsed in breadth first order. The first integer in the pair is the traversing order associated with the node whose left siblings and ancestors have already been parsed and thus have associated BOLs. The second integer is the depth of the node in the document tree which is increased by one as new depth level is reached. The BOL starts with (1,0) as illustrated in Figure 4.4 (the example given is a binary tree, but BOLs can be defined on any type of tree)

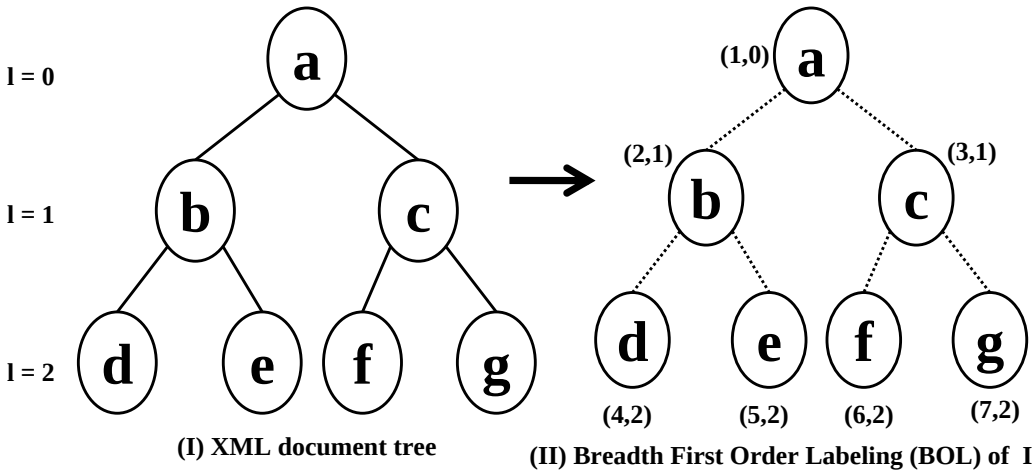


Figure 4.4: (I) An XML document tree. (II) BOL labeling. Solid and dotted lines respectively depict explicit (I) and implicit (II) hierarchy representations and storage.

4.1.1 BOL Generation

The basic idea is to associate BOLs to the XML nodes of each level from left to right starting from the level 0. To assign the BOL numbers in breadth first fashion that is for the nodes in a level of the XML

document tree we maintain a FIFO queue of unassigned child nodes of that level. After assigning BOL numbers to all sibling nodes of a level in a document order the next unassigned node in FIFO order will be processed in a similar fashion. Note that the hierarchical relationships (i.e. parent-child, siblings, left/right child) of the parsed XML nodes are not explicit in the BOL model and thus a considerable amount of memory is saved as mentioned in Chapter 2.

Let a be an XML node, $a \in d_i$, where d_i is the set of nodes (i.e., document portion) processed so far. Let a be the parent of two nodes $b, c \in d_i$. We denote a's BOL as B_a. Let f_{order} and f_{level} be two functions operating on a BOL respectively returning the BOL order (first integer of the BOL pair) and BOL depth (second integer). Let b be the last child of a that is parsed and c be the next node to be parsed. c will be assigned a BOL with the value $f_{order}(B_c) = f_{order}(B_b) + 1$. $f_{level}(B_a)$ uniquely identifies the depth level of the node a in a document d.

4.1.2 Implementing the BOL Model

To implement such a model by leveraging existing APIs, we employ the hybrid approach as mentioned in Chapter 2. This is explained as follows. The sole use of an event based API such as SAX is to keep the current parsed node in memory and the hierarchy information of the current node is therefore not maintained. In this context, the computation of a BOL is implemented as follows. For a traversed node, e.g., b, of a given level using SAX, a FIFO queue of unassigned child nodes, e.g., d and e can be extracted using DOM api. After labeling all the sibling nodes, i.e., c, of that node, i.e., b in the same level, the next node in a FIFO order in the queue of the unassigned nodes will be processed in a similar fashion. Figure 4.5 illustrates the level by level BOL computation of the XML document tree of Figure 4.4 and illustrated in the following example.

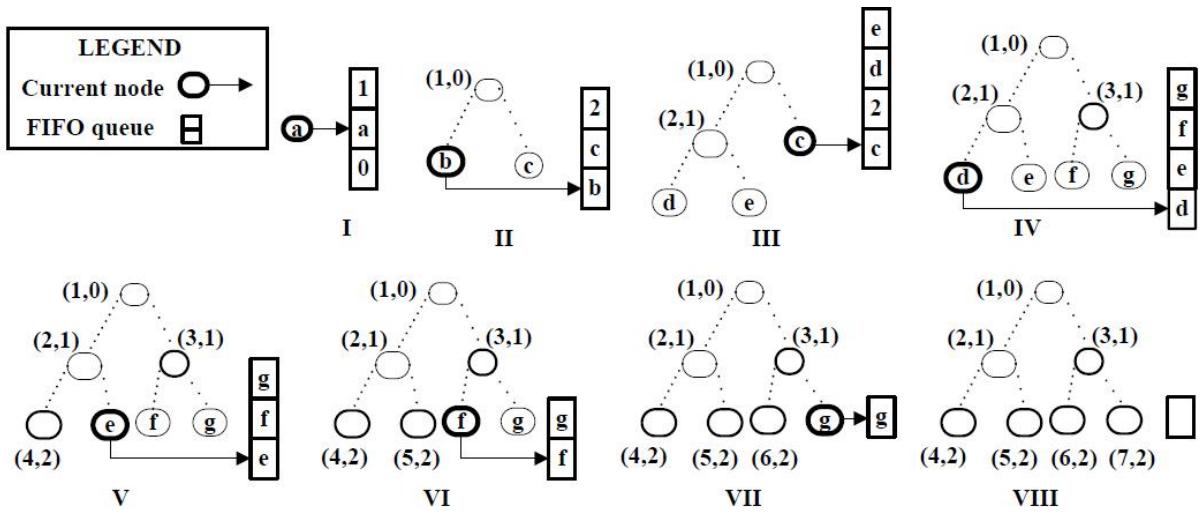


Figure 4.5: (I)-(VII) Level by level BOL computation of the XML tree of Figure 4.4.

Example 3. The BOL parsing of seven XML nodes of Figure 4.4 are described by leveraging a FIFO queue which is shown eight times in Figure 4.5 (I) to (VIII). The queue represents current parsed node and future nodes to be parsed. The nodes of a level are delimited by a level counter which is an increasing integer value starting from 0. (I) shows the initial queue containing the root node a which is delimited

by 0 and 1. The node a is associated with the BOL pair (1,0) and the children nodes, i.e., b and c, are pushed into the queue (II). Now, the current node b is fetched from the queue and associated with the BOL pair (2,1). The depth level is increased from 0 to 1 as the node b is in the next level of the node a. Now, the children nodes of b, i.e., d, e, are pushed into the queue (III). As the node c is fetched for assigning the BOL pair (3,1) its children nodes, f, g, are pushed into the queue (IV). The next node to be assigned the BOL pair (4,2) is d. The depth level, i.e., level counter, is increased from 1 to 2 as the node d is in the next level of the node c as determined by the increasing level counter (V). The nodes e, f, and g are assigned the BOL pairs (5,2),(6,2), and (7,2) respectively in a similar fashion until the node g is reached when the queue becomes empty and the BOL computation terminates.□

4.1.3 BOL-based XML Structural Properties

BOLs exhibit the following structural properties:

1. **Unique node identity:** $f_{order}(B_a)$ uniquely identifies node a in document d and the subtree d_a rooted at a.

2. **Unique identification of a parsed document portion:** Let $B^a_{Highest}$ be the largest BOL order of a parsed node in document portion d_a; then $B^a_{Highest} > f_{order}(B_z) > f_{order}(B_a)$, where $z \in d_a$.

3. **Non-decreasing node identity:** The numeric value of the BOL order gets increased according to the parsing order; $f_{order}(B_c) > f_{order}(B_b) > f_{order}(B_a)$.

The first property can be used to identify and extract a specific document portion from a document. Combined with the depth level of a node, that property ensures that any unexpected move, copy or replace of a node in the document is detected. The second property imposes an upper bound on the BOL of any queried node parsed in a document. In effect, it detects if a node is added or deleted and which one it is. Intuitively, the second property also allows the identification of an unexpected move of a document portion. The third property permits detecting any unintended swapping among the children in a received document portion (subtree).

Example 4. In Figure 4.4 (II), the BOL of node c is a pair of c's traversing order (3) and c's depth level (1). The first integer is a unique number associated to c and the nodes a, b are associated to smaller integers, i.e., $1, 2$ than that of c respectively. The second integer represents c's depth level in the XML tree. The highest BOL of node c, i.e., $B^c_{Highest}$, in the document portion rooted at c is the BOL of the node g, i.e., $(7, 2)$. □

4.2 Structural Annotation with *Concept*s for Enterprise XML

In the following, the basic set of annotation elements (see Figure 4.6) are introduced that can be used as annotations for the mapped document portions/data units (i.e., subtrees or nodes).

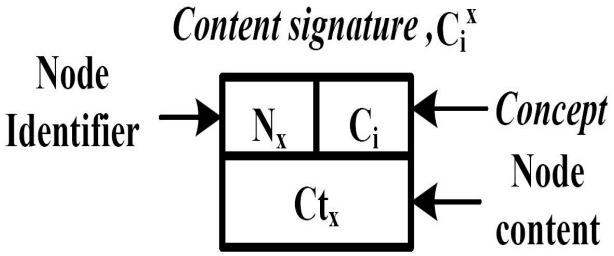


Figure 4.6: Basic Annotations of an "Enterprise XML" node, x and its content.

Node Identifier: Let x be a node in d_i. The node identifier of x denoted by N_x is a tuple formed by three elements $(doc_{id}, B_x, B^x_{Highest})$, where doc_{id} is the document identifier of d_i, B_x is the BOL of x, $B^x_{Highest}$ is the highest BOL in the document portion rooted at x.

A node identifier is unique for all documents in a *DocWF* application. As mentioned before, the depth included in B_x uniquely determines the node's level. The B_x and $B^x_{Highest}$ together determine the parsed document portion. Finally, doc_{id} resolves appropriate XML nodes considering the same *concept* that may be mapped to multiple XML nodes vocabularies of multiple documents as shown before.

Content Signature: Let C_i and x be a *concept* and an XML node respectively. The *content signature*, denoted as C^x_i, is a pair of node identifier and the associated *concept*, denoted by (N_x, C_i), where N_x is the node identifier of x and C_i is a *concept* mapped to the subtree x.

Algorithm: Annotating "Enterprise XML"

1. **Input:** $\mathcal{C}$, a collection of *concept*s $\mathcal{C} = \{C_i\}$; a set of documents identified by $\{doc_{id}\}$.
2. **Output:** Annotated document.
3. Let $B \in \mathbb{N}$ be an integer for BOL order, $l \in \mathbb{N}$ be the depth level of a node, Q be a FIFO queue.
4. FOR all documents $\{doc_{id}\}$ do
   (a) **Initialize:** set $B = 1; l = 0; Q[0] = l; Q[1] =$ root node of doc_{id}.
   (b) WHILE Q is not empty do
      i. Let x be the current node.
      ii. IF x is a depth level delimiter then
      set $l = l + 1$; Add l into Q.
      iii. ELSE
         A. **BOL Generation:**
         POP x from Q; Associate (B, l) to x, $B_x = (B, l)$.
         Add all the children nodes of x into Q.
         B. **Determining Mapping:** $\partial(f, metric)$.
         C. **Structural Encoding:**
         Determine node identifier of x as $N_x = (doc_{id}, B_x, B^x_{Highest})$.
         Determine *content signature* of x as $C^x_i = (N_x, C_i)$.
         Generate annotated content as (C^x_i, Ct_x), where C^x_i is the *content signature* of the node x, Ct_x is the content of x.

Figure 4.7: Basic annotation process of "Enterprise XML".

The *content signature* incorporates the basic semantic information such as conceptual and structural information attached to an XML node. This can be used for the following purposes:

1. **Common interpretation:** To interpret the same semantics of an XML node by all peers without knowing the vocabularies used for the XML node and its content.
2. **Selective document routing and delivery:** To determine and extract appropriate XML nodes for their later selective routing and delivery.
3. **Integrity verification:** To check document data structures (i.e., appropriate XML structures) and their semantics when security information (e.g., hash values) is added as security meta data.

Selective routing and delivery and integrity verification utilizing *content signature*s are detailed in Chapters 5 and 6 respectively. It is important to note that the annotation method is extensible in that it allows additional meta data to be added as can be required for selective dissemination by the communication infrastructure and later security verification for instance. In Chapter 6, we will see an extension of this annotation method that adds security meta data in order to consider security issues such as content integrity and fine grained access.

The algorithm in Figure 4.7 shows the basic annotation process that is represented graphically in Figure 4.8. In Figure 4.7, the lines 3 to 4.b.iii.A capture the BOL technique for XML documents of Section 4.1. The line 4.b.iii.B captures the *concept* mapping to documents of Section 3. The structural encoding of line 4.b.iii.C represents the structural annotation technique of Section 4.2.

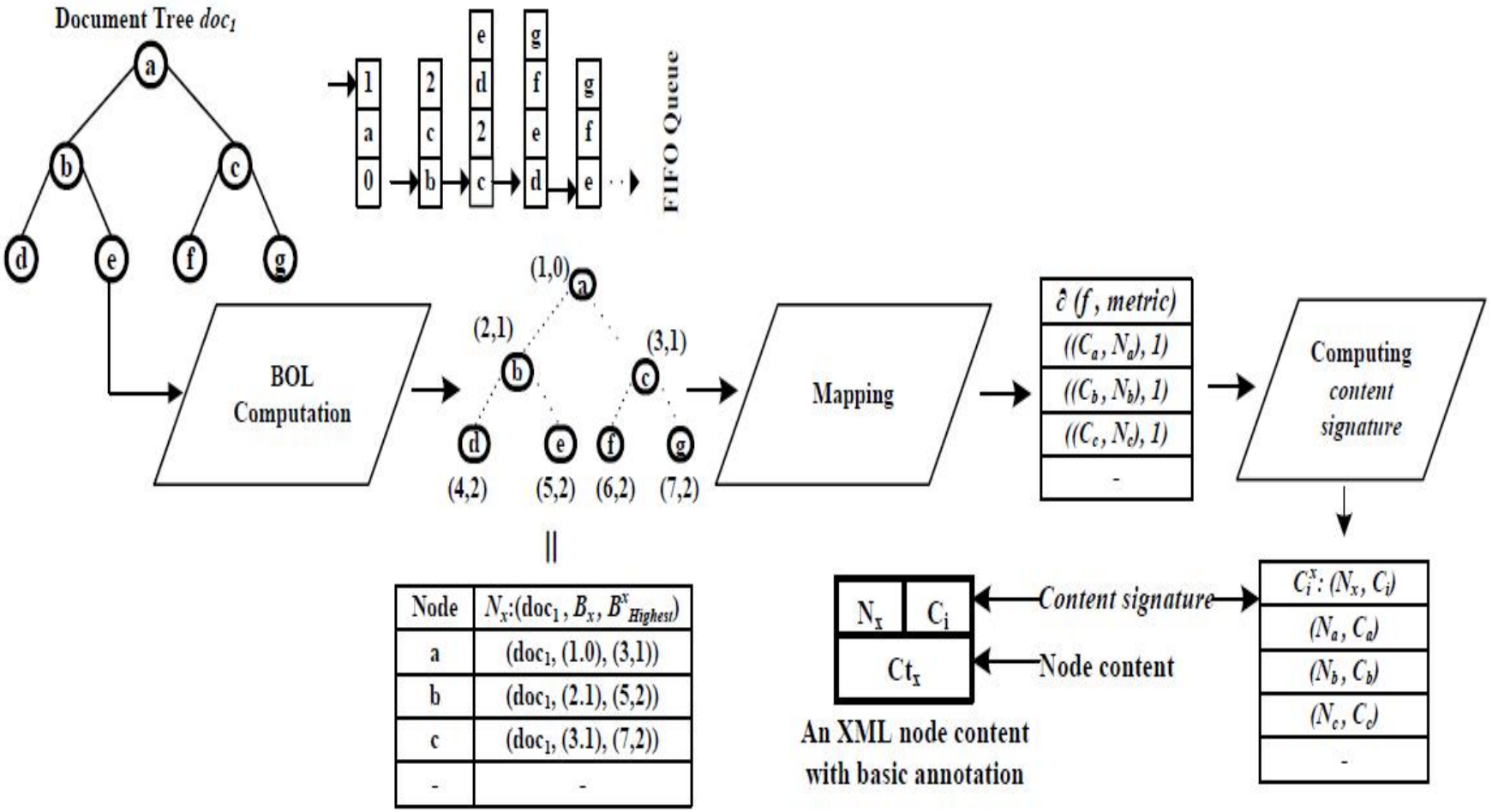


Figure 4.8: Processing steps of BOL computation and basic structural annotation with *concept*s of "Enterprise XML".

5 Comparing Enterprise XML Documents

In this section, we describe a comparison technique to determine the differences between multiple "Enterprise XML" documents as motivated in Section 2. A difference is described by an edit script which is defined as a sequence of edit operations such as node *update, insertion, deletion,* and *subtree move* performed sequentially over an initial version of a document. There might be several such edit scripts from which a *minimum cost edit script* can be determined based on an edit operation cost model. This edit script enables document convergence techniques, for instance, of [CGM97, CRGMW96, NJ02] to be applied so that actors from varying boundaries can access a consistent document. In the following, we first introduce the required functionalities of a comparison technique before discussing the limitations of existing approaches and gradually developing our solution.

5.1 Requirements for Comparing Enterprise XML in a *DocWF*

Updating tree-structured data such as XML documents results in different versions of the original XML document. Detecting changes or the similarity of tree structured data has many applications such as aggregation of similar XML databases, difference queries, versioning, merging of documents. In the context of a *DocWF* application, such a comparison technique must have the following functionalities:

1. **Sensitive XML Document Comparison:** The plain and possibly sensitive (both structure and content wise) "Enterprise XML" documents must be comparable.
2. **Nondisclosure Comparison:** The comparison technique must not allow the comparer, possibly a third party hosting a communication infrastructure node, to interpret original XML structure and content.
3. **Partial Comparison:** The comparison technique must enable comparison of fragmented document portions, e.g., one level of a source tree nodes to be compared with another level of a target tree as opposed to comparing two complete trees.
4. **Cross Generation of Documents:** The comparison technique should enable the generation of different versions of a document from a stored one whenever required.

Recalling the security requirements of a *DocWF* application in Chapter 1, "Enterprise XML" documents may carry sensitive information such as organizational strategy, marketing and financial data etc. In effect, a peer may encrypt a document portion with a key before sending it to others as to keep that document portion confidential to an adversary. As such, we focus on sensitive XML document comparison while the developed technique is equally applicable to plain text XML documents. With regard to nondisclosure comparison, a peer may send several versions of an encrypted document which need to be compared by a third party in order to deliver the most up-to-date version to another peer for instance. The third party must not be able to read the plain text content (e.g., element name, attribute name value pairs, text content) or original structure (e.g., number of nodes, size of the document). For such a non-disclosure comparison by a third party enabling partial comparison is very important as it may host fragmented document portions originated from varying boundaries. Moreover, differences between

multiple versions of a document portion are mostly localized. For example, a WSDL document containing different service operations that may evolve only by adding or removing service operations resulting into some insertions or deletions of a few XML elements at the same level within newer versions of the WSDL document. Regarding the cross generation of documents, storing all versions of a document may not be practical with respect to storage issues, for instance, at communication infrastructure nodes, yet it is required to generate other versions from a stored document version whenever needed.

5.2 Limitations of Current Comparison Approaches

Comparison techniques for tree structured data and the generation of a *minimum cost edit script* have been extensively studied as in [CGM97, CRGMW96, NJ02]. However, these studies fall short with respect to the functionalities as specified above. The main reasons are the following:

- **Comparison by a trusted party:** Typically, the comparison is assumed to be performed within a closed business boundary as opposed to cross business boundaries of *DocWF*s.
- **High memory and comparison time:** Matching algorithms require complete source trees and intermediate normalized trees to be in memory and require traversing those trees multiple times. Thus significant computation time and space are required.

Regarding the comparison by a trusted party, consider "Enterprise XML" documents containing sensitive information and thus XML elements, its attributes and content may be encrypted and thus not readable and not comparable by any match maker or comparer who does not have the necessary decryption key. However, in a *DocWF* application this non-disclosure comparison should be enabled as mentioned before. This is further illustrated in the following example.

Example 5. Imagine a company A uses various Web services from another company B. A and B have a shared key and as such, B always provides encrypted WSDLs which only A can read. Now A appoints a third party C to compare different encrypted WSDLs provided by B over a period to determine the changes. C can compare such WSDLs and can detect the changes only if it possesses the key. □

Regarding matching algorithms to determine *minimum cost edit script*s, a well known approach is to have initial matches of node pairs computed over the full trees of two versions [CRGMW96] for which comparison functions and approximations are applied. However, this requires the parsing of both the source and the normalized trees multiple times (i.e., in pre-order, in-order, post-order) and thus the matching algorithms require more time and memory. Our proposed solution overcomes these limitations as described in the following.

5.3 Characteristics of the Proposed Comparison Solution

Our comparison technique has three key characteristics including functionalities of Section 5.1:

1. **Comparing Large XML Documents:** We focus on large hierarchical structured information such as "Enterprise XML" documents. The literature tackles the tree comparison problem using at least a complete representation of the trees in memory [CRGMW96], then transforming those into possibly multiple normalized forms such as, binary branches [YKT05], which makes those techniques infeasible for "Enterprise XML".
2. **Edit Operations on Encrypted Nodes:** An edit operation is performed with respect to an encrypted tree node of a level as opposed to its plain text value. To enable partial comparison, edit operations are defined with respect to sibling relationships of XML nodes in a level rather than their parent-child hierarchy. For example, any node (leaf or internal) can be deleted irrespective of its children nodes. We allow four edit operations, i.e., *update, insert, delete, move* where each operation is performed on a single sibling node. The sibling node can be a leaf or internal as opposed to existing work which only allows, for instance, to delete a leaf node in a specific order, e.g., post-order.
3. **Minimum Cost Edit Scripts for Cross Generation of Documents:** The *minimum cost edit script* is generated in a single pass algorithm on the document versions. The algorithm starts with an empty match and empty edit script and as it proceeds it finds appropriate matches of nodes and a *minimum cost edit script*.

5.4 Comparison Solution Overview

Consider two trees, T_x and T_y of Figure 4.9, each having two levels of nodes where T_x is the initial and T_y is the edited tree, any comparer wants to find the differences between them as an edit script defined below.

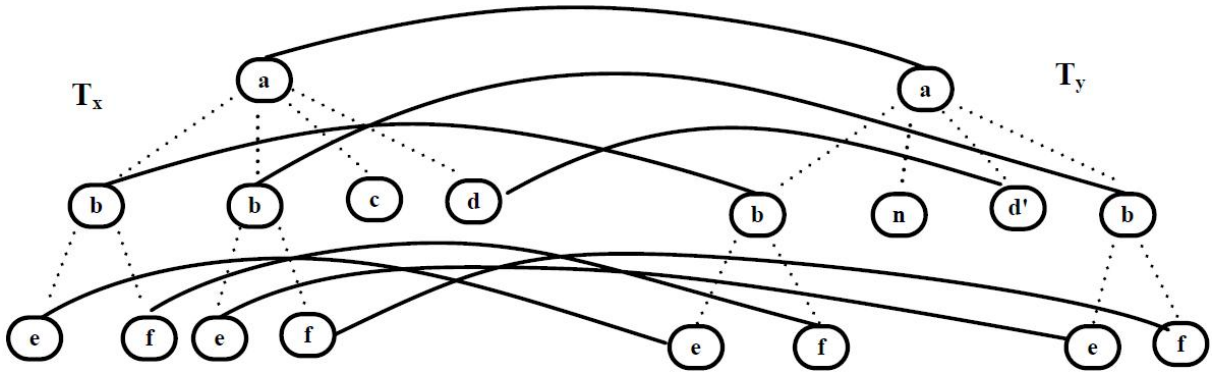


Figure 4.9: Comparing two trees, T_x (initial document tree) and T_y (edited). Solid lines represent appropriate matches.

Definition 5. Let $T_0 \xrightarrow{s_1} T_1$ where T_1 is the result of applying the edit operation s_1 to T_0. An edit script, denoted by S, is a sequence of edit operations, which when applied to T_0 transforms it to T_i. Given a sequence $S = s_1 \dots s_i$ of edit operations, we say $T_0 \xrightarrow{S} T_i$ if there exists $T_1, T_2, \dots T_{i-1}$ such that $T_0 \xrightarrow{s_1} T_1 \xrightarrow{s_2} T_2 \dots T_{i-1} \xrightarrow{s_i} T_i$.

Accordingly, the task is to find an appropriate transformation as described by a *minimum cost edit script* from T_x to T_y. In other words, to determine the correspondence (i.e., an appropriate matching)

among nodes of these two trees as shown by the solid lines in Figure 4.9. Appropriate matching also identifies an insertion or deletion of nodes as shown in the figure for the nodes n and c respectively (without solid lines). In our context, the nodes to be matched are encrypted values of XML nodes, i.e., elements, text, attributes, as opposed to plaintext values. These nodes may be updated in different versions. Whenever we refer to an XML node name we refer to its encrypted value unless stated otherwise.

To implement the partial comparison, specially designed edit operations are specified that allow editions of XML nodes independently of their hierarchy relationships (Section 7). In order to allow a third party to identify an encrypted node uniquely without knowing any original structural information about the node like, location and depth, the BOL-based annotation technique is extended with encryption, into what we call non-disclosure structural annotations (Section 6). Hence, our strategy is to utilize this unique identity and thus use a node's sibling relationship as opposed to using parent-child hierarchy in the matching algorithm. In particular, for two trees T_x and T_y, a node x in a level l of T_x is compared first with the nodes of the same level l of T_y. In this context, we define the following.

Definition 6. Level-wise Isomorphic: One level of a tree T_x is said to be isomorphic [CRGMW96] to a level of another tree T_y if nodes of both levels are identical except possibly for node names.

Surely, this strategy finds appropriate matches (if exists) at the same level without parsing the complete tree. If any node of T_x in level l remains unmatched, the other levels of T_y can be considered to find appropriate matches.

Definition 7. Document-wise Isomorphic: One document tree T_x is said to be isomorphic [CRGMW96] to another tree T_y if all levels of T_x are level-wise isomorphic to T_y.

We need to transform T_x to a tree T'_x which is document wise isomorphic to T_y using an edit script. To find a *minimum cost edit script*, appropriate matches between the nodes of T'_x and T_y incurring the cheapest cost must be determined. So our goal is to generate a *minimum cost edit script* by finding these appropriate matches (Section 8). Regarding memory consumption and computing time, our proposed *minimum cost edit script* generation algorithm has the following interesting features:

1. **No initial match:** The edit script generation algorithm does not require any initial matches. Instead, it computes the appropriate matches during algorithm execution.

2. **One pass algorithm:** The algorithm traverses the source trees in only one pass in a breadth first order to generate the *minimum cost edit script* and computes a complete set of matching node pairs.

3. **Minimum memory and computing time:** As a result of items one and two, the algorithm requires less memory and computing time compared to existing solutions. An edit script naturally allows a comparer to store only an initial version of an "Enterprise XML" and *minimum cost edit script*s whereas the latter can be applied over the initial version to generate other versions. Thus a comparer does not need to store other versions.

6 Non-Disclosure Structural Annotation of Enterprise XML

The exchanged documents need to be annotated in such a way that meta data attached with the fine grained mapped document portions do not expose sensitive information to an adversary. In order to do so, we need to extend the BOL technique of Section 4 as a BOL is by definition plain text and thus may reveal important structure specific meta data, such as number of nodes, depth of a node and thus the size of the document and even hierarchical relationships among the nodes to an adversary. Encryption over such BOL numbers as defined below prevents this information from undesirably leaking to an adversary.

Definition 8. Encrypted BOL (EBOL): Let B_a be the BOL of an XML node a. Let f_e be an order preserving encryption function [AKSX04]. The EBOL of a, denoted as E_a is a pair of integers defined as : $(f_e(f_{order}(B_a)), f_e(f_{level}(B_a)))$. While $f_e(f_{order}(B_a))$ is performed for each node a, $f_e(f_{level}(B_a))$ is performed if a is the first node in a level implying one level down while parsing.

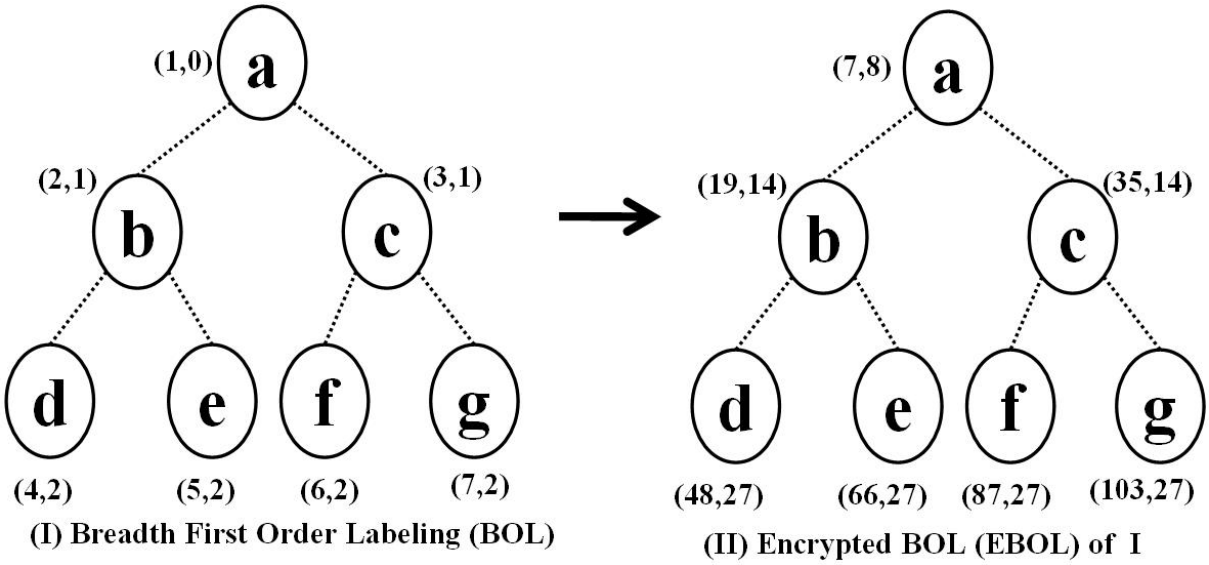


Figure 4.10: Encrypted Breadth First Order Labeling (EBOL) for "Enterprise XML".

The EBOL can be generated in the same fashion like BOL and preserves exactly the same properties of BOL (see Figure 4.10) as follows.

1. **Unique node identity:** $f_{order}(E_a)$ uniquely identifies node a in document d and the subtree d_a rooted at a.
2. **Unique identification of a parsed document portion:** Let $E^a_{Highest}$ be the largest EBOL order of a parsed node in document portion d_a; then $E^a_{Highest} > f_{order}(E_z) > f_{order}(E_a)$, where $z \in d_a$.
3. **Non-decreasing node identity:** The numeric value of the EBOL order increases according to the parsing order; $f_{order}(E_c) > f_{order}(E_b) > f_{order}(E_a)$.

The difference between EBOL and BOL is that the EBOL order value hides the actual node number and actual depth level as opposed to BOL integers and thus prevents structural information disclosure

which may be inferable with respect to a node. Most importantly, similar to a BOL, EBOL implicitly preserves a node's hierarchy information that allows to design reach edit operations based on solely node's sibling relationship as described in the next section.

7 Difference Detection of Enterprise XML

In this section, we formalize the detection of changes between two "Enterprise XML" documents using edit operations on them. Edit operations are performed on single XML nodes, in particular on single nodes. We refer to a node x with its EBOL-based identifier (in particular $f_{order}(E_x)$), its encrypted node value with the val_x and to an EBOL-based parsed XML document with the tree, T_i. T_i refers to the XML tree on which an edit operation is performed and T_{i+1} refers to the resulting tree. Formally, the edit operations are as follows:

7.1 Edit Operations Model

- **Update:** The *update* of the value of a node x in T_i is denoted as *Upd(x, newval)*. This operation leaves T_{i+1} unchanged with respect to T_i except for the value of x that is then equal to $newval$. This is depicted in T_0 and T_1 of Figure 4.11 for *Upd(59, d')*.

- **Insert:** The *insertion* of a new node x with a value val_x into T_i is denoted as *Ins(x, val_x, k)*. A node x with value val_x is inserted after the node k as its immediate right sibling node in T_i. In particular, if $r_1, ..., r_m$ are the right sibling nodes of k in that order in T_i, then $x, r_1, ..., r_m$ are the right sibling nodes of k in T_{i+1}. In case of an insertion of a node as a first sibling node, k is considered to be the dummy node. Insertion can be performed after any leaf or internal node. (T_1 and T_2 of Figure 4.11 for *Ins(53, n, 19)*.

- **Delete:** The *deletion* of a node x from T_i is denoted as *Del(x)*. The resulting T_{i+1} is the same as T_i without the node x. In particular, if $l_1, ...l_n, x, r_1, ..., r_m$ is the sibling sequence in a level of T_i, then $l_1, ...l_n, r_1, ..., r_m$ is the sibling sequence in T_{i+1}. To delete a leaf sibling node is straightforward as depicted in T_3 and T_4 of Figure 4.11 for *Del(48)*. When deleting an internal sibling node, its children are stored in the FIFO queue as shown in Figure 4.12 so that these nodes can be fetched from the queue and thus be considered for the next level comparison.

- **Move:** The *move* of a node x after a node y is denoted as *Mov(x, y)* in T_i. T_{i+1} is the same as T_i, except x becomes the immediate right sibling of y. The children of the moved node are kept in the queue in similar fashion as the delete operation (T_2 and T_3 of Figure 4.11 for *Move(59, 53)*).

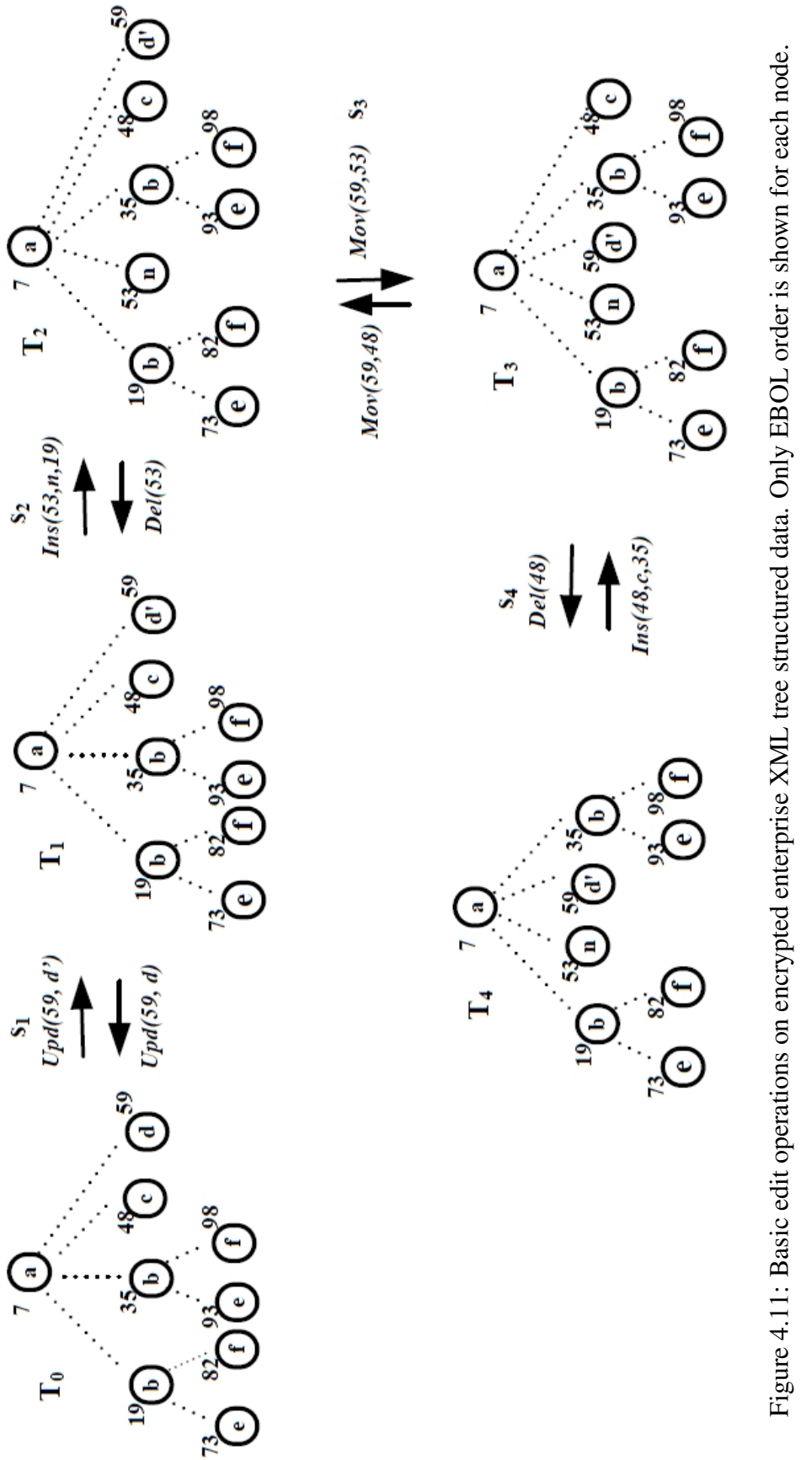


Figure 4.11: Basic edit operations on encrypted enterprise XML tree structured data. Only EBOL order is shown for each node.

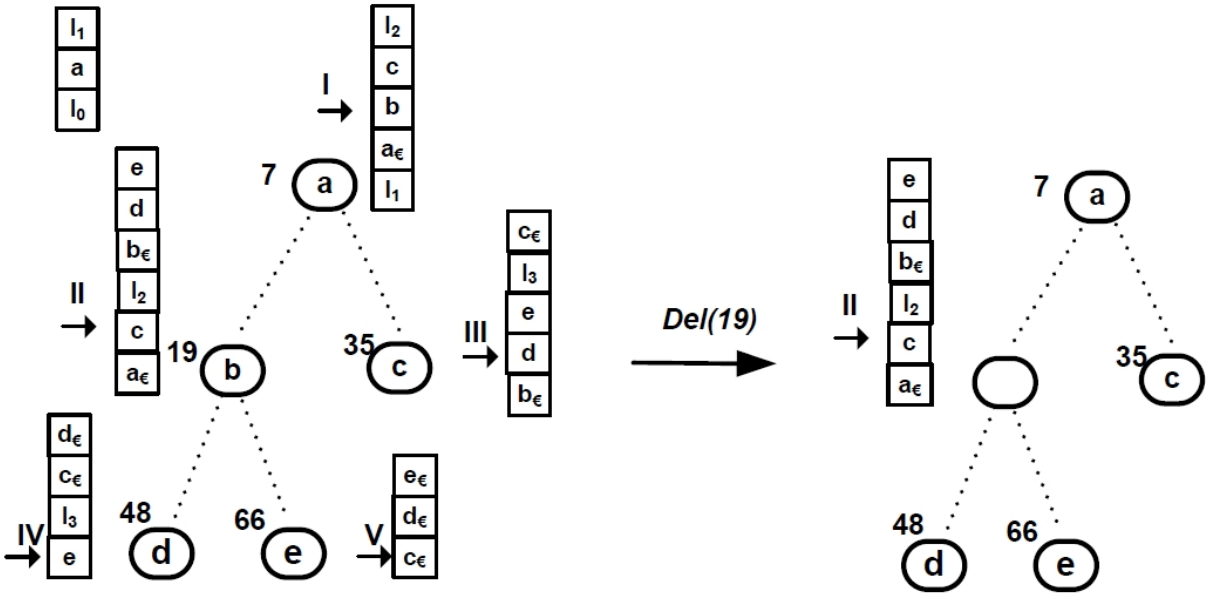


Figure 4.12: Deleting an internal node, b, (*Del(18)*) using EBOL-based XML encoding. The FIFO queue stores the nodes of the current level and keeps track of the current parsed node shown as $\xrightarrow{[I..V]}$. The nodes including the dummy nodes in one level are delimited by two l_i entries.

Regarding reach edit operations, for example, in Figure 4.12, an internal node b can be deleted without deleting its children. In terms of SAX API, when an event of *startElement* of the node b is sent, b's child nodes, i.e., d, e, including the dummy child node b_ϵ are queued into the FIFO $\xrightarrow{II}$. The memories for the parent-child relationship of the node b, its children, and sibling relationship of its children are not required as the sibling nodes, i.e., children, are stored in a sibling order naturally in the queue and thus sibling relationship is preserved implicitly. It is also possible to move an internal node as its children nodes are put into the queue in a similar fashion.

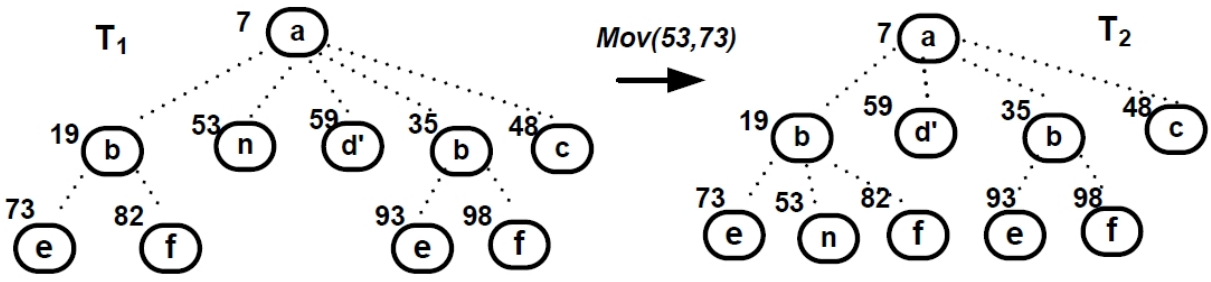


Figure 4.13: Inter level moving of the node n in 1st level of T_1 to 2nd level of T_2.

If the target level of the moved node is the same as its previous level then it is an intra level move (as in Figure 4.11). However, for any inter level move, for instance in Figure 4.13, the node n of T_1 is moved after the node e to the lower level, the plain strategy mentioned in Section 5.4 may not find the appropriate match in the same level. In particular, for the first level comparison it will be identified as n is deleted whereas it is moved to another level. Intuitively, when a node is moved upwards to a higher level it would be matched in for *insert* case as it is a new node for that level. In this regard, we maintain

an array of currently matched node pairs and the current edit script using which such an inter level move is detected (details in Section 8).

There may be several edit scripts that transform T_0 into the same resulting tree T_4. So, to find an appropriate match, an edit cost model is required as motivated in the example below.

Example 6. For example, the following edit script, $S = \{Upd(59, d'), Ins(53, n, 19), Mov(59, 53), Del(48)\}$ transforms T_0 to T_4 of Figure 4.11. Another edit script, $S' = \{Del(59), Ins(59, d', 48), Ins(53, n, 19), Del(93), Del(98), Del(35), Ins(35, b, 48), Ins(93, e, b_\epsilon), Ins(98, f, 93), Del(48)\}$, when applied in Figure 4.11 also transforms T_0 to T_4. Note that, for the insertion of $Ins(93, e, b_\epsilon)$ the dummy node b_ϵ is considered as mentioned before. Clearly, the edit script, c, performs more work than that of S and thus S' is an undesirable edit script to transform T_0 to T_4. In effect, to determine a *minimum cost edit script* a cost model is required. □

7.2 Edit Cost Model

The basic formalizations of edit script and cost model have been extensively studied in the literature [CRGMW96, LNS07, Ukk91, YKT05] and as such we utilize these formalisms from there on.

The cost of an edit operation depends on the type of operation and the nodes involved in the operation. Let $C_d(x)$, $C_i(x)$, $C_u(x)$, and $C_m(x)$ denote respectively the cost of deleting, inserting, updating and moving operations respectively. In general, these costs may depend on the value of x, as well as its position in the tree. In particular, the encrypted value represented by node x and its position in the sibling order in a level. Then a *minimum cost edit script* can be defined as follows:

Definition 9. Minimum Cost Edit Script (MCES): Given an edit cost model and a set of edit scripts, $\mathcal{S} = \{S_i\}$, where S_i is a sequence of edit operations (i.e., Update, insert, delete and move) and $i \in \mathbb{N}$. An edit script $S_j \in \mathcal{S}$ is a *minimum cost edit script* if it incurs the cheapest cost according to a cost model.

A simple edit cost model similar to [CRGMW96] is used where deleting, inserting, and moving a node x are considered to be a unit cost operation, i.e., $C_d(x) = C_i(x) = C_m(x) = 1$. For the cost $C_u(val_x)$ of updating an encrypted value associated to a node x, a function $diff$ is defined as follows: $diff(val_x, val_y)$ that returns 0 if encrypted values represented by val_x and val_y are the same otherwise a nonzero value is returned indicating that there has been an update.

The following sections describe the algorithm to determine a *minimum cost edit script*. We refer to a level of XML nodes of T_x as $l(T_x)$, to a node x as a node in a level and to a two dimensional array M as consisting of matching node pairs (x_i, y_j), where $x_i \in T_x$ and $y_j \in T_y$.

8 Computing a Minimum Cost Edit Script (MCES)

Now, we present the complete algorithm to generate a *minimum cost edit script*, S, that transforms a level of an initial version, $l(T_x)$ into an isomorphic to a level of the target version $l(T_y)$ by computing the

necessary matching node pairs. Intuitively, the algorithm can be applied repeatedly for other levels and as such T_x can be transformed into a document-wise isomorphic tree of T_y in one pass of the algorithm. The algorithm, shown in Figure 4.14, takes one level of tree nodes from T_x and T_y and uses four other functions, namely, *exist(), ArrangeSibling(), UpdateMatch(),* and *FindSibling()* shown in Figure 4.15. Additionally, it makes use of a two dimensional array, M, to compute the matching pairs in sibling order. When applied on a tree T_y (or M), the function $exist()$ returns the encrypted value val_y (or TRUE) if E_x, (i.e., $f_{order}(E_x)$), matches any node y in T_y (or x_i matches any node in M as a peer node) i.e., $\exists E_y = E_x$ or $\exists x_k = x_i$ in M, where val_y is the encrypted node value of y. Recall from Section 6, E_x is the EBOL associated to the node x. First we provide an overview of the algorithm, then we discuss how appropriate matchings of pairs of nodes are determined.

8.1 MCES Algorithm Overview

Given a tree T_x (the initial tree), a tree T_y (the edited tree), the algorithm generates a *minimum cost edit script* that transforms T_x to T_y. It does not take any initial match as mentioned in Section 5.4 and as such it determines the matches during an execution of the algorithm. It starts with an empty edit script S and adds edit operations to S as it follows through. Consequently, the algorithm requires only one-pass traversing as opposed to multiple passes on the trees.

8.1.1 Parallel Execution of Edit Operations

Each edit operation can be performed independently of others and thus any two edit operations may update the pairs of matches in M and the edit script S in parallel. While finding an insert, move and delete can be performed independently of each other, the update case is required to be performed first so that a first set of matching pairs is determined which is then used in other cases. We assume a multi threading control mechanism exists that disallows updating M and S by an edit operation while another is updating them and thus is not depicted in the algorithm. An edit operation is added to the edit script S and applies the edition to T_x only if it has access to M and S.

8.1.2 A Single Pass Breadth-First Traversal

We assume two root nodes of the initial and edited trees match without any loss of generality. The algorithm combines all the edit operations in one breadth-first traversal of T_x and T_y. Each edit operation takes one level of nodes at once consisting of all sibling nodes from one tree and compares these with a level of sibling nodes of the other tree. Once a matching pair of nodes is found, denoted by (x_i, y_j), the pair is added to M. The corresponding edit operation is added to S and applied to $T_i = T_x$ transforming it to T_{i+1}. Here, x and y are the matched node pair and i and j represent their respective position in the sibling order which may be changed by later edit operations (details follow). When the algorithm terminates, each node in T_x has an appropriate node match in T_y as shown by the solid lines in Figure 4.9.

Algorithm: Minimum Cost Edit Script (MCES)

1. Input: $l(T_x), l(T_y)$; Output: M and S.
2. $M = \epsilon$; $S = \epsilon$
3. Load the nodes of $l(T_x)$ and $l(T_y)$. /*load one level of T_x and T_y*/
4. **Update:** for each node $x \in l(T_x)$
   (a) $val_y = exist(x, -, U, -)$
   (b) if $val_y! = NULL$
   $UpdateMatch((x, y), Update)$.
   $v = diff(val_x, val_y)$. /*appropriate matching*/
      i. $if(v! = 0)$
         A. Append $Upd(x, val_y)$ to S.
         B. Apply $Upd(x, val_y)$ to T_x.
5. **Insert:** for each $y_j \in l(T_y)$; if $exist(-, y_j, -, M) = FALSE$ /*$y_j \notin M$; y_j as a peer node*/
   (a) $k = FindSibling(y_j)$.
   (b) $UpdateMatch((k, y_j), Insert)$.
   (c) Append $Ins(y_j, val_{y_j}, k)$ to S.
   (d) Apply $Ins(y_j, val_{y_j}, k)$ to T_x.
6. **Move:** Take the sequences of missarranged siblings: L_x, L_y;
   (a) $X = ArrangeSibling(L_x, L_y)$ /*Missarranged nodes of T_x*/
   (b) for each $x_i \in X$
      i. $k_n = FindSibling(x_i)$.
      ii. if $n > i$ then $UpdateMatch((x_{n+1}, y_j), Delete)$.
      /*if moved to right*/
      if $n < i$ then $UpdateMatch((x_{n+1}, y_j), Insert)$.
      /*if moved to right*/
      iii. Append $Mov(x_i, k)$ to S.
      iv. Apply $Mov(x_i, k)$ to T_x.
7. **Delete:** for each $x_i \in T_x$; if $exist(x_i, -, -, M) = true$ /*if $x_i \notin M$; x_i as a peer node*/
   (a) $UpdateMatch((x_i, \_), Delete)$.
   (b) Append $Del(x_i)$ to S.
   (c) Apply $Del(x_i)$ to T_x.

Figure 4.14: Computing a Minimum Cost Edit Script (MCES).

8.1.3 Auxiliary Functions

The function *exist(x_i, y_j,U,M)* checks the existence of a given node, x_i or y_j, of one tree in another tree (lines 4(a),5 and 7 of Figure 4.14). It is called from update, insert, and delete operations where parameters are filled in depending on the calling operation. For update x_i, U, (line 4(a)) for insert y_j, M (line 5) and for delete x_i, M (line 7) are filled in. For update, the nodes in $l(T_x)$ and for others the node pairs in M

Auxiliary functions of the MCES algorithm

1. Function *exist(x_i, y_j,U,M)*
   (a) if (U) then for each node $y_j \in l(T_y)$; /*update case*/
   do if $E_y = E_x$ return val_{y_j}; else return *NULL*; endfor
   (b) if (M) then for each node pair $\in M$
   if $y_j \notin M$; return true; /*insert case*/
   if $x_i \notin M$ return true; /*delete case*/
2. Function *UpdateMatch((x_i, y_j), editcase)*
   q, t, u, v are integers
   (a) if (editcase=Update)
   then $M[q] = (x_i, y_j)$, such that $\forall t, 0 < t < q$; $M[t] = (x_u, y_v)$ and $i > u$. /*adding pair nodes in M*/
   (b) if (editcase=Insert) for each pair $M[q] = (x_u, y_v)$, such that $u > i$, do $M[q+1] = (x_u, y_v)$. endfor
   $M[i] = (x_i, y_j)$ /*updating sibling position*/
   (c) if (editcase=Delete) for each right sibling node, $x_{u>i}$ of x_i, such that $(x_u, y_v) \in M$ do /*updating sibling position*/
   replace u with $u-1$; i.e., $(x_{u-1}, y_v) = (x_u, y_v)$. endfor
3. Function *ArrangeSibling(L_x, L_y)*
   Compute $L_{xy} = LSS(L_x, L_y)$. return $\forall x \notin L_{xy}$; /*missarranged peer node*/
4. Function *FindSibling(y_k)*
   for each $(x_i, y_j) \in M$
   if (y_k is the right sibling of y_j) return x_i. /*left peer node*/

Figure 4.15: Auxiliary functions *exist, UpdateMatch, ArrangeSibling, FindSibling* invoked by the MCES algorithm.

are compared. Each edit operation generates the matching node pairs and preserves the current sibling order of T_x in the array M by invoking the function *UpdateMatch()*, (lines 4(b), 5(b), 6(b), and 7(a)). Recall from Section 7 that an edit operates with respect to node sibling order. Depending on the invoking edit operation, *UpdateMatch()* takes care of the sibling order (see Figure 4.15). For the insert operation, the function inserts the new pair into the right position by moving the existing pairs (line 2(b)) and for the delete operation, updates the sibling position of the pair in M according to the sibling position of T_x (lines 2(a), 2(c)). The move operation invokes the *UpdateMatch()* with delete or insert parameters when a node is moved right or left respectively (line 6(b)). The insert and move operations invoke the *FindSibling()* function to determine the node after which node it should insert or move (lines 5(a), 6(b)). The *ArrangeSibling()* function computes the shortest sequence of move operations (described later) and returns the nodes of T_x to be moved when invoked by a move operation (line 3 of Figure 4.15). An edit operation is appended to S and applied to T_x after a successful matching.

8.2 Appropriate Matching of XML Nodes

The straightforward way to find appropriate matches of node pairs is to start with an initial set of matches [YKT05] determined by predefined similarity evaluation functions. Depending on the data set and domain of the matching this evaluation may vary. For example, if the comparing versions of the XML

documents have keys or unique identifiers then evaluation would first match the keys of the versions. If the data set is keyless then some domain dependent matching rules or criteria are used as evaluation characteristics. Intuitively, a criterion could state that a match is better than another if the former incurs cheaper edit costs than the latter.

We consider versions of XML trees to be keyless. However, EBOL-based encoding associates each XML node with an encrypted integer value pair that acts as a unique identifier to that node as described in Section 6.

As mentioned earlier, we would like to find appropriate matching pairs of nodes during an execution of the *minimum cost edit script* algorithm as opposed to existing matching algorithms that find an initial match in a first full tree traversal and then update that by subsequent traversals. The rationale is as follows:

1. **No Initial Match:** Initial match finding requires parsing (i.e., into memory) of "Enterprise XML" documents and their normalized trees before an actual algorithm execution which is undesirable in our context.
2. **Enabling Partial Comparison:** We want to enable the partial comparison of trees as motivated in Section 5.1, which requires the appropriate matching of sibling nodes without knowing their descendants, for instance.
3. **Matching Encrypted Node Values:** Matching should be performed over encrypted node values as opposed to plain text node values and thus it is not straightforward.

For (1), we utilize the EBOL based encoded nodes of a level as first class values for a comparison without requiring much memory nor a CPU intensive intermediate normalized forms of the trees. For (2), we define matching criteria for a node that do not require comparing descendant nodes of the other node except its direct children that are stored in the FIFO queue. For (3), matching criteria are applied over the encrypted values as the XML node values are not plaintext. Therefore, we formalize criteria that enable matching of nodes as a side effect during an execution of the algorithm. The first matching criterion can be stated informally as follows: nodes having dissimilar EBOL values should not match with each other.

Criterion 1: Sibling nodes $x \in T_x$ and $y \in T_y$ can match only if $E_x = E_y$, i.e., $f_{order}(E_x) = f_{order}(E_y)$. Recall that E_x denotes the EBOL value of the node x.

Given the first criteria is fulfilled two nodes can match after invoking the function $diff(val_x, val_y)$. Recall that $diff()$ returns 0 if val_x and val_y are the same otherwise it returns a nonzero value, where val_x and val_y denote the encrypted values of the nodes x and y respectively. If the returned value is nonzero, an updated node is detected and is used in the algorithm as an update edit operation. As we rely on symmetric and deterministic encryption, this check is then only matching the corresponding cipher text node values.

The second matching criterion is about the similarity of sibling nodes that have direct children. Stated informally, two nodes can match if their maximum number of direct children (as stored in their corresponding queues) also potentially match. Two functions $same(x, y)$ and $max(|x|, |y|)$ are defined

where x and y are the nodes to be compared and $|x|$ and $|y|$ are their number of children. The former returns the number of child nodes having the same EBOL and the latter returns an integer representing the maximum number of child nodes of the two nodes.

Criteria 2: Sibling nodes $x \in T_x$ and $y \in T_y$ can match only if

$$\frac{same(x,y)}{max(|x|,|y|)} > t; \quad where \quad 0 \leq t \leq 1.$$

The value of t is a threshold that depends on the business domain for which documents are exchanged and is chosen by the comparer. For instance, if the comparing XML trees are two purchase order documents having multiple `<item>`, `<price>`, `<quantity>` elements then it is quite likely that two documents may have more identical elements in a level and as such, the comparer can choose a higher value for $t \geq \frac{1}{2}$. If two WSDL documents are compared to check for operations change (addition or remove) then probably the value of t is lower, i.e., $t \leq \frac{1}{2}$ as the number of operations are less.

Finally, we assume that the possibility of finding identical nodes of a source and a target tree with respect to a level is higher than finding nonidentical nodes on the same level. In particular, the number of nodes in one level of a tree that are identical to nodes of a level of another tree is not smaller. As such, one node has bigger chances to match with another node if their sibling nodes also potentially match. This assumption reflects the goal of partial comparison where two versions of a document differ mostly at the same level (i.e., partial comparison) as mentioned before.

Assumption: For a domain dependent threshold value t, nodes $x \in l(T_x)$ and $y \in l(T_y)$, the number of nodes that satisfy $(E_x = E_y)$ is $\geq t$, for t as defined in criterion 2.

Now, we describe an execution of the algorithm of Figure 4.14 with respect to edit operations. While the first criterion is used for the first three cases, the second criterion is used to determine the optimal number of move operations.

8.3 Execution of the MCES Algorithm

We illustrate an execution of the *minimum cost edit script* algorithm with the example of Figure 4.9 and show the result in Figure 4.16. It transforms $T_x = T_0$ to T_4 by finding appropriate matches shown by the solid lines. Intuitively, T_4 is document wise isomorphic to T_y.

Update: In the update case, for each node x of T_0 the function $exist()$ is invoked to find whether x_i exists in T_y (Figure 4.14, line 4 (a)). If successful, the function returns val_y, then the function $diff(val_x, val_y)$ is called (Figure 4.14, line 4 (b)). For a nonzero value, we add the edit operation $Upd(x, val_y)$ to S, and we add a match (x_i, y_j) to M. Consequently, we apply the update operation to T_0. Ultimately, T_0 is transformed to T_1 by assigning $val_x = val_y$ such that $E_x = E_y$ for each node x in T_0 which has a corresponding EBOL identifier in T_y (i.e., $exist(x_i, -, U, -)$ in T_y). Note that, even if there is no updated node in T_y meaning a 0 is returned from $diff$, M may have pairs where each peer node in a pair has a corresponding matched node from the other tree. Figure 4.16(II) shows $Upd(59, d')$ when applied to T_0, the transformed tree is T_1. Figure 4.16 (II) also shows all the matching node pairs in M.

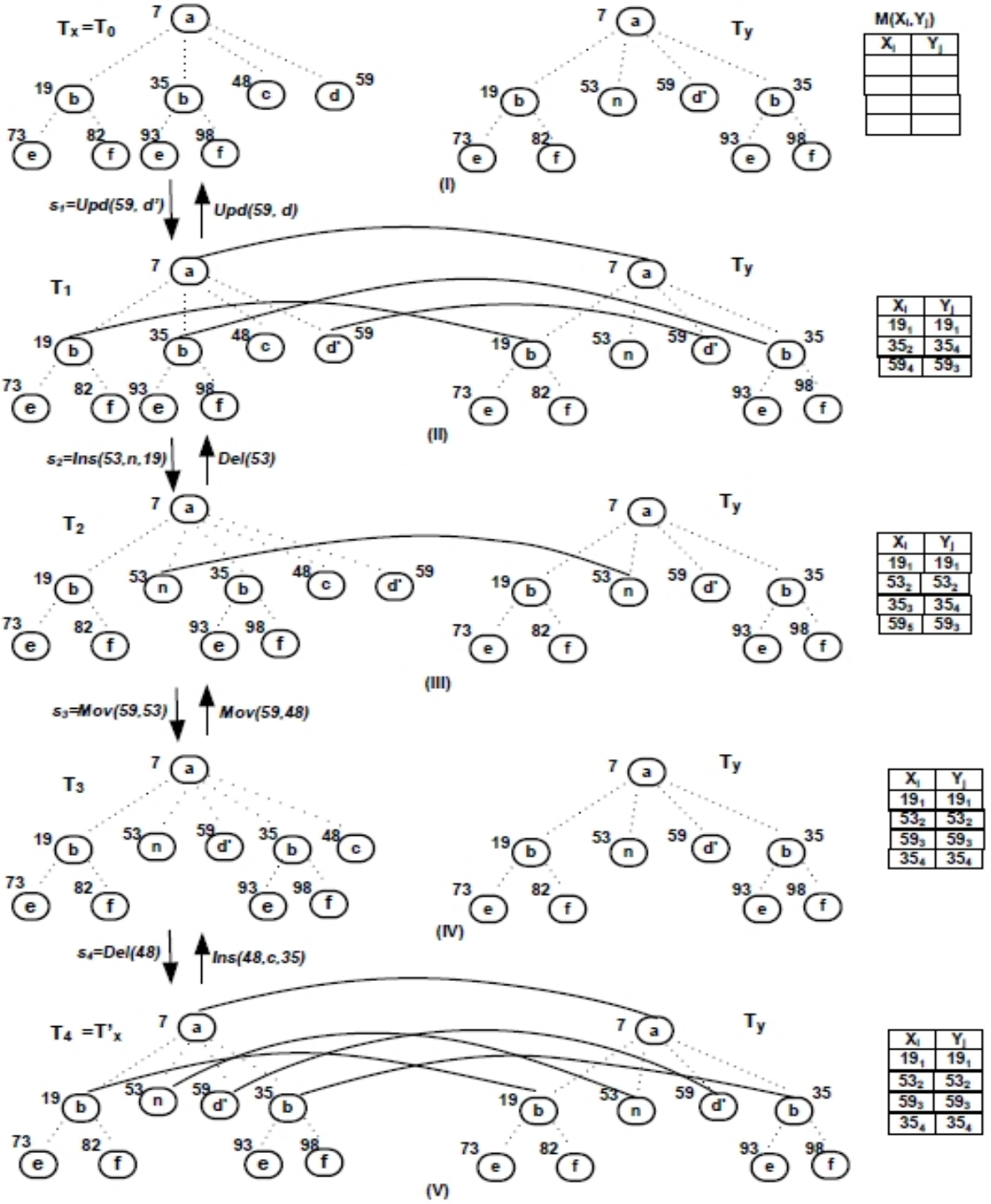


Figure 4.16: (I) The tree T_x to be transformed into T_y (of Figure 4.9). (II,III,IV,V) The transformed trees T_1, T_2, T_3, and T_4 after edit operations $Upd(59, d')$, $Ins(53, n, 19)$, $Move(59, 53)$, and $Del(48)$ respectively and $T_4 = T_y$.

Insert: To find the inserted nodes in T_y, we take the nodes, w of T_y such that w is not a peer in any of the pairs in M (Figure 4.14, line 5). For each such w we add the edit operation $Ins(w, val_w, k)$

to S, meaning w will be inserted after node k in T_x with the encrypted value val_w (Figure 4.14, line 5 (c)). The position k is determined with respect to the sibling relationship of already matched pairs of M (Figure 4.14, line 5 (a)). In particular, the peer node x_i of T_x in $M(x_i, y_j)$ for which w is the immediate right sibling of y_j, is the node k in T_x (Figure 4.14, line 5 (b)). We apply the insert operation to T_x and add the node pair, (x_{k+1}, w_{j+1}) to M (Figure 4.14, line 5 (d)). If w is the first sibling in T_y, i.e., leftmost child, then k is considered to be the dummy child node of the level in question of T_x. In effect, an insertion operation changes the sibling positions of existing peer nodes of T_x in M. The running example of Figure 4.16 (III) shows the resulting tree T_2 after $Ins(53, n, 19)$ and the updated sibling positions of peer nodes in M. For clarity, only the new solid line resulted for the new matched pair is shown in the figure.

Move: In this case, the pairs of M for which a peer node's sibling positions are not the same are considered. If it is the case, we say peer nodes are missarranged (Figure 4.14, line 6). In the Figure 4.16 (III) nodes $35, 59$ in T_2 are missarranged with respect to their respective sibling positions in T_y as depicted in M. We add move operations to S to arrange the sibling order. We explain the details in Section 8.4. In the running example a $Mov(59, 53)$ is added to S, and applied to T_2 to transform it to T_3 (Figure 4.16 (III) to (IV)). Note that no new match is found by this operation, however the sibling position is changed as depicted in M.

To address the inter level moving as shown in Figure 4.13 a simple approach is followed. For a node $x \in T_x$ which is detected as deleted in a level, the $exist(x, -, U, -)$ function can be called in T_y for other levels in T_y. If a non null value, y, is returned then $FindSibling(y)$ is called further to get the sibling position k for which y is the immediate right sibling (like an insert case). Now we can replace the $Del(x)$ with the $Move(x, k)$ in S.

Delete: To find the deleted nodes of T_x, we take the nodes x in T_x such that x is not a peer in any of the pairs in M (Figure 4.14, line 7). For each such node x, we say that either it is deleted from the level it was in T_x or it is moved to some other level. Given our partial comparison objective as motivated before we can safely conclude the former. Accordingly we can add the delete operation $Del(x)$ to S and apply it on T_x which in turn changes the existing sibling positions in M as insertion and move cases (Figure 4.14, line 7 (b) (c)). Figure 4.16 (V) shows the resulting tree T_4 after performing $Del(48)$ on T_3.

After applying the edit script $S = (Upd(59, d'), Ins(53, 19), Move(59, 53), Del(48))$ the first level of the initial tree T_0 is transformed into T_4 which is isomorphic to T_y with respect to the first level and M contains the matched peer nodes in that level. In Figure 4.16 the tree T_4 happens to be level-wise isomorphic to T_y for the next level also and thus document-wise isomorphic.

To ensure that the edit script generated by the algorithm is of minimum cost, we must first find the shortest sequence of moves to arrange the siblings as will be realized in the *ArrangeSibling()* function. This is explained below.

8.4 Finding a Shortest Sequence of Node Movement for MCES

As mentioned in the case of move and shown in Figure 4.16 (III) there might be miss-arranged peer nodes in M. In general, there may be more than one sequence of moves that will arrange the siblings in

the order of the edited tree as illustrated in the following example.

Example 7. Consider, Figure 4.17 that shows the siblings of Figure 4.16 (III). There are at least three sequences to arrange the sibling nodes of T_2 to transform it into T_3. The first consists of moving nodes c and d' after n in that order. The second consists of moving the node b after d'. The third consists of moving the node d' after n. All yield the same transformed tree. □

Clearly, the first edit script is undesirable as it requires more moves and thus concedes more cost. However, to pick the desired one among the other two having only one move is also tricky as the former has direct children as opposed to the latter and thus the former potentially requires more moves than the latter. In case of several sequences having the same number of moves any one can be picked.

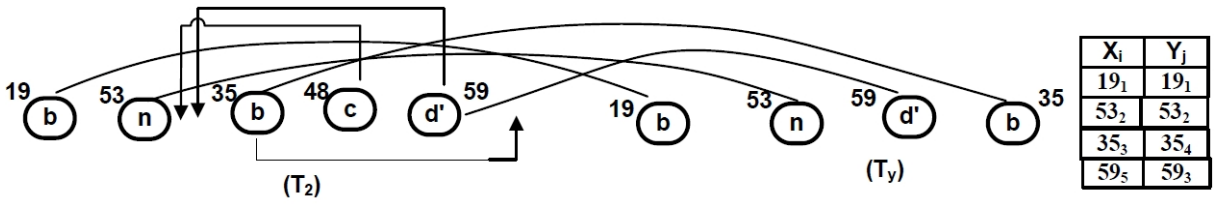


Figure 4.17: An appropriate match with rearranging the sibling nodes.

In this context, we may utilize EBOL-based identifiers for XML nodes that are integers following an equality from left to right in a sequence of siblings. The idea is to find a common sequence of sibling nodes that maintain this inequality and move the nodes that are not in the common sequence. This strategy might be useful for a potentially smaller number of miss-arranged nodes, indicating that most of the sibling nodes are in order with respect to the EBOL inequality. However, we can not rely only on the EBOL inequality property as there might be several missarranged nodes due to multiple insertion, deletion and moving of the nodes. As such, we use the second criteria of Section 8.2 as part of the common sequence definition.

Let the sequence of EBOLs of siblings in a level of T_x and T_y be $L_x = x_1, ..., x_i$ and $L_y = y_1, ..., y_j$ where i and j are the respective positions of the peer nodes x and y in the sibling order and let $(x_i, y_j) \in M$ for any $i, j \in \mathbb{N}$. A subsequence of L can be found by removing any number of EBOLs from L.

Definition 10. Longest Sibling Subsequence (LSS): The longest sibling subsequence of L_x and L_y, denoted by $LSS(L_x, L_y)$ is a sequence $L_{xy} = (x_1, y_1)...(x_i, y_i)$ of matched pair nodes such that (1) $x_1...x_i$ is a subsequence of L_x; (2) $y_1...y_i$ is a subsequence of L_y; (3) for $1 \leq k \leq i$, $y_k = exist(x_k, -, U, -)$; (4) for $1 \leq k \leq i$, $\frac{same(x_k, y_k)}{max(|x_k|, |y_k|)} > t$; where $0 \leq t \leq 1$; and (5) there is no sequence L' that satisfies the previous four conditions and longer than or equal to L_{xy}. Note that the condition four refers to the second matching criterion.

We are required to rearrange the sibling nodes, x_i in T_x, which are not in the longest sibling subsequence L_{xy}, i.e., $x_i \notin L_{xy}$, using move operations. Accordingly, we move such nodes, x_j of T_x, which transforms T_x into an isomorphic tree of T_y. In Figure 4.17, for $L_x = \{19, 53, 35, 59\}$ and $L_y = \{19, 53, 59, 35\}$, the *LSS* is $\{19, 53, 35\}$ which corresponds to the matching pairs $(19_1, 19_1), (53_2, 53_2), (35_3, 35_4)$ respectively. Then the shortest sequence of move operations is $Mov(59, 53)$ as 59 is not in $19, 53, 35$.

In [CRGMW96], a similar strategy is used to determine the shortest sequence of move operations. Due to the fourth condition which only considers the direct children as opposed to all the descendants, we are able to determine the shortest sequence of moves without parsing and loading the descendants into memory. We also use the algorithm of [Mye86] in a similar fashion to [CRGMW96] that can compute the *LSS* of two sequences in time $O(ND)$, where $N = |L_x| + |L_y|$ and $D = N - 2|LSS(L_x, L_y)|$.

9 Complexity Analysis of the MCES Algorithm

The complexity of the MCES algorithm is analyzed space and time wise. Let the number of nodes in a level of T_x and T_y be m and n respectively; and let d be the maximum number of levels of both trees.

Space: As mentioned in Sections 3 and 6 EBOL-based XML annotation avoids the need for an explicit hierarchical representation of XML nodes in memory. Clearly the algorithm requires less memory as opposed to existing techniques. As such, the space complexity is only about the nodes that are stored in the FIFO queue. Let the average space required for a node x and its EBOL identifier be s_x and s_e respectively. As mentioned the MCES algorithm does not require the memory of the parent, child and sibling relationships when taking one level of nodes of T_x to compare with one level of nodes of T_y. So, the required space is proportional to $O(ms_xns_y{s_e}^2)$ which would have been $2d$ times this proportion if the full trees and their normalized trees would be in memory as in [CRGMW96]. In particular, the timed proportion of the algorithm of [CRGMW96] is $(2d + C)$, where C is $(S_h + S_s)2d$ (S_h and S_s are the memory space required for parent-child and sibling relationships respectively).

Running Time: Let c_d and c_e be the average cost of the functions $diff()$ and $exist()$ respectively; let p and q be the number of comparisons required to find the right position of a pair in M and for the function *FindSibling()* (i.e., to find the node in T_x after which a node is inserted or moved) respectively.

The *update* case takes time $O(m(c_e + c_d)p)$ to match the nodes of a level of T_x with the nodes of a level of T_y. Now, let n_x and n_y be the number of unmatched nodes of a level of T_x and T_y respectively for the *update* case. From the assumption of Section 8.2, we know that such a number is bounded by $< t$. Consequently, n_x and n_y nodes will be matched by *delete* and *insert* cases respectively.

The *insert* case running time is proportional to $O(nc_epq)$ and for the *delete* case it is $O(mc_ep)$.

For the *move* case, it requires to compute the $same()$ function on the FIFO queues for the missarranged nodes x and y. From the same assumption of Section 8.2, we know that the number of nodes having identical EBOLs is bounded by $> t$ and as such considering the number of missarranged nodes is $|X|$, we may approximate the time taken for moving nodes of T_x is $O(|X|qpt)$.

Based on this analysis we prove the following theorems.

Theorem 1. The MCES algorithm computes a *minimum cost edit script* S and does it in time $O(N)$ where N is the maximum number of nodes in a level of T_x and T_y.

Proof: To proceed with the proof we simplify the individual edit case analysis as follows. We take $N = max(m, n)$ instead of m, n, and use $C_{ed} = c_e + c_d$.

With this simplification the running time for *update*, *insert*, *delete* and *move* cases are converted from $O(m(c_e+c_d)p)$, $O(nc_epq)$, $O(mc_ep)$ and $O(|X|qpt)$ into $pO(NC_{ed})$, $pqO(Nc_e)$, $pO(Nc_e)$ and $|X|pq$ respectively. p, q and $|X|$ refer to constant amount of work by the algorithm. As such combining the latter three cases we get the time proportional to $O(Nc_e)$ which can then be combined with the *update* case time and the total time required for the algorithm is $O(NC_{ed} + Nc_e)$. Further simplifying C_{ed} and c_e to a constant time the algorithm takes $O(N)$ in total.

We now show that S is a *minimum cost edit script*. Any edit script computed by the MCES must contain at least one of the following:

- One update operation for each node $x_i \in T_x$ such that $y_j = exist(x_i, -, U, -)$ and $val_{x_i} = val_{y_j}$;
- One insert operation for each unmatched node in T_y;
- One delete operation for each unmatched node in T_x; and
- One move operation for each pair of matched nodes $(x_i, y_j) \in M$ such that $i! = j$ (considering leaf nodes).

This is straightforward that MCES generates one edit operation for each of the above mentioned operations. Now consider the nodes having direct children stored in the FIFO queue. Those nodes need to be moved using using the shortest possible sequence of moves. The function *ArrangeSibling()* that in turn uses the longest sibling sequence *LSS* conditions ensures such a shortest sequence. Hence, S is a *minimum cost edit script*.

Theorem 2. The edit script S that is generated by the MCES algorithm transforms T_x into a document-wise isomorphic tree of T_y.

Proof: We prove this inductively by edit cases during a one pass breadth first traversal of one level of nodes, $x_i \in l(T_x)$ and $y_j \in l(T_y)$. Recall that, one level of nodes are stored in the respective FIFO queues. In the *update* case a first set of matching pairs $M = (x_i, y_j)$ are determined for all nodes x_i with a y_j that have identical EBOLs.

Consider that the unmatched nodes of $l(T_x)$ are deleted from T_x in the *delete* case and similarly the unmatched nodes in $l(T_y)$ are inserted into T_x in the *insert* case. At this point, the $l(T_x)$ contains exactly the same nodes of $l(T_y)$. The miss-arranged nodes get arranged in the *move* case. As nodes are deleted, inserted and moved the respective sibling positions are ensured by the invoked *UpdateMatch* and *FindSibling* functions. So, the algorithm transforms one level of T_x into a level-wise isomorphic to T_y.

Inductively, similar arguments apply for the further levels of comparison. So after the algorithm runs for all the levels of T_x and T_y, T_x is transformed into a document-wise isomorphic tree of T_y.

10 Related Work

There has been remarkable progress in recent years regarding data modeling and interface designing by XML-based technologies. Most of the work leverage a shared XML structure as data models and interfaces, for instance, schemas and DTDs as can be found in the literature [BLL04, DdVPS01, DdVPS02, FCG04, KMR05, MTKH03, KB08, MS03, KE06]. As mentioned in Chapter 2, there is an implicit assumption in these work, that is XML structure-based formats are shared amongst peers. As such peers interact within a close boundary, the thus mostly focus on XML structure-based access control issues as opposed to dealing with a variety of vocabularies and evolution of the formats for instance. Few other works [JWST, PSC03, FJK+08, YdmGM05, ASTK06] aim at going beyond the XML structure by introducing an ontology for instance. Similar to previous work they also focus on access control issues as described below.

10.1 XML-based data and interface modeling

10.1.1 XML structure-based approach for client/server interaction

In these approaches [BLL04, DdVPS01, DdVPS02, FCG04, KMR05, MTKH03], a document provider peer acts as the server and other peers are clients while all peers share common XML structures for interfaces so as to interpret the documents. As such, the server enforces access control policies on a per request basis. Instead, our work focuses on delegating to third parties the selective delivery of semantically equivalent content to authorized users independently of providers (detailed in Chapters 5 and 6).

10.1.2 Encryption of XML for peer interaction

In the work of [KB08, MS03, KE06], XML document portions are encrypted with a shared key before they are sent to other peers. Peers possessing appropriate decryption key can read those document portions. As such, although peers may not have any agreed data interface based on the XML structure of documents, the shared key is acting as an implicit interface between peers in that they exchange whatever document portions that can be encrypted/decrypted with that key. In [MFBK06], authors also rely on encryption amongst peers who update document portions in a collaborative fashion. Our approach in this chapter is fundamentally different as we leverage an ontology describing the document contents as an interface between peers. The annotations of the document nodes ultimately provide the actual interpretation of the documents.

10.1.3 Ontology-based approach

For given XML schemas, normalized XML elements are defined in [JWST] which are used as the basis for further access control. In [PSC03], authors model an XML document as an entity and find relations

with other such entities. Based on this semantic relationship of entities they generate a data interface specific to each pair of entities that are exchanged between peers.

As mentioned in Chapter 2, these work do not use an ontology for representing data semantics that are carried by documents. Instead, our proposed ontology specification models the data semantics exchanged in an execution of an agile business scenario and thus making annotated documents interoperable beyond business boundaries.

10.2 Document Comparison

Tree structured data comparison techniques are in general based on ideas from the string matching literature [Mye86, Wag75, WF74, WMMM90]. A comprehensive survey of edit script computation, known as tree edit distance, can be found in [Bil05]. Due to inherent differences of tree structured data from flat data these techniques vary from different dimensions, namely atomic vs bulk edit operation, order of operation, ordered vs unordered trees, key vs keyless data and usage of intermediate trees. In the following, we elaborate on these techniques.

10.2.1 Tree edit distance based approach

The authors in [YKT05] proposed a tree edit distance between two trees should be computed based on the so called 'string edit distance' and in [YKT05] the authors suggest another approach called 'binary branch distance'. Both techniques require intermediate normalized trees of the source trees to be computed. In particular, for [YKT05] two sequences of nodes by pre-order and post-order traversal and for [YKT05] two binary tree representations of the source trees are required. Our MCES algorithm requires only a single pass to find a *minimum cost edit script* and does not require any intermediate form of trees except a FIFO queue storing only one level of tree nodes.

10.2.2 Tree-based edit operations

In [JWZ94, SZ97, ZS89], authors support insertions and deletions anywhere in a tree whereas in [JWZ94] insertion is supported only before deletion. In [Sel77] insertion and deletion of single nodes at the leaf level and updating of nodes anywhere in a tree are allowed. In [CRGMW96] a subtree movement (bulk operation) for the ordered trees is introduced. For unordered tree comparison, authors in [CGM97] introduce comparison techniques including copy operation. Instead, we define four atomic edit operations (i.e., update, insert, delete, and move) with respect to a single node; to be more specific a sibling node of a level of an XML document making partial comparison possible for fine grained documents.

10.2.3 Matching algorithm complexity

The matching algorithms to find initial matches of node pairs for ordered trees are presented in [SZ89, ZS89]. The algorithm of [ZS89] runs in $O(n^2 log^2 n)$ which is further improved by the authors in [CRGMW96] as their matching algorithm runs in $O(ne+e^2)$; where n and e are the number of leaf nodes and the 'weighted edit distance' respectively. The *minimum cost edit script* algorithm of [CRGMW96] runs in $O(ND)$ time; where N is the total number of nodes of the two source trees and D is the number of miss-arranged nodes. The fundamental difference of our proposed algorithm of Section 8 with the work of [CRGMW96, ZS89] is that we do not consider any initial set of "matches of node pairs" of the source trees which would then need to be parsed fully (against our motivation of partial comparison). The "matches of node pairs" are computed as a side effect of the *minimum cost edit script* computation. Our proposed MCES algorithm requires less computation time, i.e., runs in $O(N)$, compare to [CRGMW96] that takes $O(ND)$ time; where N is the maximum number of nodes of the two levels of the source trees.

10.2.4 XML data characteristics

The comparison technique of [CRGMW96] for the ordered trees assumes that a tree node contains keyless data for which comparison is considered to be harder than the same problem of ordered trees. In addition, several domain characteristics are considered (e.g., semantic tagging of nodes in the source trees, nearly no duplicate tree nodes) to find appropriate matches. We consider XML data to be keyless, but our EBOL-based identifiers for XML nodes associate unique identifiers to the parsed nodes in a level. Our only assumption is that the number of similar nodes between two levels of source trees is greater than that of dissimilar nodes which is reasonable for "Enterprise XML" as mentioned before.

11 Conclusion

In the first part of this chapter, we demonstrated how a business domain ontology can be used in order to enable a stable interface and presented an approach for a semantic enabled document parsing. The actors in a *DocWF*, first agree on an ontology describing the content exchanged in the documents then a semantic enabled document parsing technique is used to annotate documents with adequate semantic information. Upon receipt of such documents, *DocWF* actors can understand the content irrespective of diverse vocabularies used in the documents for instance.

The latter part of this chapter presents document comparison solutions for "Enterprise XML" as identified as a vital requirement for interoperable documents. In particular, document comparison is a fundamental requirement, for instance, for document convergence and document versioning etc. as required to be performed by a third party, for instance, a communication infrastructure node. Our approach allows comparing not only plain text XML trees but also XML trees containing sensitive information. As such, this approach is equally applicable in agile business scenarios where interacting peers may exchange sanitized documents due to access control and confidentiality requirements for instance. Moreover, the matching algorithms for producing reusable *minimum cost edit script*s are shown to be better with regard to memory and required runtime compared to existing solutions.

The work of this chapter has been published in the proceedings of IFIP SEC 2009, 24th International Information Security Conference, May 18-20, 2009, Pafos, Cyprus [RRMS09] and The Fourth International Conference on Internet and Web Applications and Services (ICIW 2009), May 24-28, 2009 - Venice/Mestre, Italy [RRS09b].

Chapter 5

Communication Infrastructure of Document-based Agile Workflows

Tact is the art of making a point without making an enemy.
- Isaac Newton -

1 Introduction

In this thesis so far, we introduced the basic methodologies of document-based agile workflows (*DocWF*) and associated interoperability mechanisms. *DocWF* methods first enable business actors to design loosely coupled process models based on business goals and associated rules without requiring upfront knowledge of precise business tasks and their binding to services. Suitable business tasks can then be determined by semantic matchmaking of goals with semantic descriptions of existing process models as maintained in the knowledge-base (*KB*) of an actor. For an execution of these tasks by suitable services, a dynamic task enactment takes the states of existing services into account in a formal model before invoking them. The interoperability mechanisms of a *DocWF* include techniques for semantically annotating "Enterprise XML" and comparison solutions of "Enterprise XML" to enable non-disruptive document exchanges beyond business boundaries. Yet, these methods and techniques do not provide any guarantee on exchanges of fine grained documents amongst actors as determined from policies defined by the document providers. Considering the absence of a centralized coordinator akin to distributed environments, a decentralized communication infrastructure for an execution of a *DocWF* is necessary to assure fine grained document exchanges amongst a priori unknown actors. Relying on an ontology-based stable interface amongst actors, the communication infrastructure must address the following:

1. **Loosely coupled document exchanges:** Business actors must be able to exchange fine granular documents independently of each other.

2. **Selective document routing and delivery:** Multiple fine grained documents associated with the same ontology *concept* may originate from multiple providers. As such, those semantically equivalent document portions should be routed and delivered to appropriate peers according to a provider's policy.

3. **Fully distributed communication:** Communication amongst a priori unknown actors must be supported by a decentralized communication infrastructure as opposed to a dedicated coordination engine.

Loosely coupled document exchanges come from the organization of an agile workflow application. For instance, producers and consumers of documents may not know each other a priori and may not be available at the same time. This implies that, although being the policy maker for the document, its producer will not be able to check its policy in real time for each legitimate access request. Thus a business actor must be able to consume documents independently of the original document producers in order to advance towards a goal. Publish/Subscribe-based communication has been the main enabler to achieve such a loose coupling between publishers and subscribers. Publish/Subscribe-based distributed systems have been well studied in the literature [WJL04, RRL08, PFJ+01, EFGK03]. However, in the context of an ontology driven document exchange, basic methodologies for semantic based document publishing, subscription, and selective delivery (or dissemination) are limited. We propose a Publish/Subscribe-based decentralized communication infrastructure, denoted by a dissemination network, that hosts fine grained documents published by document providers and performs policy checks on behalf of the providers. Based on the policy checks the dissemination network routes fine grained documents between the hosts of the network which then deliver those to legitimate peers and thus document exchanges are called selective routing and delivery (also called selective dissemination). The routing and delivery of documents are semantic-based as fine grained document portions to be routed and delivered are determined through $concept\ containment$ (c.f. Chapter 4) and the ontology-based authorization approach respectively.

Regarding distributed communication amongst actors, business actors may operate from varying business boundaries and may not know each other a priori. As such, they can not rely on a central authority (i.e., a centralized coordination engine) for routing and delivery of multiple fine grained documents for the same *concept* for instance. This calls for a distributed setting of a dissemination network that is realized by a set of special entities called document distributors. This decentralized setting in turn raises several co-ordination challenges. Existing coordination protocols to support the execution of traditional collaborative applications are focused on assuring reliable transaction of message exchanges among business peers as in [MM05]. Such protocols thus do not consider the additional coordination required amongst the distributed set of coordination engines, i.e., distributors, and between an actor and a distributor. Coordination is required between distributors for selective dissemination to legitimate actors. It also involves retrieving an up-to-date version of a document by a distributor and maintaining a consistent version of a document by all distributors. Coordination between actors and the infrastructure is also required in order to manage actor subscriptions and to compute a secret key. From a recipient's perspective, several other challenges are also raised such as interpreting the semantics of several XML vocabularies and combining different documents into a composite one before proceeding further in a *DocWF* execution. As a result, new solutions for such coordinations need to be designed that offer

adequate features to support the execution of document-based agile workflow applications. In this chapter, we present an ontology driven decentralized communication infrastructure and associated protocols to address above mentioned issues. The communication infrastructure is based on Publish/Subscribe methodologies.

The rest of this chapter is organized as follows. Section 2 introduces a cross border scenario to discuss the dissemination network. An overview of this infrastructure is provided in Section 3. Section 4 extends the structural annotation method of Chapter 4 with security annotations in order to enable semantic-based selective document routing and delivery by the nodes of the network and for later security verifications by recipients. To enable the dissemination network to check policies on behalf of document providers, an ontology-based document access control model is described in Section 5. Section 6 elaborates on the ontology driven Publish/Subscribe-based dissemination network supporting distributed communication. Section 7 illustrates some integration techniques for interpreting multiple XML vocabularies. Section 8 positions our work with related literature and finally section 9 concludes the chapter.

2 A Cross Border Car Accident Scenario

Consider a car driver holding a license plate of EU country A while driving in a motorway of country B is caught exceeding the speed limit and subsequently causes a car accident. The motorway police (MP) and community police (CP) of country B rush to the spot and find the driver badly injured. CP immediately calls an ambulance of a local hospital (LH) for emergency medical help. Local media agencies (MA) rush there to cover news which will then be distributed to other agencies including foreign ones. MP notifies the accident to the corresponding authority of country A (PA) and requests driver information. PA looks up into its database and finds previous motor accident history of the driver that occurred in other countries and sends those to MP. Upon receipt of those information, MP consolidates all accident histories and then files a case in the local court (LC). To resolve cost claims by the driver, car and medical insurance authority (IN) requires information regarding car details and medical expenses that can be provided by the car seller and LH respectively. After several months, the case will be in the court requiring all details that can be provided by MP, CP, MA and PA. LC tries to find facts and evidences derived from those details for judicial proceedings. A Publish/Subscribe-based decentralized communication amongst the actors can be sought as described in the following.

Publish/Subscribe-based communication: Consider the car accident ontology excerpt of Figure 5.1 is publicly shared thus known to the actors, i.e., MP, CP, LH, MA, PA, LC. The operations required for publishing the document portions associated with some *concept*s and subscription of *concept*s can be described as follows:

- **Publishing documents associated to *concept*s:** MP publishes document portions associated with the "DriverRecord" and "CourtCaseReport" *concept*s. PA publishes document portions associated with the "MotorwayPoliceRecord" *concept*.
- **Subscribing documents mapped to a *concept*:** PA and LH subscribe to "DriverRecord" and "CarAccident" respectively. LC subscribes to "DriverRecord", "NewsAgencyRecord" and "MotorwayPoliceRecord".

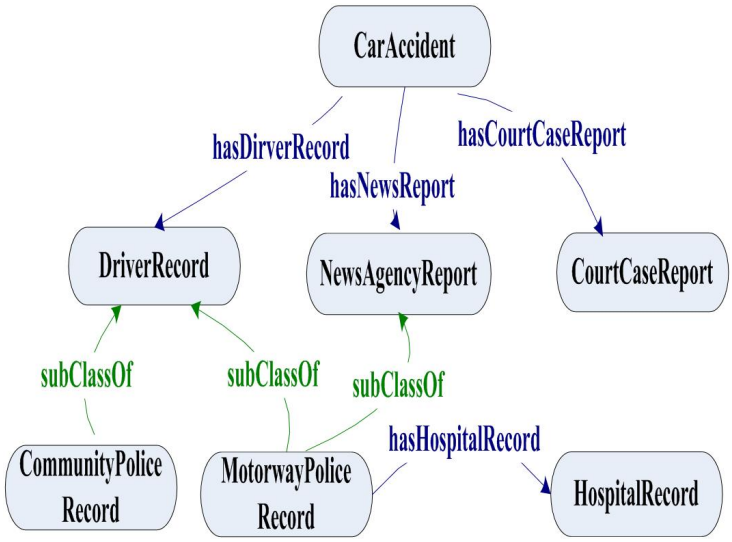


Figure 5.1: A car accident ontology excerpt.

As peers require fine grained documents originated from various authorities as opposed to complete documents, it requires a selective document dissemination amongst actors.

Selective document distribution: Parties providing information have individual XML schemas to comply with their organization and country specific policy, regulations which may not allow them to exchange full documents or all portions to everybody. For instance, CP may not disclose the driver's license plate, her social security number and insurance information to MA due to legal bindings or her bank information to IN as it can be missused. MP will not disclose the driver's exceeded speed limit and her previous accident records to LH as these are not required for medical attention, however, these could be important for court (LC) proceedings which may then be authorized to receive those.

Non-disruptive document exchanges: As can be imagined, the number of interacting parties may increase over time, for instance PA of other countries and intelligence agency (IA) may get involved in the case by providing a history of accident reports of the driver. As PA is from another country and IA is already protecting its data model they do not have any common data format with others. After a while, IA takes over the case from MP and CP as the driver is incidentally found to be a suspect of being a threat to national security. Due to proprietary data exchange formats amongst PAs and the following take over by the IA, the MP and CP (including IA) have to restructure their data models which in turn may invalidate their existing data exchange formats. However, the information must be available whenever required for later court (LC) proceedings and police cases irrespective of its publication time.

Business rule-based document processing: LC receives various details related to the car accident in a variety of formats and vocabularies from CP, MP, MAs, IA and possibly PA. Then LC determines facts and evidences from the received documents by correlating, combining, comparing different document portions based on some customized business rules. For instance, LC considers document portions related to the car accident, i.e., the police record of CP and MP, news reports of MA, the accident history of PA in order to give a verdict. One rule for such a consolidation would be: if some driver with a license plate number X is found to drive over the limitation on a 70 mph speed limited motorway in the police record of MP with the same license plate number and a MA has a news report confirming the date and time of

the incident, then the alleged driver is subject to punishment with a fine.

3 Solution Overview

A solution overview is described in terms of actors of the dissemination network, their interaction and features of the network.

3.1 Actors of the Dissemination Network

The dissemination is based on a shared ontology that models the content semantics of business domain documents including their relationships and to which every system participant agrees to (Figure 5.1 sketches a car accident ontology excerpt motivated by the scenario before). As mentioned in Chapter 4, the definition (or nomination) of a shared ontology is the prerequisite for any interaction between *DocWF* actors. The dissemination system distinguishes three kinds of actors (Figure 5.2):

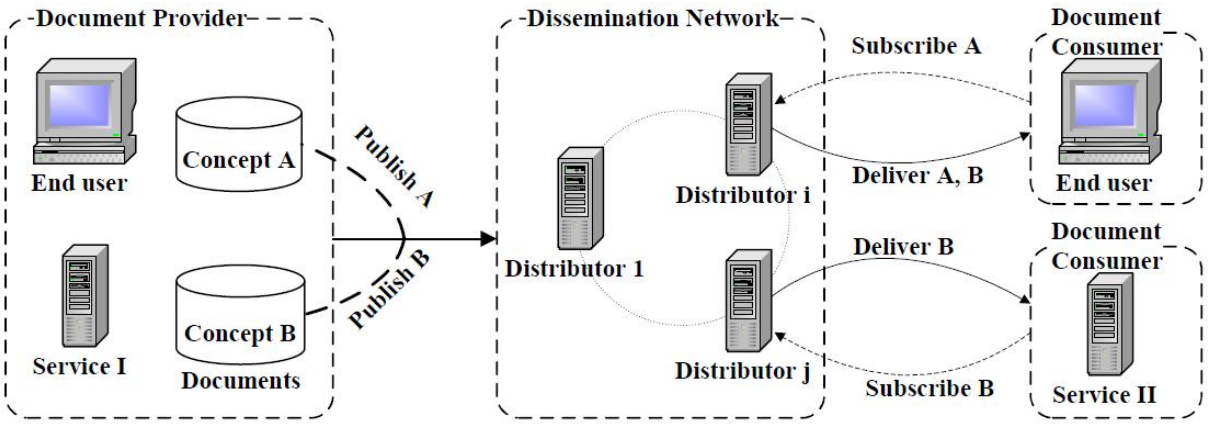


Figure 5.2: A Publish/Subscribe model of an ontology-driven decentralized communication infrastructure for fine grained document exchanges.

1. **Document providers:** Peers that publish encrypted and annotated XML document portions are document providers.

2. **Document consumers:** Peers that subscribe to *concept*s and receive encrypted and annotated XML document portions according to policies of the providers are document consumers.

3. **Distributors:** A distributor is a piece of software running either within or outside corporate boundaries, i.e., nodes of the network, and thus is distributed. A distributor manages subscriptions and realizes selective document routing and delivery. It thus takes care of the actual content delivery to document consumers.

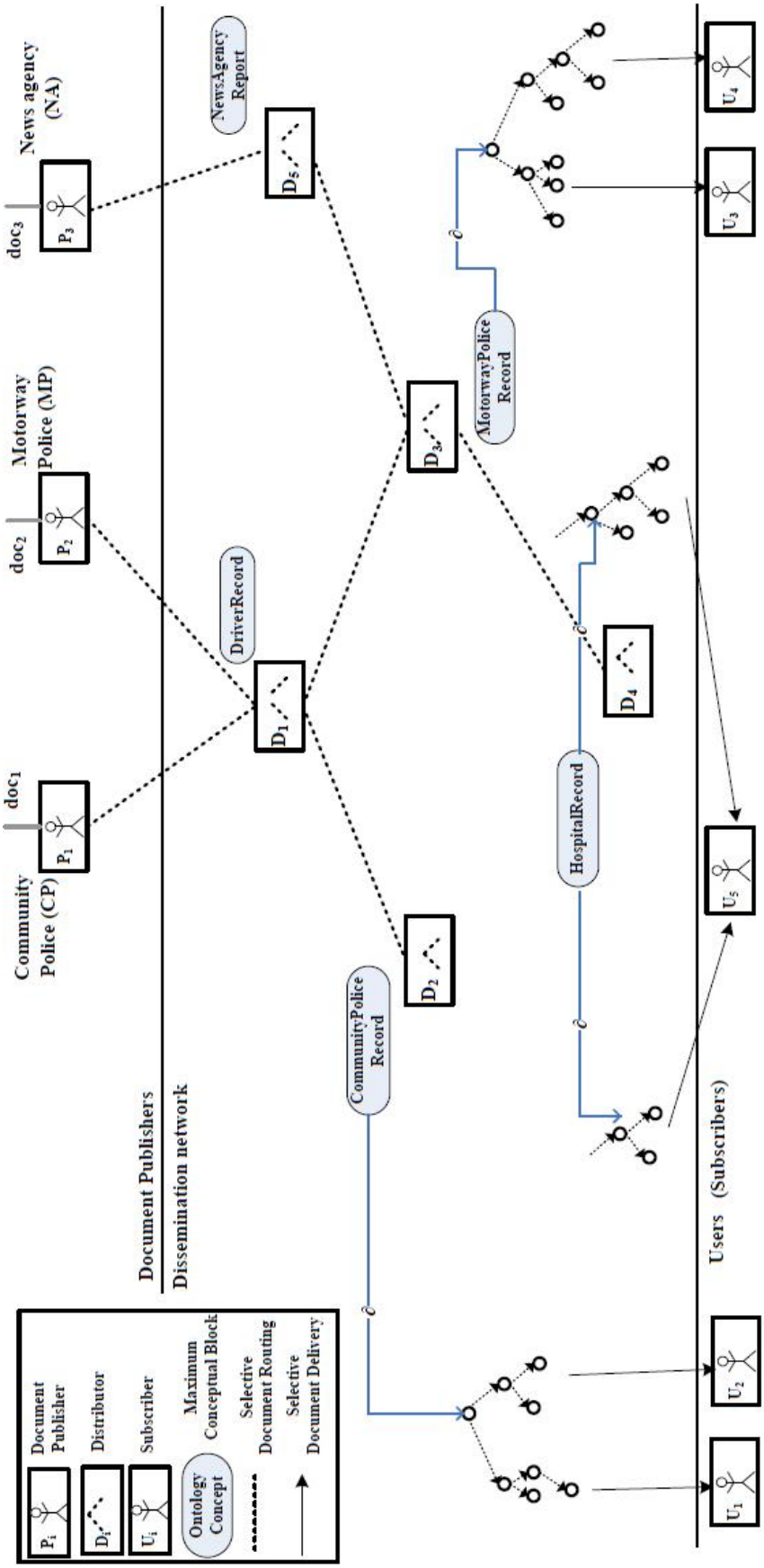


Figure 5.3: Ontology-driven Publish/Subscribe-based document exchanges between a priori unknown document publishers and subscribers.

In Figure 5.2, the *concept* A contains B, i.e., $A \preceq B$, and the peer (e.g., end user) is an authorized subscriber of the *concept* A. Then the peer may receive documents associated with the *concept*s A and B depending on the policy checks by the distributor.

Figure 5.3 shows ontology-driven Publish/Subscribe-based document exchanges between a priori unknown peers based on the model of Figure 5.2. Peers U_1 and U_2 subscribe to *concept*s, $Community\text{-}PoliceRecord$. Peers U_3 and U_4 subscribe to *concept*s, $MotorwayPoliceRecord$ and peer U_5 subscribes to *concept*, $HospitalRecord$.

3.2 Features

The decentralized communication infrastructure that we developed in this chapter for a *DocWF* application features the following:

- **Publish/Subscribe-based decentralized communication:** A document provider (i.e., creator) is a publisher and a consumer peer of the document is thereby called a subscriber. Document providers do not care about routing authorized content to the subscribers and subscribers are not concerned about the providers (c.f. Section 6).
- **Semantic-based document publishing and subscription:** Publishers send documents annotated with meta data including security meta data (Section 4) to the dissemination network. Subscribers subscribe to ontology *concept*s rather than to associated documents. Distributors of the dissemination network check policies of the publishers by a semantic enabled policy checking technique (Section 5) before delivering fine grained documents to legitimate peers.
- **Selective document routing and delivery:** The dissemination network consists of a logical set of document distributors hosted in the nodes of the network. Distributors form a routing topology in accordance with the agreed domain ontology and this topology eventually decouples publishers and subscribers. Different mapped portions of a published document are routed to appropriate distributors for initial hosting. A concerned distributor then delivers fine-grained documents to the authorized subscribers by pruning document nodes according to policy checks (c.f. Section 6).

3.3 Interaction Overview

The actors interact as follows (see Figure 5.4):

1. Prior to the first document publication, a publisher needs to provide authorization policies that determine user authorizations and which will be enforced by the distributors. Publishers may also issue inference rules describing constraints, for instance, unsubscription and separation of duty rules over their plain policies.
2. An end user sends a subscription request for a *concept* with valid credentials to a distributor which in turn checks associated policies (provided by the publishers) and triggers the computation of a

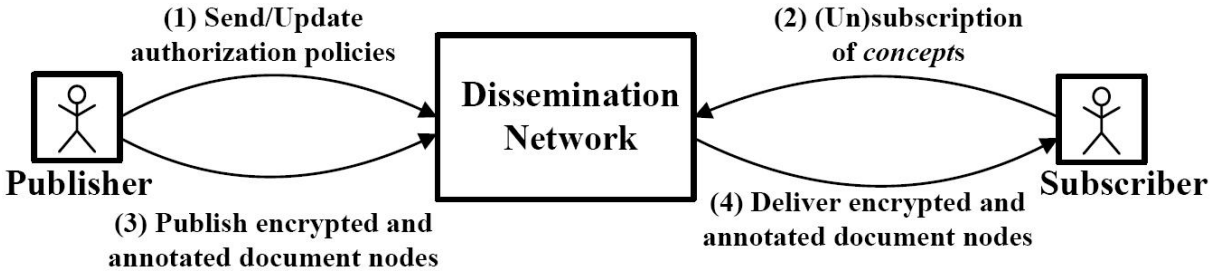


Figure 5.4: Loose coupled interaction between a priori unknown document providers and consumers.

secret key for every group of authorized subscribers to the same *concept* (detailed in Chapter 6). Unsubscription might be done on a user request or be triggered by a distributor according to an inference rule (e.g., if the user credentials expire or if authorization policies change) provided by document providers.

3. The publisher of an XML document annotates XML document nodes with semantic and security meta data, encrypts the nodes in a stipulated granularity with the secret key computed for the *concept* (detailed in Chapter 6), and sends the encrypted nodes along with their annotations to the dissemination network.

4. Distributors follow two protocols (immediate delivery and catch-up delivery) as part of required coordination in order to route and deliver fine grained document portions selectively to other distributors and all authorized subscribers. Immediate delivery is triggered by the publication of a document and catch-up delivery is triggered by an authorized subscription to *concept*s. Distributors extract the relevant encrypted nodes by matching the subscriber's authorized *concept*s with the annotations of the nodes and thus can route and deliver appropriate document nodes without decrypting the nodes (Section 6).

5. Subscribers can verify the received XML content by decoding annotations and checking the document content both semantically and structurally (detailed in Chapter 6).

In order to enable a distributor to extract appropriate encrypted nodes according to a provider's policies and subscriptions, providers need to annotate nodes and attach annotations to encrypted nodes before publishing. The next section describes such annotations of enterprise XML nodes.

4 Annotating Sensitive Documents

Basic annotation elements of Chapter 4 are extended to be able to annotate an "Enterprise XML" with security meta data such as hash values of subtrees of nodes for their integrity protection (Figure 5.5). Similarly to basic annotations, these security annotations are also extensible. While mechanisms are shown for the finest granular level of an XML document, i.e., a node, they can be equally applied for coarser grained documents, i.e., subtrees. The novelty of this mechanism lies in the way the annotation elements are assembled, which allows a recipient to verify the semantics of a node and security properties when decoding the nodes in a reverse order (c.f. Chapter 6).

Node Identifier: Let x be a node in d_i. The node identifier of x denoted by N_x is a tuple formed by three elements $(doc_{id}, E_x, E^x_{Highest})$, where doc_{id} is the document identifier of d_i, E_x is the EBOL of x, $E^x_{Highest}$ is the highest EBOL in the document portion rooted at x.

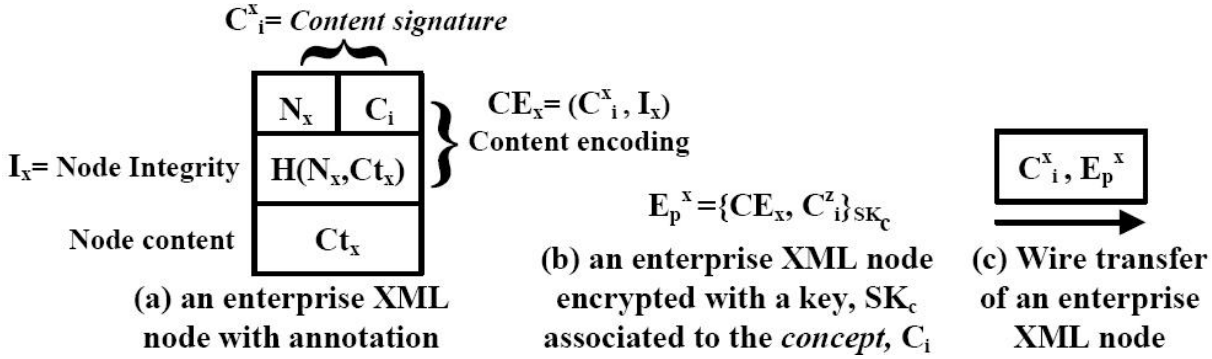


Figure 5.5: Annotation, encryption and wire transfer of a fine grained "Enterprise XML" node.

As opposed to the node identifier of the basic annotation method, this node identifier consists of the encrypted versions associated to a BOL, i.e., E_x and $E^x_{Highest}$, including the document identifier. As before, a node identifier is unique for all documents in the system. The depth included in E_x uniquely determines the node's level. E_x and $E^x_{Highest}$ together determine the parsed document portion. Finally, doc_{id} resolves appropriate XML nodes of the associated document with respect to the same *concept*.

Node Integrity: The node content consists of attributes, their values and text content inside the tag but not any descendants of the node. The node integrity is a hash computed out of the concatenation of a node identifier and content, denoted by $I_x = H(N_x, Ct_x)$, where N_x is the node identifier, Ct_x is the content of x, and H is a one way collision resistant hash function (Figure 5.5 (a)).

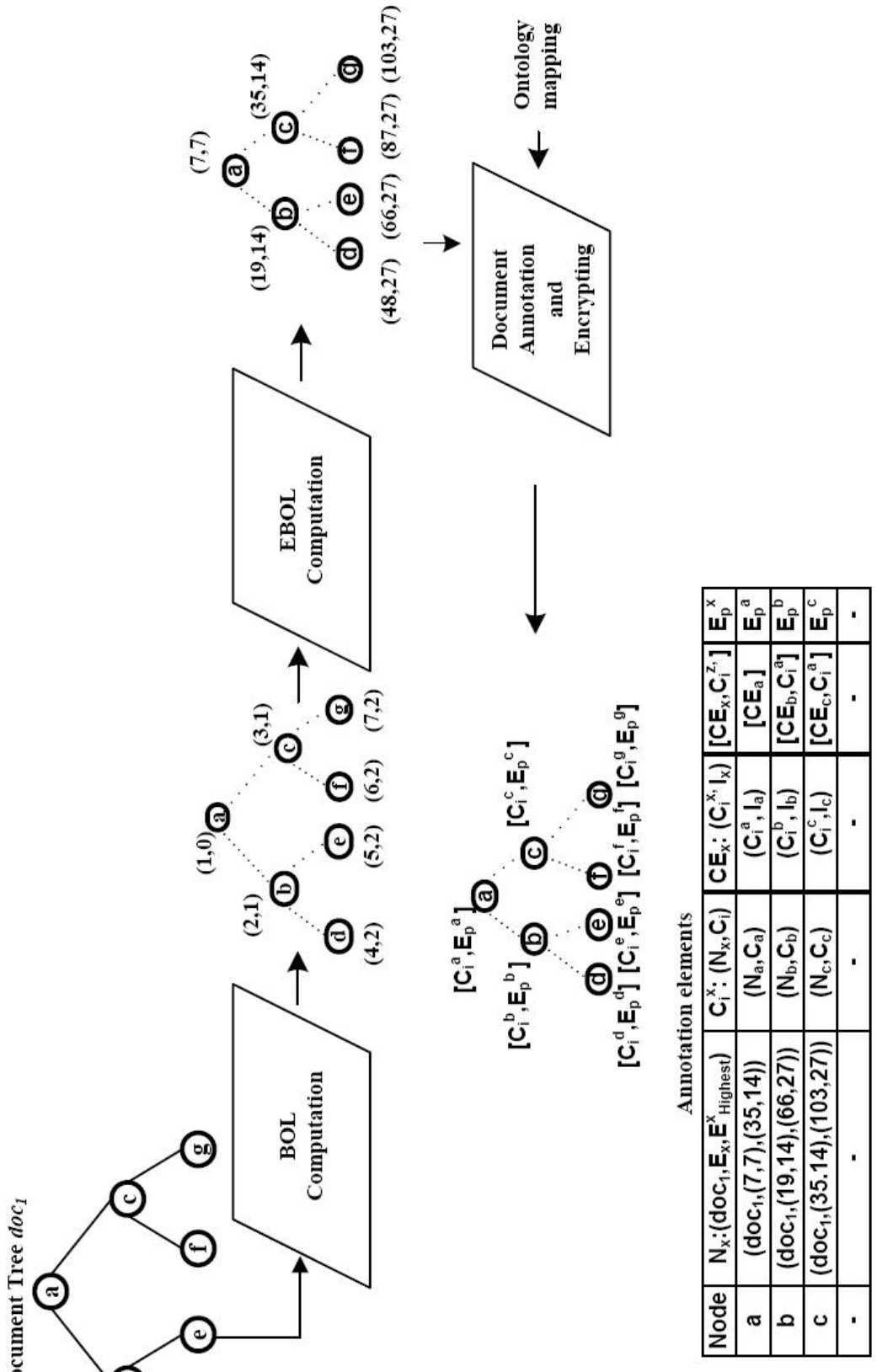
Content Signature: As defined in Chapter 4, a *content signature* of the node x, denoted by C^x_i, is a pair (N_x, C_i), where N_x is the node identifier of x and C_i is a *concept* mapped to x (Figure 5.5 (a)).

In order to relate an integrity value with its node a content encoding is defined as follows.

Content Encoding: An encoding information CE_x of a node x is $CE_x = (C^x_i, I_x)$, where C^x_i is the *content signature* and I_x is the node integrity respectively.

The content encoding can be attached to a node as its annotation. However, to protect from any pruning of such an annotation from its node, this is further related with its parent node's *content signature* as follows. Each XML node x is encoded as a pair $[CE_x, C^z_i]$, where CE_x is the content encoding of node x and C^z_i is the *content signature* of the parent node z of x. For the root node of a document the encoded node is $[CE_x]$.

To protect from unauthorized disclosure of the content above mentioned content encoding is encrypted with a key (Figure 5.5 (b)). The key is associated with the ontology *concept* that is mapped to the content and subscribed by a group of peers. Peers compute the key in a distributed fashion (detailed in Chapter 6). An associated *content signature*, i.e., C^x_i, is further attached with the encrypted content to enable a distributor to process it (Figure 5.5 (c)). This processing includes the following operations as detailed in this chapter:



Annotation elements

Node	N_x:(doc_1,E_x,$E^x_{Highest}$)	C_i^x: (N_x,C_i)	CE_x: (C_i^x,I_x)	[CE_x,C_i^z,]	E_p^x
a	(doc_1,(7,7),(35,14))	(N_a,C_a)	(C_i^a, I_a)	[CE_a]	E_p^a
b	(doc_1,(19,14),(66,27))	(N_b,C_b)	(C_i^b, I_b)	[CE_b,C_i^a]	E_p^b
c	(doc_1,(35.14),(103,27))	(N_c,C_c)	(C_i^c,I_c)	[CE_c,C_i^a]	E_p^c
-	-	-	-	-	-

Figure 5.6: Annotation and encryption steps of "Enterprise XML" by document providers.

1. **Selective document dissemination:** Routing, extraction and delivery of fine grained XML documents based on its *content signature* (c.f. Section 6).

2. **Subscription management:** Subscription and unsubscription of a peer (c.f. Section 6).

Document Encryption: After encryption, an enterprise XML node x is represented as $[C_i^x, E_p^x]$, where C_i^x is the *content signature* of x and E_p^x is the encrypted value of the content encoding pair $[CE_x, C_i^z]$ of the node x (Figure 5.5 (b)).

Finally, the actual assembled enterprise XML node, x, with its annotation sent by a provider is $[C_i^x, E_p^x]$ (Figure 5.5 (c)). Figure 5.6 depicts the annotation and encryption process of XML nodes using EBOL as described above. A distributor can then further perform selective routing and delivery based on a semantic enabled policy checking as described in the next section.

5 Semantic Enabled Policy Checking

As mentioned in Chapter 1, whenever new applications and organizations hit the market to compete they will use different policy specifications and a variety of policy languages even in the same business domain. As shown in [KPS$^+$04], an ontology-based policy specification can constitute a common framework for integrating various policy specifications with a minimal impact by reducing, for instance, policy language translation from one to another. In the context of this thesis, this translation affects are multiplied by document subscriptions based on content semantics and fine grained document publications with annotations. Moreover, policy checks can not be done by providers themselves. As such, we leverage an ontology-based policy specification maintained by the distributors of the dissemination network. The result is as follows:

1. **Flexible policy specification:** Our approach enables a flexible policy specification allowing a provider to change its policy whenever needed.

2. **Leveraging ontology-based inferences:** Our approach enables delegated parties such as distributors to check authorization policies provided by the publishers by leveraging the inference power of an ontology.

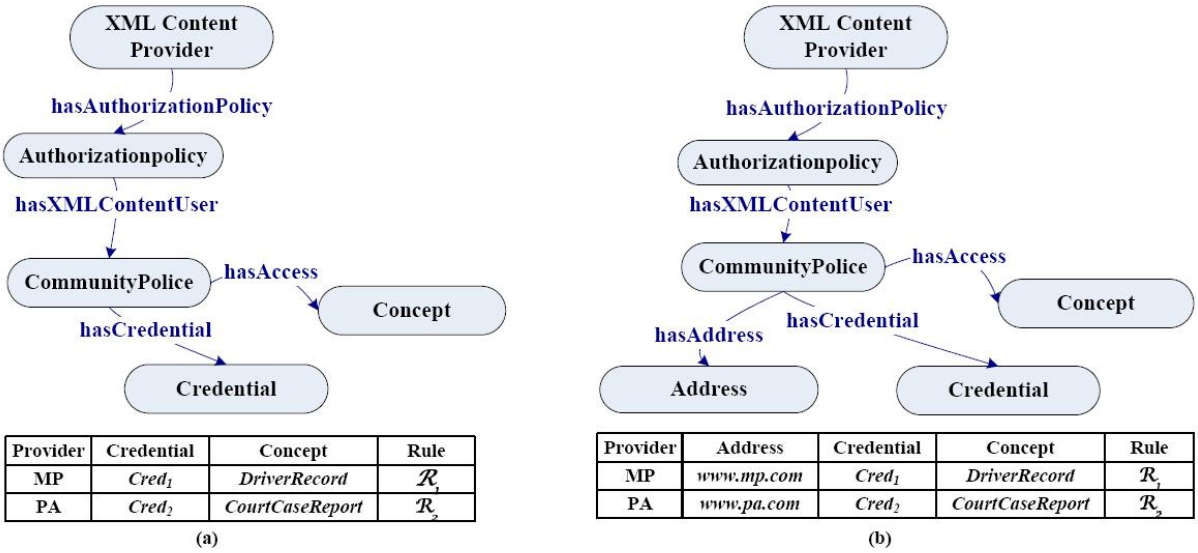
3. **Ontology-based fine-grained document exchanges:** Our approach enables semantic-based selective XML content routing and delivery by distributors.

Given a domain ontology as illustrated in Chapter 4, business peers determine the *concept*s for which the mapped XML document portions could be sensitive. This allows them to specify an authorization policy on the *concept*s which can then be realized at the level of the XML data structure by a peer independently of other peers. The next section describes how such policies can be specified and can be updated by a peer.

5.1 Semantic-based Authorization Policy Specification

Peers describe an ontology-based authorization policy as a set of explicit rules which apply globally in the system. The construction of a rule is as follows ($[x+]$ is used to denote a non-empty set of elements of type x).

1. Rules take the general form $[user\_credentials, [C_i]+, \mathcal{R}]+$ stating that access over one or more *concept*s identified by C_i is allowed to a user holding $user\_credentials$ provided $\mathcal{R}$ is true.
2. The expression $\mathcal{R}$ is an inference rule, characterizing relationships and constraints verified by browsing the semantic graph (such as of Figure 5.1). In particular, for the same pair of $user\_credentials$ and C_i, different rules may imply different authorizations or prohibitions. This expression enables a publisher to restrict access to content annotated with eligible concepts *concept*s of the ontology, and may be parameterized by $user\_credentials$ or elements of $[C_i]+$, as described in Section 5.2.



Provider	Credential	Concept	Rule
MP	$Cred_1$	*DriverRecord*	$\mathcal{R}_1$
PA	$Cred_2$	*CourtCaseReport*	$\mathcal{R}_2$

(a)

Provider	Address	Credential	Concept	Rule
MP	*www.mp.com*	$Cred_1$	*DriverRecord*	$\mathcal{R}_1$
PA	*www.pa.com*	$Cred_2$	*CourtCaseReport*	$\mathcal{R}_2$

(b)

Figure 5.7: The policy ontology is maintained by the distributors (i.e., communication infrastructure nodes). (a) An initial ontology-based policy. (b) Evolved policy ontology after adding the *concept* "Address".

As opposed to the business rules of Chapter 3, such rules are taken into account for fine grained document access control. These rules are enforced by encryption resulting into selective document routing and delivery as described in Section 6. The table of Figure 5.7 (a) shows an example of an ontology-based policy specified by two XML content publishers MP and PA of the car accident scenario of Figure 5.1 as depicted below.

Example 1. $\mathcal{R}_1$ for the user with credential $Cred1$ is: if a user is allowed to access the *concept* $DriverRecord$ then he is also allowed to access all the contained *concept*s of $DriverRecord$. $\mathcal{R}_2$ for the user with credential $Cred2$ is: the user is allowed to access the *concept* $CourtCaseReport$ if he has access to the *concept* $MotorwayPoliceRecord$.

To implement the flexible rules $\mathcal{R}_1$ and $\mathcal{R}_2$, SPARQL [SPQ] queries are used as described in the next section. Regarding the need for a flexible policy specification, any change in the policy such as adding an address parameter for request filtering or adding meta data about providers in the policy ontology of Figure 5.7 (a) can be done irrespective of the variety of vocabularies to represent an address that might be used locally by providers. In particular, the policy ontology of Figure 5.7 (a) can be changed by introducing additional ontology *concept*s (e.g., address) and relationships (e.g., hasAddress) among them as in Figure 5.7 (b). Moreover, inference rules make policy changes easy as peers may change only by, for instance, introducing new rules or changing associated inference rules as shown in the example below.

Example 2. PA changes the inference rule $\mathcal{R}_2$, expressing that the peer is allowed to access $CourtCase\text{-}Report$ if any other provider provides the peer the same access right.

The inference rule can be easily applied over an ontology due to the natural inference capability of an ontology. This in turn allows a distributor to automate policy checks on behalf of the providers, which results into the granting of fine-grained access rights over documents and thus the distribution of documents in a selective fashion. This policy checking using inferences of ontologies a distributor needs to traverse both domain and policy ontologies. This is illustrated in the next section.

5.2 Policy Checking by a Distributor

As described before, at the distributor level, authorization policies express access rights over the agreed domain ontology *concept*s, which first need to be evaluated against the subscription requests of peers. Then for a positive evaluation the resulting authorized *concept*s must be grounded to the mapped XML document level. In this section, we describe this process, denoted as semantic access control, that leverages the inference power of ontologies.

5.2.1 Semantic Access Control for Selective Document Delivery

To check authorization policies distributors maintain two kinds of ontologies:

1. **Domain ontology:** Publicly shared domain ontology (as in Figure 5.1) that specifies the semantics of document content.
2. **Policy ontology:** An ontology describing the authorization policies (as in Figure 5.7) of the providers.

Upon a subscription request a policy check requires reasoning on these ontologies and it is a three step process:

1. **Determine candidate *concept*s:** Determine concerned *concept*s by traversing the domain ontology graph. Such *concept*s will be checked for potential authorizations of a peer.

2. **Find relevant authorization policies:** Find relevant authorization policies to check the authorizations of the *concept*s of step one by traversing the policy ontology graph.

3. **Check authorizations:** Apply the policies over the *concept*s of step one to determine the authorized *concept*s.

We suggest SPARQL [SPQ] to implement such policy checks. SPARQL queries over the domain ontology graph allow a distributor to determine candidate *concept*s through reasoning on the semantic graph patterns. SPARQL queries over the policy ontology are also used to evaluate the policies by computing aggregated authorized *concept*s for a user. A distributor uses two types of SPARQL queries to perform the abovementioned steps. The result of the first query is feed into the second query and the evaluation of the latter gives resulting authorized *concept*s. To this effect, distributor nodes have to host an engine like Joseki [JOS] to interpret SPARQL queries. The first query is crafted to find *concept*s which a user can be implicitly granted access to starting from one *concept* to which the user is explicitly granted access. The result of such a query consists of a set of *concept*s related through a *concept containment* relationship and that should equally be granted access according to the provider's policy. A distributor uses an XACML engine to receive a subscription request for *concept*s and to return a response (i.e., Permit/Deny) to the user. In case of a "Permit", the distributor responds by sending the *content signatures* (c.f. Section 4) associated with the accessible *concept*s.

Figure 5.8 depicts required components for a distributor to perform policy checks and is illustrated in the following sections.

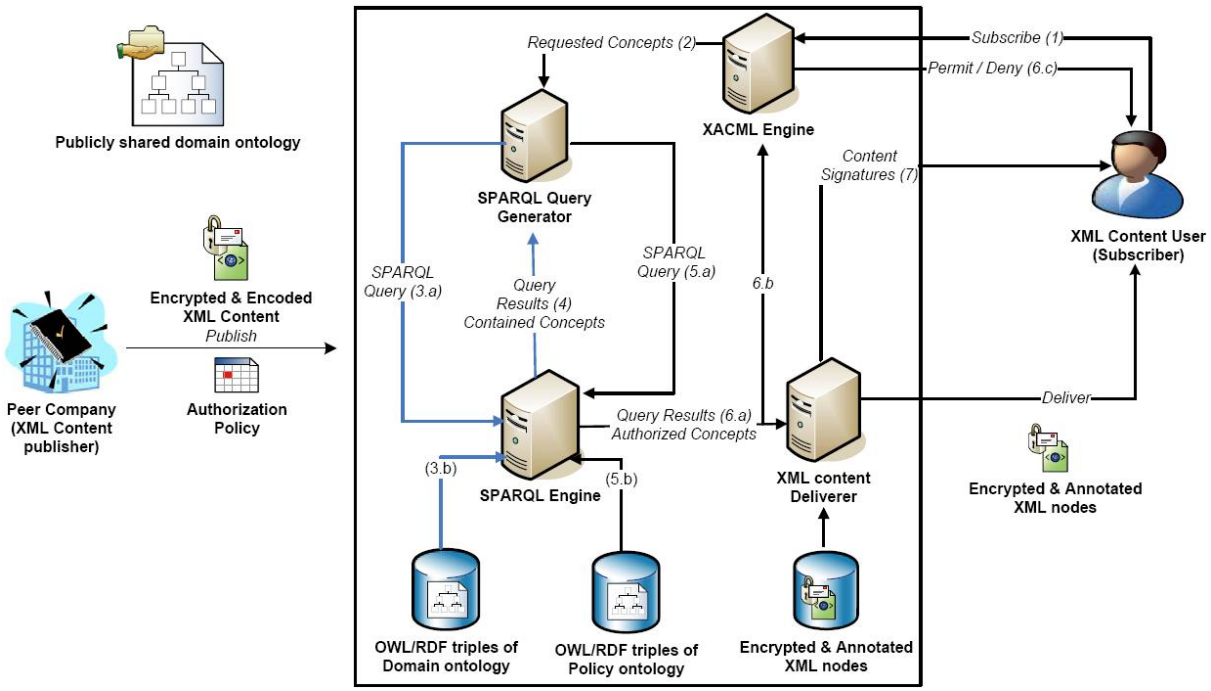


Figure 5.8: Ontology-based selective XML content distribution system; a distributor. The numbered lines depict sequence of operations upon a subscription request and blue lines represent recursive steps.

5.2.2 Determine candidate *concept*s

The first phase of the process begins starts with a subscription request for a *concept* (see Figure 5.8).

- The XACML engine receives a subscription request for a set of *concept*s (1).
- The XACML engine forwards such a request to the SPARQL generator (2) to convert it into SPARQL queries (3.a) using the requested *concept*s over the shared domain ontology represented as OWL triples (3.b). For instance, a subscription request for the *concept* $DriverRecord$ from a user with credential $Cred1$ is converted into the following SPARQL query by the query generator:

```
PREFIX  ca: <http://www.owl-ontologies.com/
Ontology1223675912.owl#>
SELECT ?subClasses
WHERE { ?subClasses rdfs:subClassOf ca:DriverRecord. }
```

- A distributor determines all the contained *concept*s of the requested *concept*s (by *concept containment*) to get all the candidate accessible *concept*s. The above SPARQL query returns all the subclass *concept*s of $DriverRecord$ (4), i.e., $CommunityPoliceRecord$ and $MotorwayPoliceRecord$. If any of these resulting *concept*s also have subclass *concept*s then similar queries are performed recursively. To this end, multiple candidate *concept*s are determined while an initial request might only be for one *concept*.
- In case the user does not request for specific *concept*s, all *concept*s in the ontology are candidate *concept*s to be evaluated further. In particular, a similar query of step two for the most general *concept* should be performed to determine all *concept*s in the domain iteratively.

5.2.3 Find relevant authorization policies and check authorizations

- In order to determine the authorized *concept*s for a requesting user, the above query result (i.e., $CommunityPoliceRecord$, $MotorwayPoliceRecord$) is then used into a further SPARQL query (5.a) which evaluates associated policy triples from all providers (5.b).
- The result of this query is the maximal set of aggregated *concept*s (possibly empty if none is permitted) that are accessible to the requester (6.a). The inference rule $\mathcal{R}_1$ of the publisher MP described previously allows the user to access the subclass *concept*s. The following query is used to evaluate this rule:

```
PREFIX po: <http://www.owl-ontologies.com/
 Ontology1224765032.owl#>
SELECT  ?concept

WHERE{
      {?user po:hasCredential po:Cred1}
      {?user po:hasAccess ?concept.}
 }
```

The first triple in the WHERE clause determines the users with credential $Cred1$ and the second triple determines the accessible *concept*s for those users. If the result set contains the $Community\text{-}PoliceRecord$ and $MotorwayPoliceRecord$ *concept*s then the XACML engine returns a "Permit" response to the user (6.b,6.c).

- The XML content deliverer in the system then extracts the *content signature*s of the authorized *concept*s by manipulating only the encrypted and annotated content for the requested user and sends those as a response to a successful registration (7).
- Otherwise, none of these *concept*s is accessible to the requester and the XACML engine simply denies access (6.c).

6 Decentralized Communication Infrastructure for a *DocWF*

Document semantics as represented by a domain ontology are the only interface among peer organizations, and drive the XML content exchanges. Distribution is performed by selective document routing and delivery as mentioned in Section 3. In this section, we elaborate on this ontology-driven Publish/Subscribe-based decentralized communication infrastructure by first describing its initialization followed by communication protocols between peers to implement the required coordination.

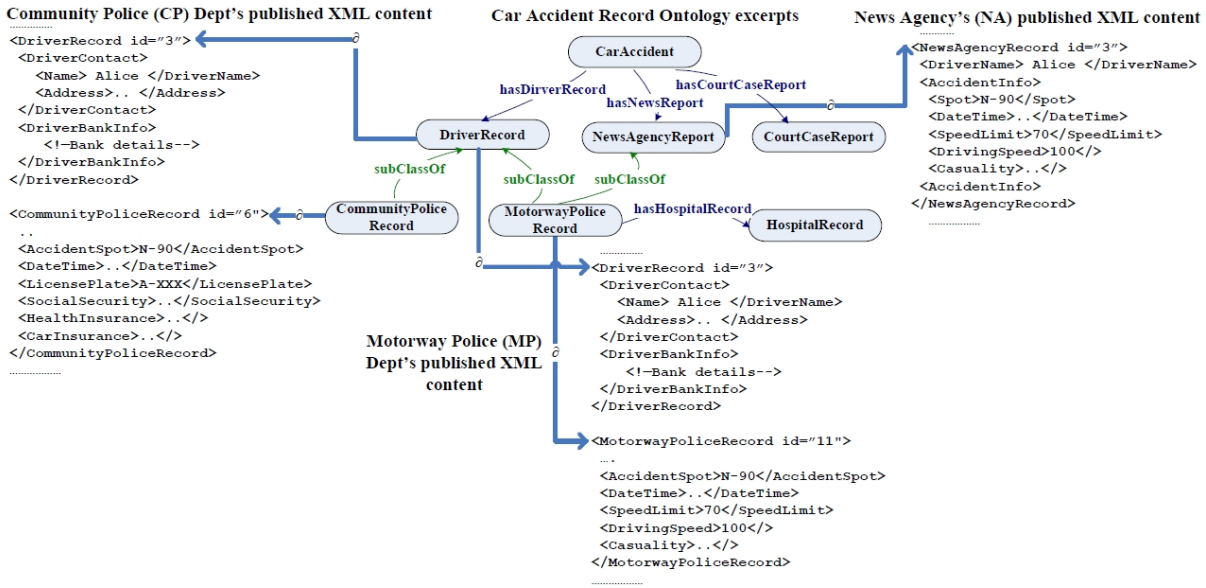


Figure 5.9: A car accident ontology excerpt and ontology *concept*s mappings to the individual data models of community police (CP), news agency (NA), and motorway police (MP).

6.1 Initialization

The initialization of the infrastructure sets the responsibility of distributors for document routing and delivery and can be done as soon as the domain ontology is agreed upon (i.e., everyone has the knowledge

of the publicly shared ontology). We define the $Maximum\ Conceptual\ Block$ as the representation of a distributors's initial responsibility of document distribution. It determines a collection of *concept*s for which mapped encrypted document portions are hosted and can be disseminated by a distributor.

Definition 1. $Maximum\ Conceptual\ Block$: Let C_i be a *concept*. The maximum conceptual block for C_i is the set of all *concept*s that are reachable by following a succession of $concept\ containment$ from C_i.

Example 3. Figure 5.9 shows a collection of *concept*s $\mathcal{C} = \{CarAccident, DriverRecord, NewsAgencyReprot, CourtCaseReport, CommunityPoliceRecord, MotorwayPoliceRecord, HospitalRecord\}$ of the car accident ontology. $Maximum\ Conceptual\ Block$s of $DriverRecord$ and $NewsAgencyReprot$ are $\{CommunityPoliceRecord, MotorwayPoliceRecord\}$ and $\{MotorwayPoliceRecord\}$ respectively. Community police (CP) and motorway police (MP) map $DriverRecord$ *concept* to their individual document portions rooted at `<DriverRecord>`. The *concept*s $CommunityPoliceRecord$, $MotorwayPoliceRecord$ and $NewsAgencyReport$ are mapped to corresponding XML document portions of CP, MP and NA respectively. □

Lemma 1. $Maximum\ Conceptual\ Block$s are always monotonically decreasing.

Proof: Let C_i and C_j be two *concept*s such that C_i contains C_j; let M_i and M_j be the $Maximum\ Conceptual\ Block$s for C_i and C_j respectively. As C_i contains C_j the number of classes reachable from C_i is always more than that of C_j. Therefore $M_j \subset M_i$. Transitively for any *concept* C_k such that $C_j \preceq C_k$ then $M_k \subset M_j$.

Distributors are initialized in the following ways:

1. **Set distribution responsibility:** One $Maximum\ Conceptual\ Block$ is assigned to a distributor for dissemination of mapped XML content. In accordance with the monotonicity of $Maximum\ Conceptual\ Block$ of the *concept*s the responsibility of a distributor for storing, routing and delivering of XML content is determined. Thus distributors having more storage and computing capability can be assigned to disseminate more *concept*s than others. In case of equally capable distributors, assignments can be random.

   To illustrate, let D_i, D_j be two distributors where D_i is having more storage and computing ability than that of D_j. Let M_p and M_q be the $Maximum\ Conceptual\ Block$s for *concept*s C_p and C_q respectively such that $M_p \subset M_q$. Then M_q is assigned to D_i.

2. **Set routing paths:** Each distributor (including publishers) maintains a distributed hash table (DHT) where the key fields and the values are the *concept*s representing the $Maximum\ Conceptual\ Block$ and references (i.e., URL/IP) of other distributors respectively. The ordering of the key fields are determined as follows: each $Maximum\ Conceptual\ Block$ is assigned in a key field in a monotonically decreasing fashion and the reference addresses of the next distributors are assigned in the value fields for each such key.

   The next distributors are further defined into two categories: uplink (U) and downlink (D) distributor as follows. Let D_i, D_j be two distributors that disseminate two $Maximum\ Conceptual\ Block$s represented by *concept*s C_i, C_j respectively. If $C_i \preceq C_j$ holds then D_i is an uplink (U) distributor of D_j and D_j is a downlink (D) distributor of D_i. D_i puts C_j as downlink distributor such that $C_i \preceq C_j$ and C_k as uplink distributor such that $C_k \preceq C_i$ as its key and corresponding references as value fields in its DHT respectively.

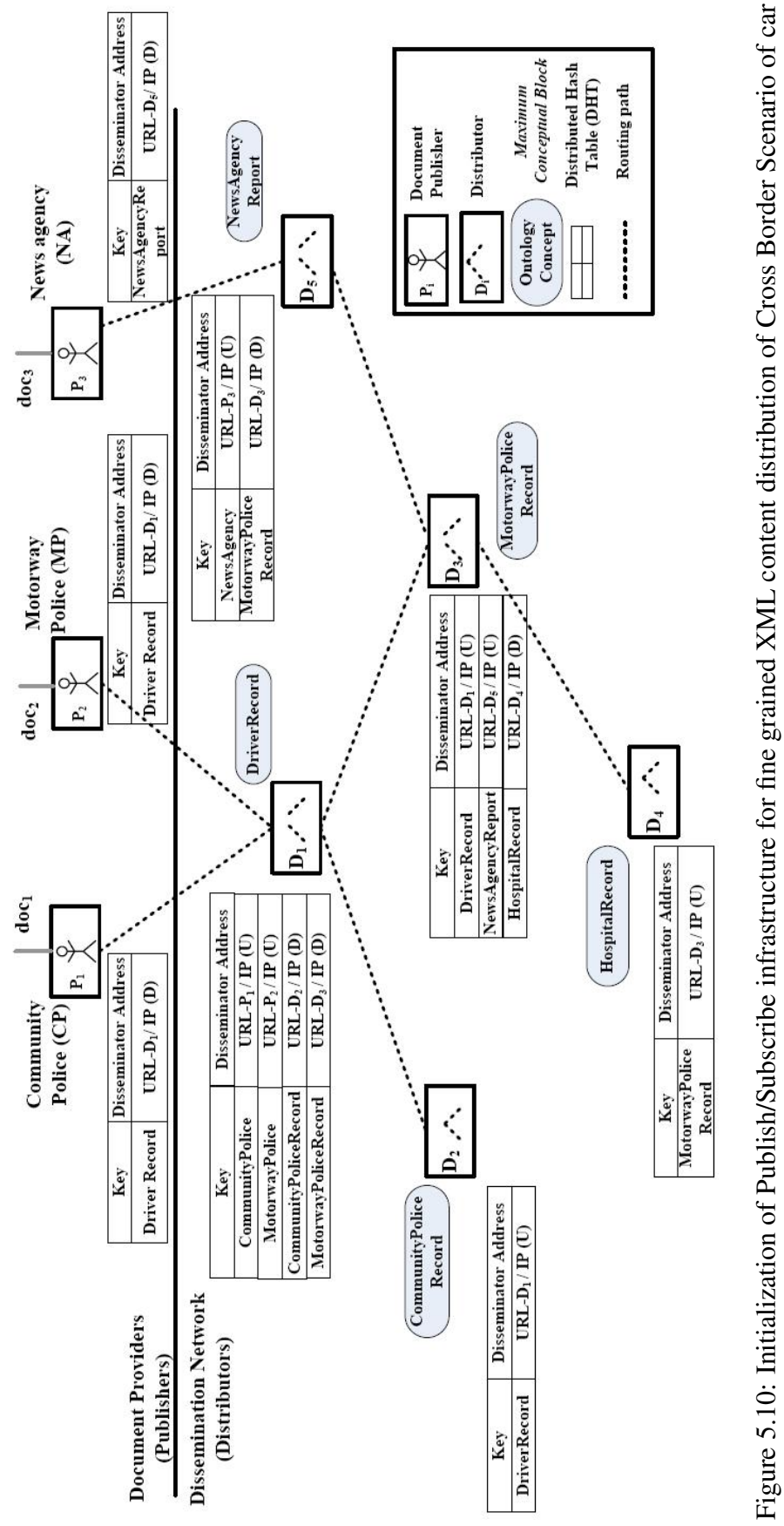


Figure 5.10: Initialization of Publish/Subscribe infrastructure for fine grained XML content distribution of Cross Border Scenario of car accident.

In effect, distributors form a directed acyclic graph (DAG) topology based on $concept\ containment$ where document publishers comprise multiple publishing points in the dissemination. Figures 5.10 and 5.3 show such a dissemination topology. The following example depicts an initialization.

Example 4. In Figure 5.10, the distributors D_1, D_2, D_3, D_4 and D_5 are initialized to the $Maximum\ Conceptual\ Block$s "DriverRecord", "CommunityPoliceRecord", "MotorwayPoliceRecord", "Hospital-Record", and "NewsAgencyReport" respectively. D_1's uplink distributors are two providers (i.e., CP and MP) as a provider's $Maximum\ Conceptual\ Block$ will always contain any distributor's $Maximum\ Conceptual\ Block$. Similarly, D_1's downlink distributors are initialized to two distributors (i.e., D_2 and D_3) as their $Maximum\ Conceptual\ Block$s are contained by D_1's one. DHTs of other distributors are initialized similarly. □

6.2 Communication Protocols for Document Exchanges in a *DocWF*

This section describes the communication protocols required to coordinate peers and infrastructure nodes based on the Publish/Subscribe scheme. The scheme elaborates on the roles of the infrastructure actors and mechanisms for (un)subscription, publication and selective routing and delivery (see Figure 5.11). The protocol description makes use of the annotation element *content signature* of Section 4 and two functions, $served\_list(d)$ and $auth\_list(u)$. Let us recall that a *content signature* is comprised of XML nodes' structural and conceptual information. The function $served\_list(d)$ returns the set of *concept*s that are to be disseminated by a distributor d as represented by the $Maximum\ Conceptual\ Block$ in d's DHT. The function $auth\_list(u)$ returns a set of *content signature* which is used by a subscriber u as a means to verify the satisfaction of security properties over the received XML content.

6.2.1 Subscription of *Concept*s by a User

The protocols start with a user subscription.

1. **Subscribe *concept*s:** User u sends a subscription request (1) (together with its credentials) for a *concept* C_i to a distributor D_r. Upon receipt of such a request, D_r determines the authorizations of the user u, ($auth\_list(u)$) based on the publishers policy as described in Section 5 (2).

2. **Register a user:** If all authorized *concept*s of $auth\_list(u)$ are contained in the list of served *concept*s of D_r, (i.e., $auth\_list(u) \in served\_list(D_r)$), then D_r successfully registers the user u as an authorized subscriber by sending the associated *content signatures* (3) and the subscription protocol ends. A successful subscription of a user triggers an independent secret key (associated with the *concept*) computation by the user (detailed in Chapter 6) using which it can encrypt/decrypt a document.

3. **Forward subscription request:** Otherwise *concept*s of step 1 include at least one *concept*, $C_k \in auth\_list(u)$ such that $C_k \notin served\_list(D_r)$. If C_k contains D_r's served *concept*s, i.e., $C_k \preceq \forall C_i \in served\_list(D_r)$, then D_r sends the request to the uplink distributors. Otherwise, D_r sends the request to the downlink distributors.

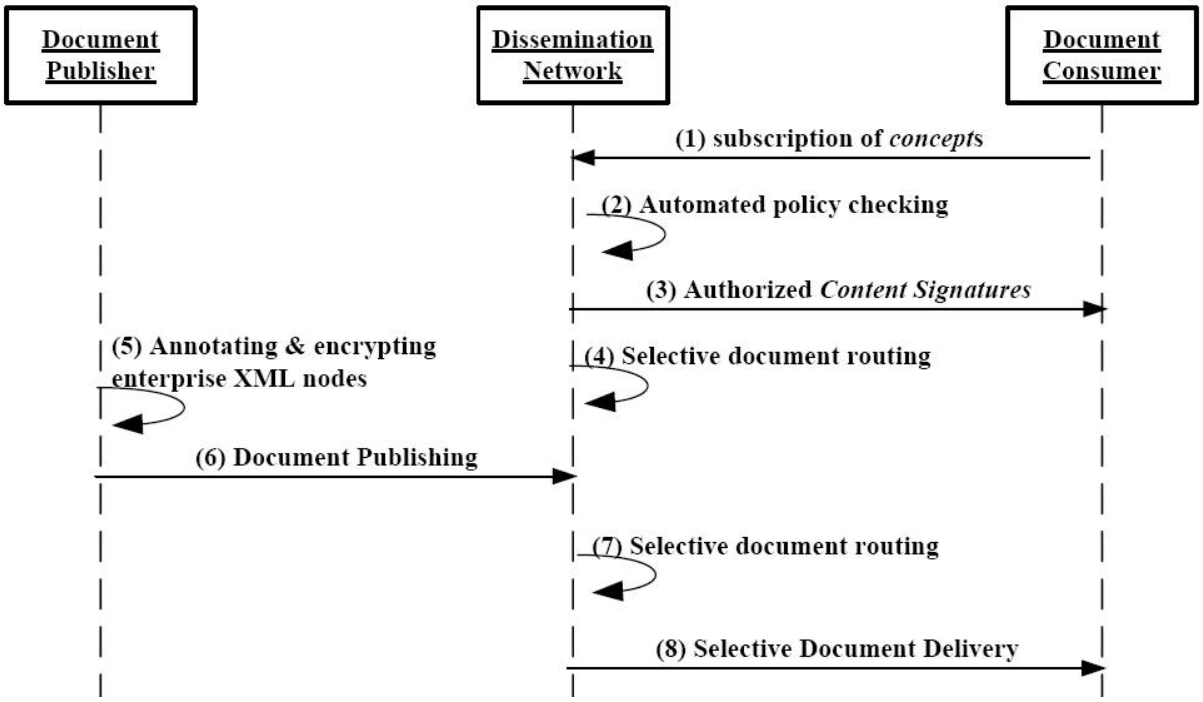


Figure 5.11: Publish/Subscribe-based fine grained document exchanges of peers and dissemination network of distributors (The numbers are only for reference in the description of the protocol).

4. **Route *content signatures*:** After receiving a request for C_k from D_r, a distributor D_m determines if it hosts the content by checking if there exists either a $C_k \in served\_list(D_m)$ of step 3 or a $concept\ containment$ relation $(C_m \in served\_list(D_m)) \preceq C_k$. If so, D_m returns the associated *content signatures* of C_k with success as a response to D_r, else D_m recursively forwards the request as in step 3 to other distributors in its DHT.

5. **Update and register:** After receiving the responses possibly from several distributors, D_r selects a sending distributor using a selection policy described below, updates its list of served *content signature* by adding the new one and notifies other distributors in its DHT accordingly. Now, the distributor D_r is able to register the user u and sends a response by sending the *content signatures* to it.

6. **Deny registration:** In case $auth\_list(u)$ of step one is null, this means that the user u is not authorized to receive XML content associated to the *concept*s C_i and its contained *concept*s.

Steps three, four and five together realize selective document routing (4) upon a successful subscription. Similarly, the publication (6) of a document after its annotation and encryption (5) also trigger selective document routing (7) followed by the actual document delivery to the subscribers (8). These are described in the next sections.

Selection policy: A simple selection policy would be to consider the first response from a distributor as the sending distributor. However, it does not consider any routing optimization for instance. In that context, a selection policy based on a notion of the 'concept distance' aiming at minimizing the hops

required to route the XML content is as follows. Let C_i, C_j be two *concept*s identified by $O_1.C_i, O_2.C_j$, where $O_{i\in[1,2]}$ are two path expressions and $|O_i|$ denotes the number of hops required as entailed by $concept\ containment$. Then concept distance between C_i and C_j is defined as $||O_i| - |O_j||$. The receiving distributor chooses the sending distributor with the smallest 'concept distance' from itself.

6.2.2 Publishing Documents by Document Providers

Publishers take charge of individual XML document data models and their policies associated to the models. They also define a mapping relation of the business domain ontology *concept*s into their individual data model as described in Chapter 4 and shown in Figure 5.9. For a new instance of a document, a publisher annotates and encrypts the mapped document portions (5) as described in Section 4 and finally sends those to its downlink distributors (6). The algorithm of Figure 4.7 is extended in Figure 5.12 to describe this process empirically.

Algorithm: Annotation and encryption of enterprise XML

1. **Input:** $\mathcal{C}$, a collection of *concept*s; a set of documents identified by $\{doc_{id}\}$.
2. **Output:** Encrypted and annotated document.
3. Let $B \in \mathbb{N}$ be an integer for BOL, $l \in \mathbb{N}$ be the depth level of a node used as a delimiter, Q be a FIFO queue.
4. FOR all documents $\{doc_{id}\}$ do
   (a) **Initialize:** set $B = 1; l = 0; Q[0] = l; Q[1] =$ root node of doc_{id} extracted by event-based parser.
   (b) WHILE Q contains XML node do
      i. Let x be the current front value in the Q.
      ii. IF x is a depth level delimiter then
      set $l = l + 1$; Add l into the rear of Q.
      iii. ELSE
         A. **BOL Generation:**
         POP x from Q; Associate (B, l) to x, $B_x = (B, l)$.
         Extract children nodes of x using a tree-based parser and add them to the rear of Q.
         B. **EBOL Computation:** Compute $(f_e(f_{order}(B_x), f_e(f_{level}(B_x))))$.
         C. **Determining Mapping:** $\partial(f, metric)$.
         D. **Document Annotation:**
         Determine node identifier of x as $N_x = (doc_{id}, E_x, E^x_{Highest})$.
         Determine *content signature* of x as $C^x_i = (N_x, C_i)$.
         Compute node integrity of x as $I_x = H(N_x, Ct_x)$.
         Encode node x as $CE_x = (C^x_i, I_x)$.
         E. **Document Encryption:**
         Encrypt the document node as $E^x_p(CE_x, C^z_i)$; where C^z_i is the content signature of the parent of x.
         Generate encrypted and annotated content as (C^x_i, E^x_p).

Figure 5.12: Annotation and encryption of enterprise XML by a document provider before publication.

Instead of a BOL generation an EBOL is computed for each node. In effect, lines of 4.b.iii.A are

actually performed as part of EBOL computation of the line 4.b.iii.B in Figure 5.12. The lines 4.b.iii.C and 4.b.iii.D are similar to the basic annotation algorithm of Figure 4.7 of Chapter 4 except mapping and annotations are performed for encrypted document nodes. Finally, in line 4.b.iii.E each node is encrypted.

6.2.3 Selective Document Routing Between Distributors

The end result of a publishing is that the published annotated and encrypted document nodes are selectively routed to distributors who store and can route the nodes further to other distributors or deliver those to the authorized subscribers. For such selective routing of annotated and encrypted XML nodes, a distributor D_r filters stored XML content by matching conceptual annotations of the received content with its DHT's $Maximum\ Conceptual\ Block$ assignment. This consists in a three step process as described below.

1. **Determine served content:** Upon receipt of annotated and encrypted XML content associated to a set of *concept*s (e.g., C_i), D_r first determines if some or all content are already added to its $served\_list(D_r)$, i.e., $\forall C_i \in served\_list(D_r)$. If some are not added then it determines the new content that it can serve now from the rest by matching conceptual annotations of the content with the *concept*s of its assigned $Maximum\ Conceptual\ Block$. The determined XML content are then added to its $served\_list(D_r)$.

2. **Filter content:** D_r separates XML content which is not its dissemination responsibility and thus is not added in step 1 (i.e., associated to the *concept*s not in its $Maximum\ Conceptual\ Block$) in order to be routed further.

3. **Route content:** D_r then sends the XML content of step 2 to either downlink or uplink distributors. The content associated to *concept*s C_k such that $C_k \preceq C_i$ are sent to uplink distributors and *concept*s C_j such that $C_i \preceq C_j$ are sent to downlink distributors.

6.2.4 Selective Document Delivery to a Peer

In this section, actual XML content delivery ((8) of Figure 5.11) is described. A document provider may update a document depending on the necessity. For instance, the police authority of the country B (PA) of the cross border scenario has multiple accident reports of the same driver resulting into publishing different versions of the accident report. For fine-grained delivery of an appropriate document (e.g., most up-to-date or previous) a distributor can apply document comparing techniques of Chapter 4 in addition to node filtering activities. Moreover, a distributor that does not possess the XML content associated with the *concept*s that peers subscribed needs to fetch the relevant XML nodes. Given that, the delivery of fine grained XML content by a distributor, D_r, occurs after either a publication of a document or a successful subscription as described in the following.

Immediate delivery after publishing: For each subscribed user u, D_r performs the following steps in sequence (Figure 5.13).

1. **Separate allowed *concept*s:** Find the authorized *concept*s of the user u for which D_r hosts the published documents (i.e., $\forall\, C_i \in auth\_list(u)$ such that $C_i \in served\_list(D_r)$).

2. **Determine possible nodes:** Match *concept*s of step 1 with annotations of the encrypted content.

3. **Determine allowed nodes:** Determine appropriate encrypted nodes by comparing nodes of step 2 (i.e., apply the MCES algorithm on a stored document tree).

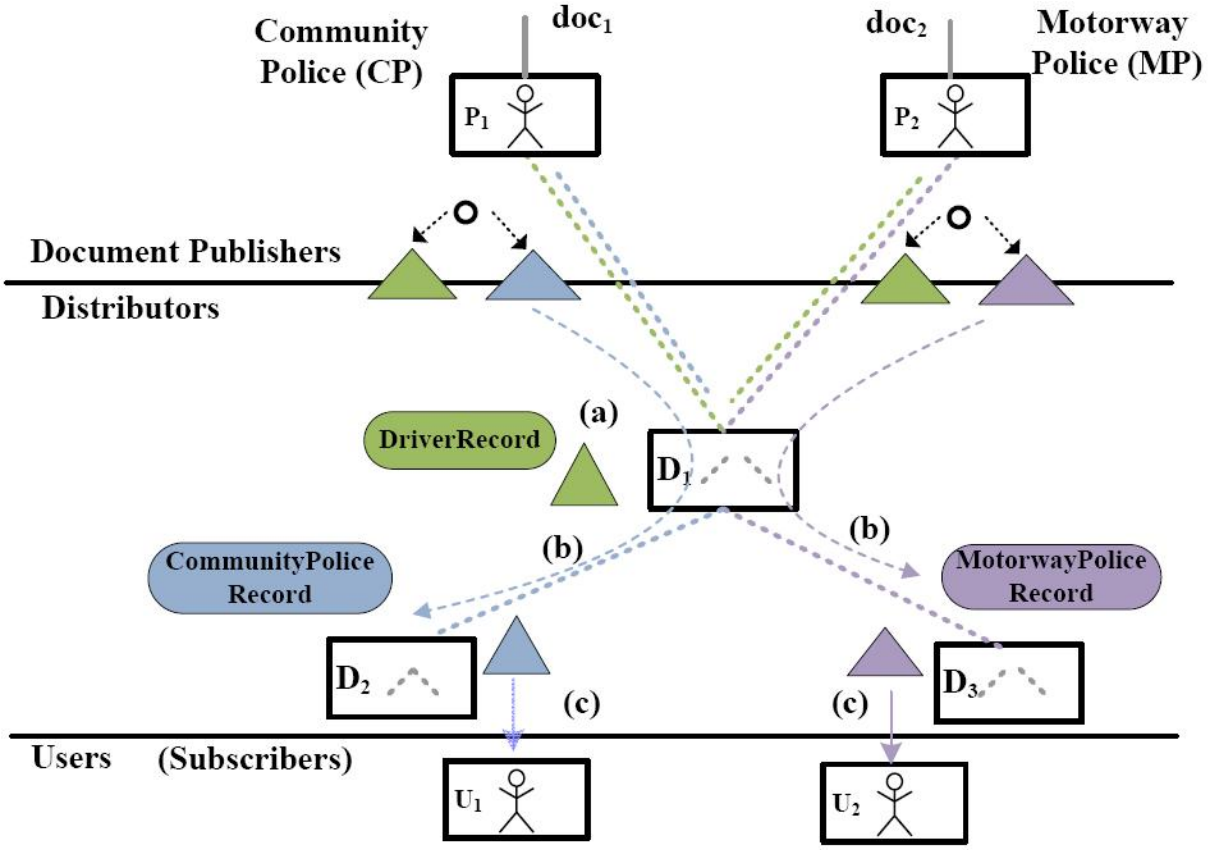


Figure 5.13: Selective document routing by the distributor D_1 to the distributors D_2 and D_3. Immediate delivery of the *concept*s, $CommunityPoliceRecord$, $MotorwayPolice\ Record$ by the distributors D_2 and D_3 to users U_1 and U_2 respectively.

4. **Extract content:** Extract associated encrypted and encoded XML nodes (i.e., $[C_i^x, E_p^x]$).

5. **Deliver content:** Finally, send the XML nodes extracted in step 4 to u.

In Figure 5.13, CP and MP both publish their document portions associated to the *concept* $Driver\text{-}Record$, of which only one copy is stored in the distributor D_1 (a). Published document portions associated to the *concept*s $CommunityPolicRecord$ and $MotorwayPoliceRecord$ are filtered in D_1 (b) and sent to D_2 and D_3 respectively according to their $Maximum\ Conceptual\ Block$ assignment. Distributors D_2 and D_3 immediately deliver (c) the document portions associated to the *concept*s $CommunityPolicRecord$ and $MotorwayPoliceRecord$ to the authorized subscriber U_1 and U_2 respectively.

Catch-up delivery after subscription: If a subscribed *concept*, C_k, of a user u is part of $served\_list(D_r)$ then D_r executes the same operations for delivery after publishing as it already possesses the respective content. Otherwise the subscribed *concept* C_k, is not in its $served\_list(D_r)$. This means D_r needs to fetch the corresponding content from other distributors. Now, D_r can fetch the XML content directly from the distributor D_k selected by the selection policy during subscription by requesting desired *concept*s. In case D_k does not host the content, a fetching protocol similar to the steps 3-5 of subscription (Section 6.2.1) is performed (Figure 5.14). The steps in the fetching protocol differ from those of the subscription in that the distributors fetch the actual content and updates their DHTs whereas using the subscription protocol distributors only retrieve the *content signatures*. This is described below:

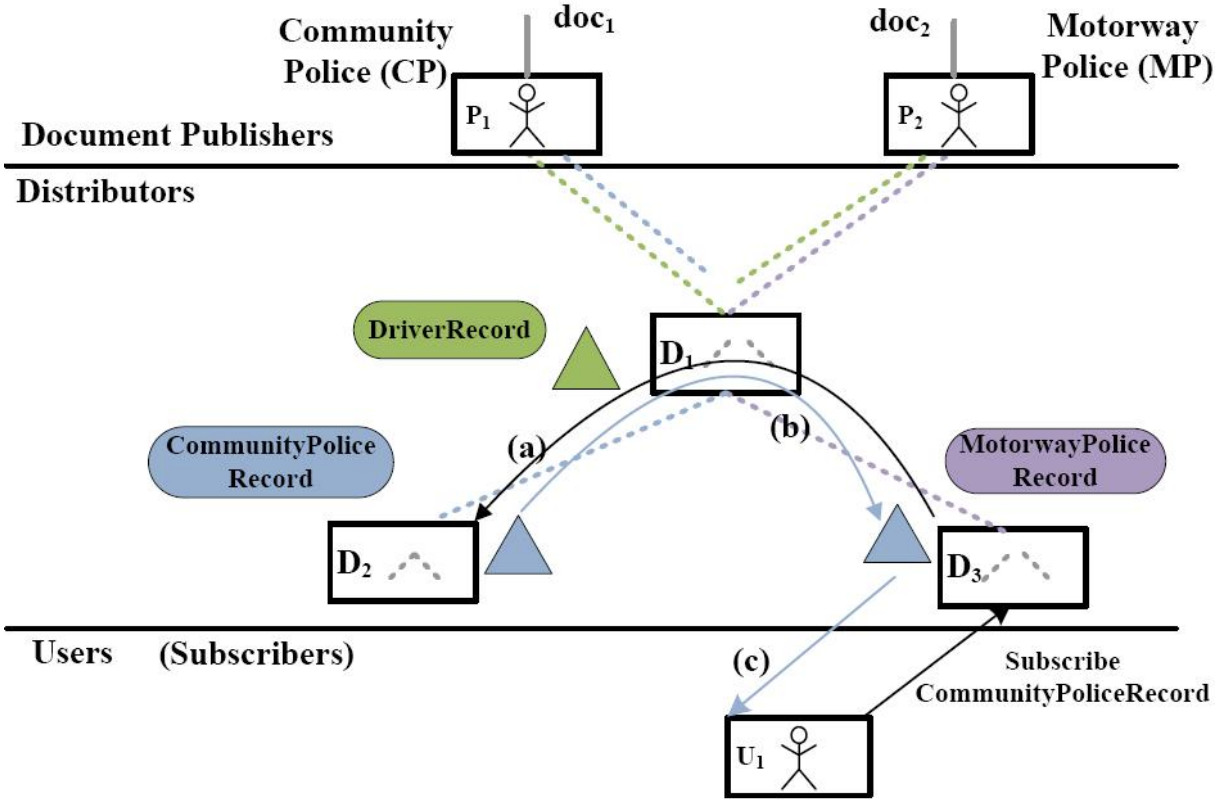


Figure 5.14: Selective document routing by the distributor D_1 to the distributors D_2 and D_3. Catchup document delivery of the *concept*, $CommunityPoliceRecord$ by the distributor D_3 to the user U_1.

1. **Forward subscription request:** If subscribed *concept*, C_k, contains D_r's served *concept*s, i.e., $C_k \preceq \forall C_i \in served\_list(D_r)$, then D_r sends the request to its uplink distributors. Otherwise, D_r sends the request to its downlink distributors (Figure 5.14 (a)).

2. **Update DHT and route content:** After receiving a request for C_k from D_r, a distributor D_m determines whether it hosts the associated content by checking if there exists either a $C_k \in served\_list(D_m)$ or a *concept containment* relation $(C_m \in served\_list(D_m)) \preceq C_k$. If so, D_m adds D_r's reference as the value field in its DHT for the *concept*, C_k, and returns the mapped annotated and encrypted XML nodes of C_k as a successful response to D_r (Figure 5.14 (b)), else D_m recursively performs the same step 1 for other distributors in its DHT. The update in its DHT allows D_m to route XML content associated to the *concept*, C_k, to D_r that are published later.

3. **Update and deliver:** After receiving the first response from a distributor, the distributor D_r updates its $served\_list(D_r)$ by adding the newly received content and adds the corresponding distributor reference as the value field of its DHT for the *concept*, C_k. Now, the distributor D_r is able to deliver the subscribed content to the user u (Figure 5.14 (c)).

In Figure 5.14, CP and MP both publish individual document portions associated to the *concept*, $DriverRecord$, of which only one copy is stored in the distributor D_1. Published document portions associated to the *concept*s, $CommunityPolicRecord$ and $MotorwayPoliceRecord$ are filtered in D_1 and sent to D_2 and D_3 respectively according to their $Maximum\ Conceptual\ Block$ assignment. D_3 does not host the associated document portions of the *concept*, $CommunityPoliceRecord$, subscribed by the user U_1. D_3 then fetches those from D_2 through D_1 which in turn updates its DHT by adding the *concept*, $MotorwayPoliceRecord$ and the reference of D_3 for later routing. D_3 then delivers the content to U_1.

6.2.5 Unsubscription of a Peer

As mentioned before, to protect from any disclosure of sensitive information to an adversary, peers encrypt document portions with a key associated with the *concept*. We rely on a distributed key agreement scheme that is required to be executed by a group of subscribers in a subscription phase in order to compute the key (c.f. Chapter 6). An unsubscription may occur either due to a change of a document provider policy resulting into a prohibition for a peer or in case of a peer request. In both cases, distributors need to update their subscription list. While a new secret key should be computed by a group of subscribers in the event of a new subscription for the same ontology *concept* (detailed in Chapter 6), the existing secret key can be used in case of the unsubscription of an existing peer of the same group. This is because for a successful unsubscription the responsible distributor simply stops delivering the associated XML content to that user. For an unsubscription of a *concept*, C_i, of a peer u, the distributor D_r performs the following steps:

1. **Determine content:** D_r determines the authorized XML content based on the authorizations of u for the *concept*, C_i.

2. **Unregister and stop delivery:** D_r sends a successful unsubscription response to u and stops sending encoded and encrypted XML content of step 1 to u. Then D_r checks whether any other authorized peer has currently subscribed for the same *concept* C_i. If no then D_r also forwards the unsubscription request for the *concept* C_i to other distributors in its DHT in order to prevent future document routing for the *concept*, C_i. However, if at least one legitimate subscribed peer exists for the same *concept* D_r does not forward any request so as to allow future document delivery to existing subscribers.

3. **Unregister and stop routing:** Upon receipt of an unsubscription request for an ontology *concept*, C_i, from another distributor, (i.e., D_r), D_i sends a response back to D_r stating that unsubscription is successful and stops routing annotated and encrypted XML content associated to C_i to D_r. D_i further checks whether any other authorized peer or distributor has currently subscribed or requested for the same *concept*, C_i. If no, then it performs similar steps as in step 2.

7 Subscriber-end Document Processing for a *DocWF* Execution

Upon receipt of various annotated and encrypted XML nodes, an authorized subscriber decrypts and then decodes those to detect precise integrity violation, for instance, any node deletion, node swapping, order change, node update (c.f. Chapter 6). The subscriber then processes this content depending on its role in the dissemination. For instance, the role of the local court (LC) is to find facts and evidences in order to give a verdict. In order to do so, it subscribes to *concept*s "MotorwayPoliceRecord", "CommunityPoliceRecord" and "NewsAgencyReport" to receive published content of MP, CP and NA respectively as document portions associated with these *concept*s would contain facts and evidences. LC may also publish its verdict later on. In the following, we illustrate the process on such scenarios.

7.1 Sharing Partial XML Schema Model

A publisher and a subscriber may share the same XML schema model, for instance, CP and MP departments have a bilateral agreement to have the same partial schema for a driver record (see Figure 5.9). As such, receiving multiple instances of a driver record with different value for the `id` attribute from CP represent multiple driver information.

7.2 Sharing Partial Schema Mapping

This scenario is similar to the previous one except that peers share only the mappings related to published document portions. In particular, a publisher also publishes the associated mapping from a *concept* to its schema elements as part of the annotations of an XML node so as to be accessible by the authorized subscriber only. Such a mapping of each node of a document portion accompanied by annotations provides a clear understanding of the semantics and structure of those received document portions.

7.3 Applying Customized Rules

We illustrate this scenario in Figure 5.15. Let LC be an authorized subscriber of the content associated with the *concept*s, $DriverRecord$, $CommunityPoliceRecord$, $MotorwayPoliceRecord$ and $NewsAgencyReport$. Considering no integrity violation occurs, LC applies the following general steps:

1. *Document portion identification rule*: is the initial step performed over the annotations of document nodes to identify the desired nodes for further processing. The rule is: determine all nodes annotated with desired *concept*s and possibly filter those further by separating nodes for desired publishers.
2. *Business rules*: is a set of rules applied over the document nodes from step 1 in order to perform an application specific processing.

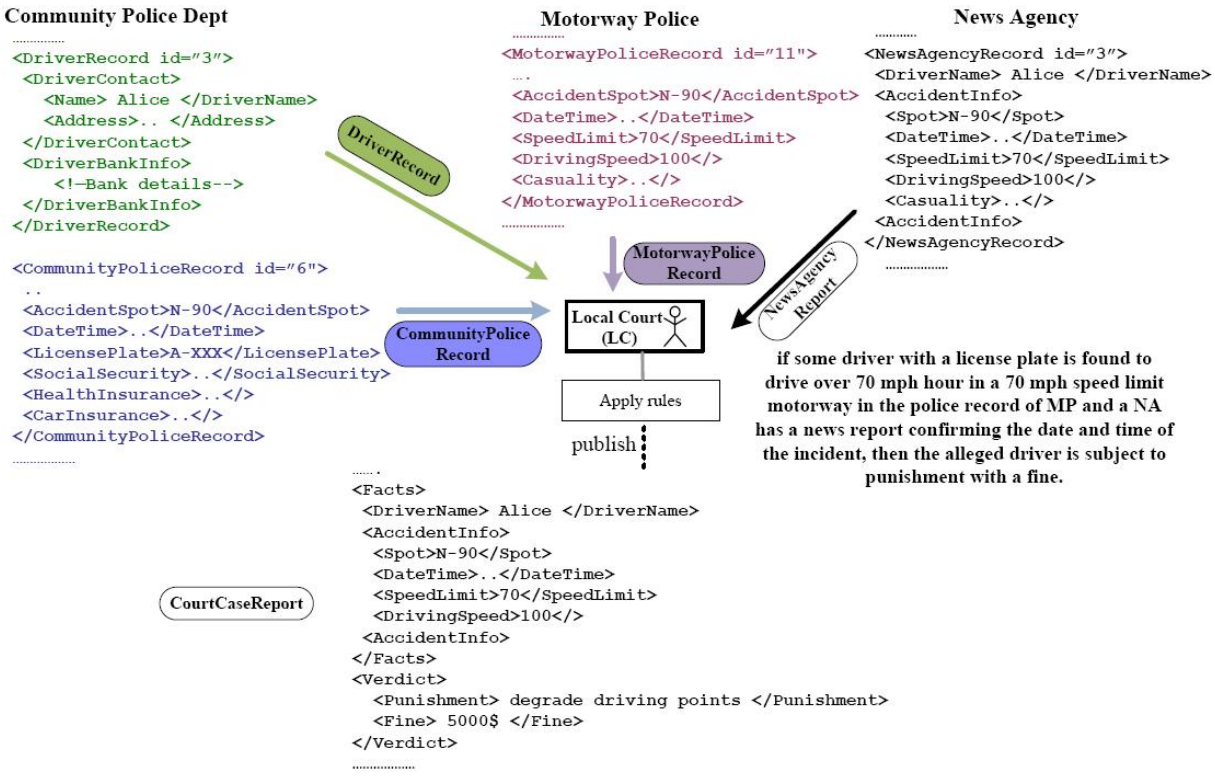


Figure 5.15: Applying customizing rules in order to process various document portions.

LC identifies document portions associated to the *concept*s, $DriverRecord$, $CommunityPoliceRecord$, $MotorwayPoliceRecord$ and $NewsAgencyReport$. Further, it separates the `<DriverRecord>` and `<CommunityPoliceRecord>` of CP, `<MotorwayPoliceRecord>` of MP and `<NewsAgencyRecord>` of NA. LC has two business rules for: document correlation and document composition where the first determines the related document instances and the other builds LC's own document portion associated to the *concept*, $CourtCaseReport$, that it may publish. In particular, document correlation corresponds here to finding a driver name having the same license plate no in some CP and MP records that states a driving speed $>$70 mph in the motorway N-90 on the same date and time. This can leverage the MCES algorithm of Chapter 4 to compare two document portions of CP and MP for instance. If the NA report also corresponds to the same driver then LC considers those as facts and gives its verdict: degrading driving points and a 500 USD fine. A simplified document composition rule: build a document with `<Facts>` and `<Verdict>` accordingly with consolidated facts and verdict respectively.

If none of the mentioned scenarios applies, the subscriber-end processing looses precise semantics of different XML vocabularies (for instance, a subscriber cannot interpret the received XML tags) despite each received document portion is annotated with a *concept*. In such cases, schema matching solutions [ABM05, FZT04, RB01] can be used for mostly structural matching given that full schemas of all parties are known (i.e., against our motivation). As such, we suggest a technique to build up a customized recipient schema based on shared ontology that can be used for further processing (detailed in [RPRS09]).

8 Related Work

The development of communication infrastructures to support distributed workflow execution has been an active research area for a long time [AAM97, EP99, AMAA97] for complex cross-organizational processes. With the document-based agile workflows, we propose to go one step beyond by making it possible to handle agile business scenarios whose peculiar nature, for instance, no a priori known peers and business tasks do not allow neither a centralized nor a pre-planned workflow execution. The areas of research tackled by the work presented in this chapter include:

- Workflow executions in a centralized environment.
- Workflow executions in a decentralized environment.
- Publish/Subscribe-based distributed execution of a workflow.

The related work discussion of this chapter is based on these three topics.

8.1 Centralized Communication Infrastructure for Workflow Execution

An exhaustive survey of centralized workflow management systems can be found in [AMAA97, AS96]. All of them rely on a central coordination engine for workflow message dispatching amongst known peers. This centralized engine becomes a bottleneck in cross organizational workflows like agile business processes that require message dispatching amongst a priori unknown peers beyond business boundaries.

8.2 Decentralized Communication Infrastructure for Workflow Executions

Various contribution of work support distributed executions of workflows. They design communication protocols for transferring workflow data between peers which in contrast are implemented as document exchange protocols between peers in this chapter. They all lack flexibility in the loosely coupled data exchanges between a priori unknown peers which is a critical issue to meeting the requirements of agile environments. These are described in the following.

8.2.1 Decentralized Execution by Known Peers

The distributed architectures developed in [AAM+95, GAC+97] focus on the execution of workflows between a priori identified peers rather than enabling the execution in an agile environment with a priori unknown peers. The authors in [MM05] proposed a workflow architecture for pervasive environments wherein suitable business partners are discovered at runtime to perform pre-defined business activities in

a pre-planned execution sequence. It proposes a fully distributed execution by assigning tasks to suitable peers and thus can be a complementary with our approach in terms of increased dynamicity. In particular, this approach can be used in a *DocWF* execution in assigning suitable peers for executing determined business tasks.

8.2.2 Decentralized Execution by Message Passing

In order to exchange messages between peers beyond boundaries, architecture specific message structures are devised in several pieces of work [BMR94, MM05]. As such peers are limited to an architecture specific message structure for every other business domain when sending messages and they also lose interoperability. In contrast, in our case as long as an ontology describing a business domain is specified peers can build any document structure with varying vocabularies given that the documents are annotated with semantic descriptions.

8.2.3 Decentralized Execution by Mobile Agents

Agent-based solutions [VBS04, Wan00] can also be used for the execution of distributed workflows. Mobile agents, being active software components as opposed to documents, travel by themselves between known peer sites to execute precise tasks in a workflow. In contrast, document exchanges in a *DocWF* are at peers' discretion, thus peers can control which document to send and which to receive. Agent-based approach requires security solutions to protect peer applications from malicious mobile code as opposed to protecting passive components like documents in our *DocWF* (c.f. Chapter 6). Besides mobile agents pose a problem if a host crashes which is not the case for us as data resides in the documents and thus are not lost.

8.3 Publish/Subscribe-based Decentralized Communication

In [EP99], authors proposed an event-based architecture for a distributed execution of workflows wherein actors publish events (e.g., administrative, process, exception) which are consumed by interested peers in a workflow. For grid computing based web services, the work of [SHM08] developed a Publish/Subscribe-based distributed workflow management system leveraging WS-Notification [WSN] framework. All these work focus on only loosely coupled data exchanges irrespective of data models leading to frequent disruptions whenever there is a change in a peer data model. A combination of a semantic level data modeling with Publish/Subscribe-based data exchanges, for instance, in [hZyLL05] and as pursued in this thesis is a prerequisite for *DocWF* applications.

9 Conclusion

This chapter describes an ontology-driven decentralized communication infrastructure to support *DocWF* executions. Its deployment can be supported in a scalable way based on a Publish/Subscribe infrastructure. It allows a peer to send annotated (e.g., domain *concept*, structure and security meta data) documents containing various extensible vocabularies to the potential executor of one of the next goals in a *DocWF* in a loosely coupled manner. While authorization policies for published documents are specified over business domain *concept*s their enforcement relies on encryption. The use of ontologies allows peers to exchange fine-grained documents in a loosely coupled fashion by leveraging a family of document exchange protocols but also enables a distributor to check policies of the document providers by reasoning over domain and policy ontologies. The employed dissemination network of distributors takes charge of dispatching relevant documents in a fine grained fashion to interested peers and thus separates the concern of loosely coupled data exchanges from that of the decentralized execution of a *DocWF*. Such a separation allows actors to handle documents independently as illustrated in different scenarios, for instance, building a composite document out of multiple received documents.

The work of this chapter has been published in the proceedings of The Fifth International Conference on Information Assurance and Security (IAS 09), August 18-20, 2009, X'ian, China [RRS09c] and IFIP SEC 2009, 24th International Information Security Conference, May 18-20, 2009, Pafos, Cyprus [RRMS09] and in a special issue of Journal of Information Assurance and Security (JIAS), volume 6, http://www.mirlabs.org/jias, [RRS].

Chapter 6

Security of Document-based Agile Workflows

Security is the chief enemy of mortals.
- William Shakespeare -

1 Introduction

In this thesis so far, we described the document-based agile workflow (*DocWF*) models and their runtime enactment, interoperability solution, and ontology-driven decentralized communication infrastructure of a *DocWF* application. For runtime enactment of a *DocWF* model, peers beyond business boundaries can proactively determine suitable business tasks and their binding to IT components by leveraging semantically enriched documents. To support a distributed execution of these tasks by a priori unknown peers, such documents are then distributed selectively through the communication infrastructure in a loosely coupled fashion. Like any complete decentralized system, a *DocWF* execution needs to be supported by appropriate security solutions. As opposed to typical distributed workflow management systems where task authorization is the main security issue, a *DocWF* execution raises new security concerns and challenges with respect to documents and the communication infrastructure. Document security concerns include, for instance, the integrity protection of fine grained documents, the enforcement of authorizations over fine grained documents, and their a posteriori verification. Security challenges related to the communication infrastructure are raised by key management among others to enable fine-grained document exchanges between a priori unknown peers. Encryption over fine grained documents can be used as an enforcement technique of fine grained document authorization. However, computing a group key associated with an ontology *concept* for peers of various origins while limiting disruptions in a running *DocWF* execution is a challenging task. In addition, tracing back actions of a priori unknown peers on documents

whenever required is also challenging. Existing distributed workflow management systems are mostly devoted to either access right management at the business task level [LZS09, ACM01, CLW05, KPF01] or to guaranteeing execution proofs against a pre-specified series of business tasks [MM07]. Therefore security issues related to documents and communication infrastructure in a *DocWF* setting for agile scenarios are not addressed yet.

In this chapter, we present security solutions to support the secure execution of a *DocWF*. These solutions describe distributed access control enforcement mechanisms for fine grained "Enterprise XML" documents. These mechanisms capitalize on a group-based cryptographic technique called tree-based group Diffie-Hellman (TGDH) [KPT00]. We adapt the TGDH technique to enable a group of peers with the same subscription for an ontology *concept* to compute independently a group key associated with the *concept*. In particular, we aim at restricting document access through the encryption of a document portion with a key computed by a group of peers with similar access rights. This independent key computation allows a legitimate peer to encrypt/decrypt documents autonomously in a fine grained manner and prevents malicious business actors from performing unauthorized access and actions in documents, for instance, illegitimate updating, deleting, inserting and even moving of nodes. The adapted TGDH further limits the scope of rekeying when peers dynamically join or leave in an agile environment so as to enable non-disruptive document exchanges for an extended period of time required for an execution of a *DocWF*. Traceability for an posteriori verification is also enabled thanks to this mechanism combined with special security annotations. The design of the suggested mechanisms is strongly coupled with the communication infrastructure specification which eases their integration into the infrastructure.

The rest of this chapter is organized as follows. Section 2 depicts a motivating scenario and introduces security issues including an attacker model of a *DocWF* application and goes on to discuss related security requirements. In Section 3, the TGDH scheme is briefly described by highlighting its pros and cons for *DocWF*s. Section 4 describes the enforcement mechanisms for fine grained document authorizations for an ontology *concept*. It details the adapted TGDH solution. Section 5 describes the various document protection enforcement techniques. Section 6 introduces special security annotations and their usage for traceability. The security analysis of the proposed mechanisms are presented in Section 7. Section 8 finally compares the solution presented in this chapter with related work, followed by a conclusion.

2 Problem Statement and Security Requirements

Digital documents are an increasingly central concern in today's inter organizational collaborative processes, as illustrated by the multiplication of XML standards for instance. As mentioned in Chapter 1, in many cases, such documents are "Enterprise XML" where multiple authorities are in charge of their own portion of the document and of ruling who may edit or read fine grained parts of the documents. The document edition process has followed a similar increase in complexity in that it is getting more and more collaborative. Peers join and leave groups of collaborating hosts depending on context evolution and on their mobility or churn rate. In what follows, a collaborative document exchange scenario, a review of issues regarding the integration of the interoperability and communication infrastructure solutions into the scenario, an attacker model and finally the security requirements of a *DocWF* application.

2.1 A Cross Border Crime Scenario

This section illustrates a document exchange based collaboration of a EU IST research project called "R4eGov" [R4E]. Document exchanges take place between two European Union (EU) administrative bodies, Europol and Eurojust, and the law enforcement authorities of 27 associated member states whenever there is an occurrence of cross border organized crime [BN]. Europol and Eurojust have representatives called Europol National Members and Eurojust National Members respectively, for each member state. Each member state has its national contact point (National Authority) for Europol and Eurojust. The collaboration starts with a request called Mutual Legal Assistance (MLA) from a concerned authority. Peers define and work on their document portions resulting into a composite document called the European Arrest Warrant (EAW) as follows.

1. A Europol National Unit of country A, ENU A, makes a written request of assistance (for a witness protection) to a Eurojust National Member of country A, EJNM A.

2. The EJNM A opens a Temporary Work File (Twf) in a local Case Management System (CMS).

3. The EJNM A contacts the Eurojust National Member of country B, EJNM B, by forwarding the request of assistance.

4. The EJNM B contacts the responsible National Authority of country B, NA B. Steps are taken by the responsible NA B to provide the requested assistance.

2.2 Integrating Interoperability and Communication Infrastructure

To justify the security requirements of a *DocWF*, this section first illustrates the integration of the mechanisms described in Chapters 4 and 5 for the cross border crime scenario.

2.2.1 Applying Semantic Enabled Document Modeling and Comparison

By leveraging a publicly shared cross-border crime ontology describing *concept*s such as $WitnessRecord$, $MutualLegalRequest$, $Case$ etc. peers can apply the semantic enabled document modeling techniques of Chapter 4. Due to distinct laws and regulations, peers (i.e., ENU A, EJNM A, EJNM B, and NA B) have their individual sensitive data model. Figure 6.1 illustrates four simplified mapped document portions of the composite EAW document originating from respective data model (distinguishes the origins (i.e., publishers) of different parts by dotted rectangles around subtrees). Peers also integrate document comparison techniques in order to determine the latest MLA request. For instance, when receiving multiple MLA requests from ENU A, EJNM A distinguishes the requests by comparing the encrypted nodes `<RequestID>` and determines the latest MLA by comparing the decrypted nodes `<RequestTime>`.

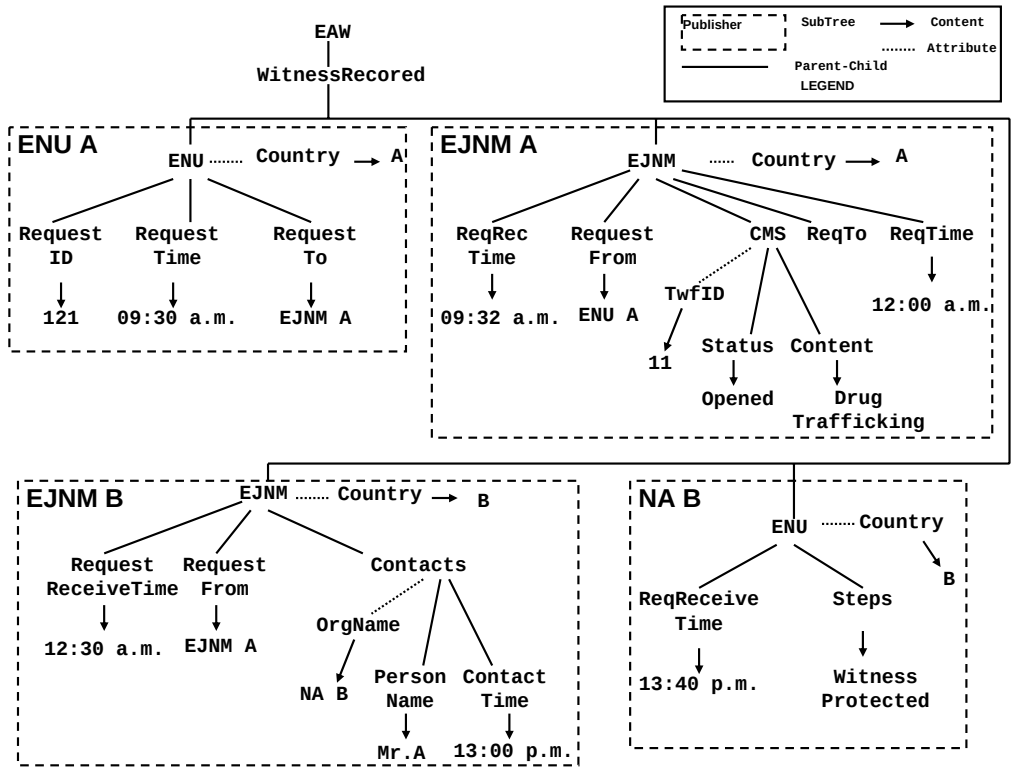


Figure 6.1: Four simplified document part instances rooted at <ENU A>, <EJNM A>, <EJNM B> and <NA B> of the EAW document.

2.2.2 Leveraging the *DocWF* Communication Infrastructure

The cross border crime scenario features the following in terms of document exchanges.

1. **Distributed Document Sources:** No central repository is available. Documents are instantiated by concerned peers depending on local requirements and regulations as opposed to being instantiated by default by a centralized source. For example, one country may record the religious belief of a suspect which may be prohibited by law in another country resulting in using different document instances and different authorization policies over those instances.

2. **Document Distribution:** Peers exchange documents autonomously. Document editors send new or updated documents to other peers proactively. An ENU A employee, for instance, sends a MLA request to an EJNM A employee (Figure 6.1).

3. **Fine Grained Document Access:** Access patterns on documents may range from the whole document to an individual element as can be specified by ontology *concept*s which can then be mapped to individual document portions. NA B may need to access a deeply nested element <PersonName> containing a suspect's name originated by the manager of EJNM B (Figure 6.1).

4. **Distributed Document Access:** One peer may need to access document parts originated from other peers, even if the originator is unavailable. The provider of <EJNM B> may not be available forcing NA B to get the element <PersonName> information from some other participants of the workflow if possible (Figure 6.1).

Surely, the above features can not be meet by a centralized communication infrastructure. Collaboration in such a document-based approach can then leverage the communication infrastructure of Chapter 5 for exchanging fine grained documents. In Figure 6.1, ENU A is the publisher (i.e., owner) of the subtree rooted at <ENU>. Similarly EJNM A, EJNM B and NA B are the publishers of the subtrees rooted at nodes <EJNM A>, <EJNM B> and <NA B> respectively. Then EJNM A subscribes the *concept*, $MutualLegalRequest$, without knowing a priori whoever needs such information. When ENU A publishes the content of the subtree <ENU A> (representing the initial MLA request) the communication infrastructure delivers <ENU A> to the EJNM A. Receipt upon such content EJNM A opens a Twf (i.e., <TwfID>) in its *CMS* and processes further according to EJNM A business rules as described in Chapter 3. EJNM A may further publishes the subtree <CMS> for potential subscribers of <Status>. When using such a decentralized infrastructure as described in Chapter 5, however, one must take into account more adversary activities as described in the next section in order to ensure a secure *DocWF* execution.

2.3 Attacker Model

Various kinds of adversaries and their activities can target documents and the communication infrastructure to breach the security of a *DocWF*.

1. **Internal and external adversaries:** Peer data models (e.g., XML schemas) describe information sensitive for peer companies and will not be shared. Authorizations will be given to access document content related to particular semantics, as described in Chapters 4 and 5. Thus any internal (i.e., peers, hosts of distributors) or external entity without any authorization to *concept*s is considered to be an adversary.

2. **Malicious distributors:** Distributors are honest in that they host fine grained encrypted XML nodes along with their annotations (i.e., fragmented documents) as described in Chapter 5. Thus a distributor node can read only the semantics (i.e., annotations) of every node it is responsible for dissemination. But they are curious in knowing the content and the structure of original composite documents.

3. **Malicious activities:** Communication channels between a peer and a distributor and between distributors are not secured. So any adversary may get hold of exchanged documents and perform malicious activities, for instance, illegitimate node deletions, node swapping, node order changing, node updates of originally published XML nodes. Malicious distributors may also illegitimately prune encrypted document nodes including the abovementioned malicious activities when routing and delivering those.

2.4 Security Requirements

Considering the attacker model, we specify the following security requirements that need to be addressed for a secure *DocWF* execution. In the rest of the chapter, documents or document portions refer to the annotated and encrypted forms of them unless otherwise stated.

2.4.1 Document Authorization

As opposed to task authorizations, the main security requirements for a decentralized execution of *DocWF*s are specified as follows:

1. **Enforcing Fine Grained Document Authorization for an Ontology *Concept*:** Fine grained documents associated to subscribed ontology *concept*s must not be readable by any adversary but authorized business peers.
2. **Enforcing Authorization Decision Delegation:** Access control decisions over fine grained documents cannot be performed by their providers themselves as actual recipients are determined only at runtime and providers may also be unavailable.

In the cross border crime scenario, multiple peers may subscribe for the same domain ontology *concept* and thus form a group having the same access right. Enforcement of fine grained document authorization for a *concept* is done by encryption as will be described in Section 4. Regarding delegation, providers will know neither future subscribers of a *concept* nor possess mapped document portions (originating from other providers) associated with the *concept*. As such providers can not determine the appropriate document authorizations for future subscribers in advance. Instead, distributors are delegated to do this as described in Chapter 5. Annotations of the XML nodes allow a distributor to be a delegate of the nodes provider. In particular, the actual filtering of XML nodes according to providers policies are performed by the distributors as opposed to the providers themselves.

2.4.2 Document Protection

During a *DocWF* execution, fine grained documents are published to distributors which then deliver those to legitimate peers using the communication protocols of Chapter 5 that leverage the semantic annotations of documents. This raises document security requirements in terms of integrity and authenticity as follows. Upon receipt of XML nodes a peer should be able to verify the following:

1. **Fine Grained Document Authenticity:** It receives the nodes of an original editor.
2. **Semantic Integrity:** The semantics of the received XML content should be protected with respect to the *content signatures* (c.f. Chapter 5) received during subscriptions so as to ensure that received document portions represent expected ontology *concept*s. The semantic verification by a recipient peer is further categorized as follows:

(a) **Authorization Verification (I):** Have all *concept*s been received? To check whether all received XML nodes are associated with the recipient's authorization.

(b) **Pruning Verification (II):** Have all XML nodes from desired documents been received? To check whether XML nodes have been received for all subscribed *concept*s.

(c) **Mapping Verification (III):** Do the document nodes correspond to nodes associated with desired *concept*s? To check whether received XML nodes have appropriate semantic annotations with respect to its *content signature*.

3. **EBOL-based Structural Integrity:** The structure of the received XML nodes should be protected. Based on the EBOL-based encodings of Chapter 4, such structural integrity protection of XML nodes can be further categorized as follows:

(a) **Illegitimate Node Update Verification (I):** Has the node content been changed?

(b) **Illegitimate Node Pruning Verification (II):** Has some XML nodes not been received (i.e., not deleted, illegitimately filtered)?

(c) **Illegitimate Node Move Verification (III):** Have some nodes been moved?

(d) **Illegitimate Node Order Changing Verification (IV):** Has the node order been changed (or swapped)?

4. **Document Containment:** The filtered/pruned XML nodes that a peer receives were originated from a composite document, a stringent document integrity property that we call "document containment".

In *DocWF* applications, peers may publish a portion of a composite document as opposed to publishing a complete document. Checking the document containment of the document portion in the composite might be necessary in some scenarios. For instance, in Figure 6.1 when a peer is authorized only to receive `<Content>` of the document portion rooted at `<CMS>` then the peer might need to know whether the `<Content>` containing information "Drug Trafficking" was contained in the document portion `<CMS>` and thus related to the appropriate `<TwfID>`.

2.4.3 Traceability for a Posteriori Verification

In a decentralized execution of a *DocWF*, candidate peers for the same *concept* authorizations are determined at runtime by the semantic enabled policy checking as described in Chapter 5. The subscription of a peer and thus its involvement in an execution remains anonymous to other peers. In some critical business scenarios however, like for instance the cross border crime one, detecting document originators and thus identifying a peer may be required to resolve any dispute or conflict on any document update. Similarly, tracing back a document editor might also be required. As such depending on scenarios traceability can be one of the following:

1. **Peer Anonymity:** A member peer, P_i, of a group of m a priori unknown peers (thus preventing their identification) with the same *concept* authorization, should not know all cryptographic shares of peers for the group key computation.

2. **Traceability of Document Access:** All access by peers should be tracked if possible to check if previous access performed were authorized. Our proposal addresses update access only, in particular, to ensure that the update of a document part was performed in conformance with the authorization policy of its originator.

2.5 Key Management Challenges

As mentioned in Section 1, an access right to a *concept* leads to the selective delivery of multiple encrypted document portions (from multiple providers) to authorized peers of a group. To decrypt or encrypt updated documents peers need to compute the same key. Key management challenges are regarding the group key computation and its renewal as described below.

2.5.1 Group Key Computation

There can be two general approaches to compute a group key to encrypt/decrypt multiple document portions that originate from multiple providers.

1. **Separate key associated with a *concept*:** A separate key can be computed for each *concept* by each provider to encrypt its mapped document portion. However, this approach may result into additional key management overhead for peers and distributors as multiple keys need to be managed for the same ontology *concept*.
2. **Dedicated key associated with a *concept*:** A dedicated secret key associated with an ontology *concept* can be computed by all peers so as to reduce the above mentioned overhead.

We follow the latter approach in this chapter (Figure 6.2). While key management overhead is reduced, additional coordination by distributors is required during subscription to ensure a group secret key computation (c.f. Section 4).

2.5.2 Group Key Recomputation

Depending on its needs, a peer may (un)subscribe to an ontology *concept* as mentioned in Chapter 5. A successful subscription to a *concept* leads the subscribed peer to join the group associated with the *concept* and an unsubscription to the *concept* leads to a leave from the group. Both joining and leaving require recomputing the group key associated with the *concept* which can be performed in either of the following:

1. **Centralized:** A publisher can compute the group key associated with a *concept* and distribute the key afterwards.

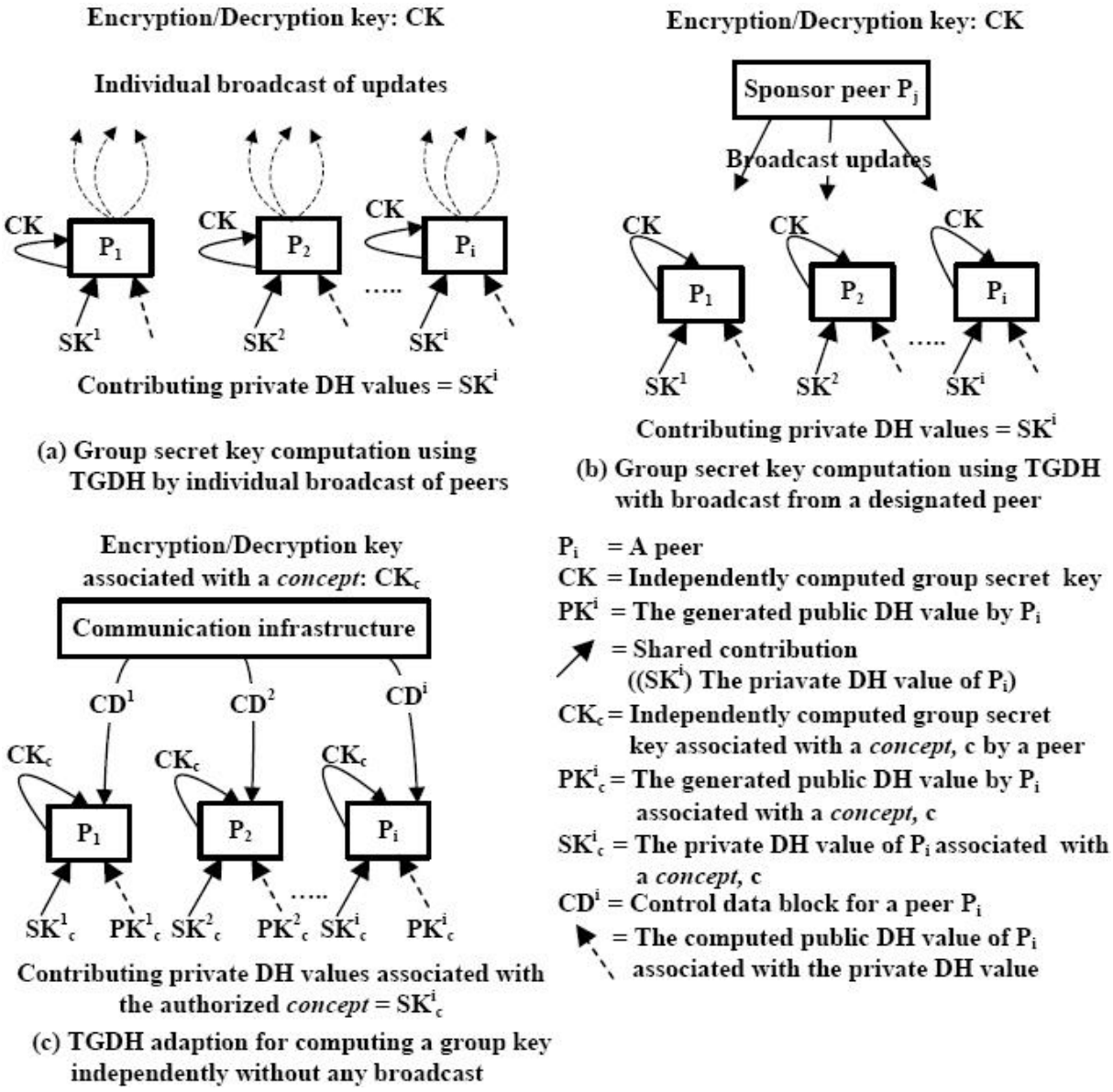


Figure 6.2: Group key computation schemes by a peer.

2. **Distributed:** The publisher and authorized subscribers may compute the group key independently.

Regarding centralized recomputing, multiple publishers may send document portions associated to the same *concept* and multiple subscribers may have the authorizations for the same *concept*. As such, publishers need to synchronize themselves during publication so as to compute the same group key associated with the *concept*. In a *DocWF* application, as future subscribers are not known in advance key distribution is infeasible for a publisher. Moreover, distributing the group key in an insecure communication channel may expose the key to an adversary and thus may compromise document confidentiality and integrity. Similarly to the distributed approach, we developed a group key (re)computation technique based on TGDH [KPT00] that allows publishers and subscribers to compute the group secret key independently without a priori knowledge of each other (Figure 6.2 (c)).

Security solutions presented in this chapter are flexible in the sense that depending on the peculiarity of an agile business scenario one or more security requirements can be enforced as opposed to enforcing all. In particular, the security meta data to enforce semantic integrity, EBOL-based structural integrity, and document access traceability are independent so that one or more can be employed depending on the scenario. The next section gives a brief introduction of the TGDH protocol followed by the enforcement solutions of document authorization (Section 4), document protection (Section 5) and document access traceability (Section 6) of fine grained documents that are exchanged in a *DocWF* execution.

3 TGDH Overview

TGDH [KPT00] is a group based cryptographic protocol operated on a logical binary key tree as depicted in Figure 6.3 where the key tree is a binary search tree. The list of nodes in the path of a peer from the leaf to the root is the key path of the peer. A sibling-path of the peer is the list of sibling nodes of the key-path nodes. This scheme allows a group of peers to compute a group secret without relying on a central authority. It assumes that, when a peer joins or leaves a group, current peers rekey together either synchronously as in the original scheme or semi-synchronously using an interval based rekeying [LLY02] or a non-blocking rekeying [LYGL01].

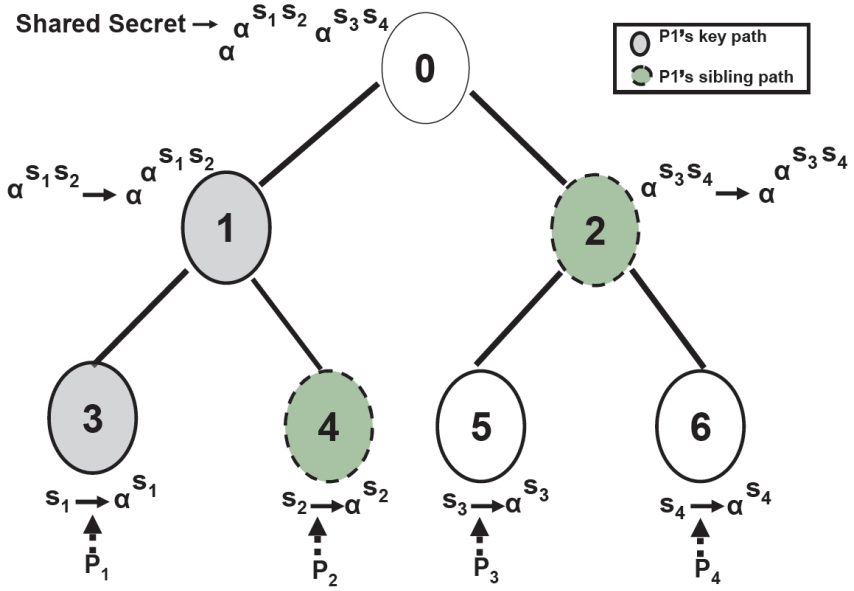


Figure 6.3: A logical key tree, T_k, of 4 peers P_1, P_2, P_3, P_4 in a tree-based Group Diffie-Hellman (TGDH) key computation. Peers host their public DH values in the leaves. The notation $k \rightarrow \alpha^k$ of a node means that a peer computes the DH private value (k) followed by the DH public value $BK = \alpha^k$. P_1's key-path is the nodes 3 and 1 and P_1's sibling-path is the nodes 4 and 2.

3.1 Group Secret Key Computation

Each peer hosts its share in a leaf node in the logical key tree T_k and maintains a set of keys associated to its key-path which are used for generating a group secret share (see Figure 6.3). Every node in the tree is assigned a unique number v, starting with the root node that is assigned 0: the two child nodes of a non-leaf node v are set to $2v+1$, and $2v+2$ respectively. Each node v is associated with a key pair consisting of a secret key K_v (i.e., Diffie-Hellman (DH) private key) and of a blinded key $BK_v = \alpha^{K_v} mod\ p$ (i.e., DH public value). The blinding relies on the hardness of solving the discrete logarithm. For every node v, K_v is computed recursively as follows:

$$K_v = \begin{cases} \text{if v is a non-leaf node;} \\ (BK_{2v+1})^{K_{2v+2}} \text{ mod p; where p is a prime number} \\ = (BK_{2v+2})^{K_{2v+1}} \text{ mod p} \\ = \alpha^{K_{2v+1}K_{2v+2}} \text{ mod p} \\ \text{if v is a leaf node;} \\ SK^i \text{ which is the DH private key of peer } P_i. \end{cases}$$

In short, computing the DH private value K_v of a non-leaf node requires the knowledge of the DH private value of one of the two child nodes and the DH public value of the other child node. In effect, a peer only needs to compute the DH private values along its key-path. In other words, a peer only needs to know the DH public values associated to its sibling-path nodes where the sibling nodes of the key-path nodes comprise a sibling-path. Therefore, the value K_0 computed for the root is the shared secret for all peers. At this point, the group secret key, CK, is derived from the secret as follows:

$$CK = h_1(K_0); h_1 \text{ is a hash function.}$$

3.2 Advantages of TGDH

The group secret computation is distributed in that each peer computes its own key-path values independently of others. It is also contributory as each peer initially provides the public DH value in its leaf node as its share [KPT00]. Furthermore, this scheme has four essential advantages that perfectly suit a *DocWF* application.

1. **Partial knowledge:** A peer, P_i, only needs to know the public DH values associated with its sibling-path to compute the group secret by itself.

2. **Local key-path generation:** A peer, P_i, can compute the group secret without generating the complete key tree (i.e., only the key-path).

3. **Dynamic computation:** The key computation is completely dynamic as the partial knowledge required at step 1 is computed by the current members of a group.

4. **Forward and backward secrecy:** This scheme also exhibits cryptographic properties of Group Key Secrecy, Forward Secrecy (if a new peer joins), Backward Secrecy (if a peer leaves) and Key Independence [KPT00].

While forward and backward secrecy is identified as an advantage we do not further use it for the specification of the adapted TGDH solution.

3.3 Limitations of TGDH for *DocWFs*

The rekeying process is synchronous leading to some adverse affects on loosely coupled document exchanges and thus eventually on fine grained document exchanges in a *DocWF* execution. In particular, limitations are the following two:

1. **Broadcasting public DH key values:** Peers need to broadcast the updated tree node values (i.e., public DH keys) of their key-paths to others or need to request the updates of their sibling-paths to others. It means peers need to know all others in the group a priori to do the broadcasting [KPT00] (see Figure 6.2 (a)). One peer may be nominated beforehand who can take charge of broadcasting the updated public DH key values of current peers after a peer joins or leaves as in the classical TGDH (see Figure 6.2 (b)).
2. **Latency time for a key recomputation:** After a peer joins or leaves existing peers must wait for the new secret key to be computed before any further document exchanges.

In a *DocWF* application, the distributors may take charge of the broadcasting task. However, this significantly increases the key management overhead on the part of distributors as mentioned before. On the other hand, during the latency period the newly delivered documents cannot be read by a peer as a new decryption key needs to be computed. To address these issues, the TGDH scheme is adapted in two significant ways: 1) in initiating key computation and 2) for key synchronization in case of joining and leaving peers. This adaption is made at the initialization of the communication infrastructure of Chapter 5, and rekeying, two phases that are described in the next section.

4 Enforcing Fine Grained Document Authorization by Adapted TGDH

In the scenario of Section 2, a peer may have different subscription interests (different ontology *concept*s) depending on its needs. Then a priori unknown peers having the same *concept* authorizations need to compute a group secret key associated with the *concept*. As argued before, the presence of a centralized entity to compute and distribute the group secret key would constitute a single point of failure. In contrast, when these interests are notified to a distributor by sending corresponding subscription requests as shown in Figure 5.8, authorized subscribers and publishers of documents can share their contribution

to compute a group secret associated with the same *concept* in a distributed fashion. The presence of distributors in the infrastructure (c.f. Chapter 5) facilitates this key computation (see Figure 6.2 (c)) that leverages the adapted TGDH. In particular, for each domain *concept* a group key is computed which is then used by peers to encrypt and decrypt the mapped fine grained document portions. Such a key is computed independently by peers and thus only allows them to access the relevant document portions. This independent key computation supported by the annotations allows a publisher to delegate the authorization decision delegation to the distributors as described in Chapter 5. In effect, the adapted TGDH solution enforces fine grained document authorization as follows:

1. **Dynamic authentication:** The authentication of a peer and determination of a peer authorization for a *concept* is triggered by a subscription request (not statically established).
2. **Independent group key computation:** An authorized peer computes a group key locally and independently of other peers in the group.
3. **Asynchronous rekeying:** The asynchronous rekeying of the group key (as opposed to a synchronous approach of the TGDH scheme) will be based on annotations of documents, that is, it will consist in a document-based lazy rekeying.

4.1 Dynamic Authentication of Peers by Public DH Keys

To identify a peer and to uniquely describe its subscription, a peer uses a special signature in its subscription request that includes the *concept* and an associated DH public key value. Each peer P_i generates a DH key pair: (SK_c^i, PK_c^i) for an ontology *concept*, c, which it subscribes. Based on a new unique private DH key, SK_c, associated with the *concept*, c, the peer generates its corresponding public DH key using the basic Diffie Hellman protocol [DH76] where arithmetics are performed in a group of prime order p with a generator α.

$$PK_c^i = \alpha^{SK_c^i} \quad mod \quad p \tag{6.1}$$

A peer P_i sends a signed (using the private DH key, SK_c^i) subscription request for a *concept* c that also includes the corresponding public DH key, PK_c^i, as a parameter of the request (see Figure 5.8 of Chapter 5; register request (1)) to a distributor D_j.

$$P_{i\in[1,n]} \xrightarrow{[SubscriptionRequest()]_{SK_c^i}} D_j \tag{6.2}$$

The signed request is used by the distributor D_j as follows:

1. **Unique identification:** D_j authenticates the peer P_i by uniquely identifying her through verifying its signature.

2. **Policy checking:** The public DH key, PK_c^i, is further used for policy checking as described in Chapter 5.

3. **Computing control information:** To generate control information for independent key computation. This is detailed in the following section.

A distributor determines groups of peers with the same document authorizations as defined below.

Definition 1. Common Interest Group (CIG): Given a set of subscriptions, a common interest group for an ontology *concept*, C, denoted by CIG_c, is a set of requester identities as can be represented by the public DH keys PK_c^i of the requesters. Formally:

$$CIG_c = \{\ PK_c^z \text{ for z}\in [1,m] \text{ m peers }\}.$$

A CIG_c defines a set of peers who subscribe to the same *concept* and satisfy the policy of the providers of document portions that are associated with the *concept*, C, as described in Chapter 5. Any publisher of the document portions associated to the *concept*, C, is considered to be a member of the CIG_c. In the rest of the chapter, P_i also refers to its associated public DH key PK_c^i.

4.2 Independent Key Computation by a Peer

As mentioned before, the adapted TGDH solution includes a specialized initialization process involving the document providers and communication infrastructure. To initiate the group key computation, a publisher needs to know the first set of public DH values, as denoted by "control data blocks", of the subscribers. In the following, we elaborate on the independent key computation by a peer using "control data blocks".

4.2.1 Enabling Independent Key Computation

As described in Section 3, a peer needs only the values associated to its sibling-path to compute the group secret key (called "control data block"). The initiating publisher (i.e., document originator) of a group of peers, denoted by, I, computes only once this information for each peer right after the initialization process of the distributors (c.f. Chapter 5). A "control data block" is associated to a member peer and essentially is a dynamically computed set of public DH values associated to the sibling path nodes of the peer's local key-path and destined to that member. The knowledge of a "control data block" enables a peer to compute the associated group secret key independently of other members. "Control data blocks" are denoted by $CD_I^{z\in[1,m]}$, and defined as follows:

$$CD_I^{z\in[1,m]} = \{SP_z\}_{PK_c^z} \text{ where } SP_z = \{PK_c^l\};\ l \leq m \text{ for m members in a group}$$

SP_z is a "control data block" that is a set of public DH values associated to the sibling-path nodes of peer P_z. Those values are required to compute the key-path. The initiating publisher, I, encrypts each block SP_z with the public DH key PK_c^z of the peer P_z (resulting into the encrypted block CD_I^z) as determined from its submitted credentials in the signed subscription requests. The distributor then sends it to the peer P_z as follows. Upon receipt of an encrypted control data block CD_I^z, a distributor D_j, first checks whether a peer P_z with a matching public DH key PK_c^z exists in its subscription list. If so, then it sends the matched CD_I^z to P_z:

$$D_j \xrightarrow{CD_I^z} P_z$$

Then, the distributor D_j, forwards other unmatched control data blocks to its downlink distributor which in turn performs the same. Such a message cannot be read by an adversary since each block is encrypted with the public DH key of an authorized member and thus can only be decrypted by a peer possessing the corresponding private DH key.

4.2.2 Initiating a Group Key Computation

The communication infrastructure determines the "control data blocks" of each CIG_c in a three phase protocol. This process is triggered after the first subscription evaluation of all the distributors (c.f. Chapter 5).

- **Phase 1:** A distributor associated to the least $Maximum\ Conceptual\ Block$ sends its subscription list (containing public DH values, i.e., PK_c^i) to its uplink distributor which in turn sends its list together with the received one. This process continues until the distributor associated to the largest $Maximum\ Conceptual\ Block$, denoted as the uppermost distributor, receives a collective list of all successful subscriptions.
- **Phase 2:** The uppermost distributor then determines a group of publishers and subscribers associated to each ontology *concept*, i.e., the publishers who published the document portions with that *concept* and authorized subscribers of the same *concept* that is in a CIG.
- **Phase 3:** For each CIG, the upper most distributor selects (possibly using the selection policy of Chapter 5) the initiating publisher I, to whom the associated CIG is sent. The received CIG list is then used by the publisher to compute the "control data blocks".

4.2.3 Initial Group Key Computation by Peer Contribution

Upon receipt of the CIG list, the initiating publisher, I, exploits the logical key tree structure of TGDH to compute the initial group key. In particular, it generates a key tree by providing its DH private value in one leaf node and taking the DH public values of other peers one by one in other leaf nodes (see Figure 6.3). In the process of such a bottom-up computation of the DH private values in its key-path, a group

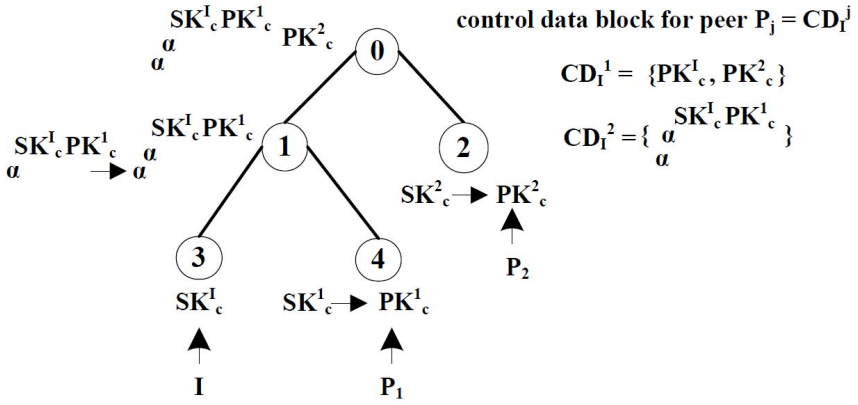


Figure 6.4: Initiator I's key tree with two peers P_1 and P_2. I computes the control data blocks CD_I^1, CD_I^2 for peers P_1 and P_2 respectively.

secret is computed at the root node. Figure 6.4 illustrates this process: a local key tree is generated by an originator, I, as an initiator by taking the public DH values of other two peers P_1 and P_2.

As the initiator of a group collaboration, the initiating publisher does not make the *DocWF* execution a centralized one as all the peers need to compute the group secret key along their key-paths by themselves for later document exchanges. Moreover, the computation is still contributory in nature as other publishers and subscribers in the same group contribute by providing their private DH values in the leaf nodes. Once the initial group key is computed for a *concept*, the publisher can encrypt its corresponding mapped document portions with the computed key before publishing those together with the annotations as described in Chapter 5.

Now, dynamically managing joins and leaves of peers requires updating the current group secret key associated to a *concept* after a successful subscription and unsubscription respectively. As there is no centralized entity, these dynamics must be managed by the peers themselves. To this effect, documents with semantic and security meta data are be used. In order to enforce a non-disruptive document exchanges, a recipient of a document rekeys only if a new key is required to decrypt it. The next section describes such a lazy approach, where rekeying is only triggered on the basis of document arrivals.

4.3 Document-based Lazy Rekeying for Non-disruptive Document Exchanges

As described in Chapters 4 and 5, "Enterprise XML" document portions can be annotated with structural, semantic, and security annotations before publishing so as to delegate the policy checking, routing, and extracting tasks of encrypted nodes to the communication infrastructure. Part of these annotations can also be partial group knowledge, denoted as group update meta data required by a peer for a local key-path computation. This knowledge consists of a cryptographic value accumulated at the immediate child nodes of the root node of the logical key tree. These updates are then communicated when a document is delivered by a distributor to a subscribed peer, thereby suppressing the need for broadcasting. Upon receipt of a document, a subscribed peer can rekey if required to access one particular part of the

document, a process we call lazy rekeying. As such, the lazy rekeying scheme is asynchronous and triggered only when required, that is, upon receiving a document encrypted for the modified group of authorized peers. The group update meta data is further formalized in the following.

4.3.1 Group Update Meta Data based on a Key Tree

Definition 2. Neighbors: A peer P_i's neighbors is a list of peers whose hosted DH public values at the leaves are used to compute the DH private values along the key path of P_i.

Definition 3. Top End Key-path Value (TEK): A peer P_i's TEK is the computed DH private value associated to the topmost node of its key-path.

Definition 4. Top End Sibling-path Value (TES): A peer P_i's TES is the received (as part of a group update) DH public value associated to the topmost node along its sibling-path.

According to these definitions, neighbors have exactly the same TEK and TES for a common interest group as illustrated in the following example.

Example 1. In Figure 6.3, P_1 and P_2 are neighbors to each other and so are P_3 and P_4. The private DH value at node 1 (i.e., $\alpha^{S_1S_2}$) and public DH value at node 2 (i.e., $\alpha^{\alpha^{S_3S_4}}$) are the respective TEK and TES of P_1. On the other hand, for P_2 the TEK and TES are $\alpha^{S_3S_4}$ and $\alpha^{\alpha^{S_1S_2}}$ at nodes 2 and 1 respectively. □

4.3.2 Leveraging Dynamic Group Updates for Lazy Rekeying

It can be observed from the point of view of a peer that any dynamic change in its neighbors due to joining and leaving of peers incurs an update in its TEK and similarly any dynamic change in its non-neighbors incurs an update in its TES. In particular, any incurred dynamic change causes new DH values to be computed in the corresponding key-path and sibling-path that are accumulated in TEK and TES respectively. Technically, these accumulated values of TEK and TES together form a group update meta data.

Lazy rekeying relies on the usage of the group update meta data maintained by each peer in a CIG that are attached to the document as security meta data annotation (i.e., TES/TEK values). The usage of the neighbor list and TES/TEK are as follows:

- **Knowledge of collaborating peers:** The neighbor list contains the public DH values (sent during subscriptions) of previously unknown peers with which P_i is now collaborating. Peers can use this list to identify current collaborating peers for instance. As will be discussed in a later section this neighbor list can be seen as a penalty when enforcing peer anonymity.
- **Lazy rekeying:** A peer updates its neighbor list and TEK value only when receiving a document containing a new TEK value indicating there has been a change in its neighbors. Similarly, a peer updates its TES value only when it receives a document containing a new TES value meaning

there has been a dynamic change (i.e., peers join or leave) in its sibling-path. The TES/TEK value pair being attached with a document merely adds a simple value to the document that makes it suitable for a potentially large scale *DocWF* system.

When group membership changes, an initial re-computation is performed only for the key-paths associated with the initiating publisher and the peer that is subject to join or leave. As such, the initiating publisher and the subject peer can immediately compute the new group key along their key-paths while other peers in the group may be unaware of this new key. The lazy rekeying is illustrated in the following sections.

4.3.3 Lazy Rekeying after a Peer Joins

A joining case of a peer in a group is illustrated as follows. Based on the "control data blocks" of Figure 6.5, the peers P_1, P_2, and P_3 independently compute the initial group key as depicted below.

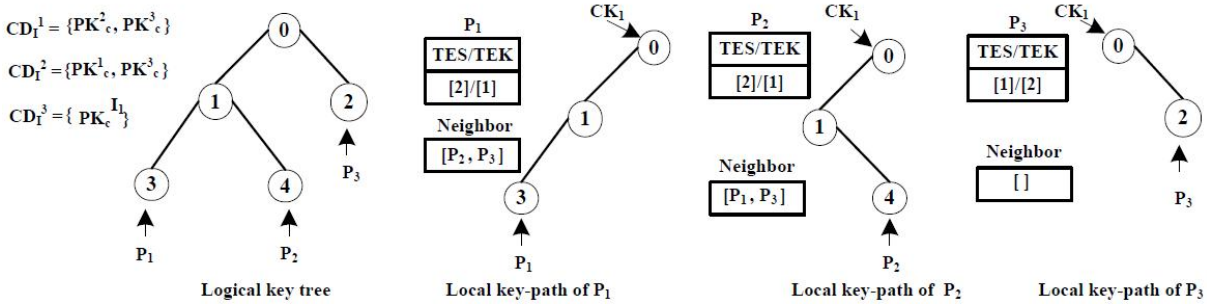


Figure 6.5: Logical key tree of the peers P_1, P_2, and P_3 and their local key-paths.

- **Knowledge of collaborating peers:** In Figure 6.5, the "control data blocks" of P_1, P_2 and P_3 are CD_I^1, CD_I^2, and CD_I^3 respectively. Based on $CD_I^1 = \{PK_c^2, PK_c^3\}$, P_1 determines its initial Neighbor List as $[P_2, P_3]$. Similarly, P_2 and P_3 can also determine their Neighbor list as $[P_1, P_3]$ and $[\quad\quad]$ respectively. Note that, P_3's neighbor list is empty as its $CD_I^3 = \{PK_c^{I_1}\}$, where $PK_c^{I_1}$ is the accumulated cryptographic value, that is the TES for P_3. Thus, P_3 can compute the group key without the knowledge of any public DH keys and therefore without a priori knowledge of any collaborating peer.
- **Updating group meta data:** P_1 and P_2's TES/TEK is [2]/[1][1] and P_3's TES/TEK is [1]/[2].
- **Computing the group key:** All of them have computed the local key paths and thus the group key CK_1.

Now, after a successful subscription of a new peer, P_4, for the same ontology *concept* (Figure 6.6), the distributors can send the associated public DH key (i.e., $PK_c^{p_4}$) to the initiating peer of the same group associated with the *concept* in a similar fashion as in Section 4.2.2. We assume that P_1 is such a

[1]The labeled integer value of a key tree node represents corresponding DH values.

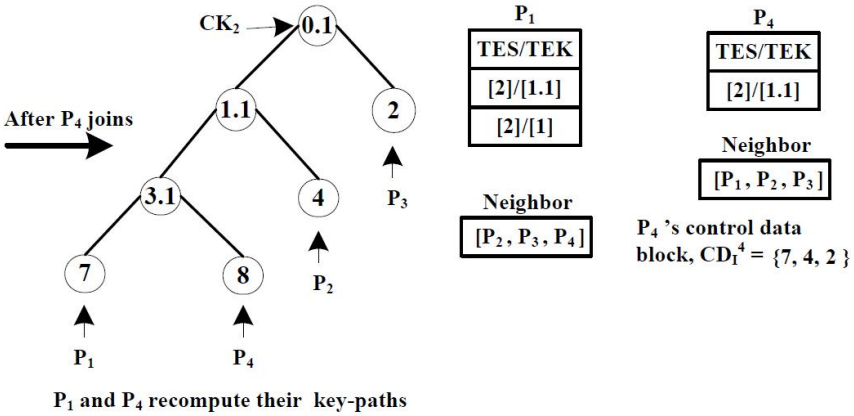


Figure 6.6: Initial rekeying of P_1, and P_4 after P_4 joins.

peer and can initiate the new group key computation with the new peer P_4 as described before. As such, P_1 and P_4 can compute their local key-paths as illustrated below.

- **Knowledge of collaborating peers:** P_1, P_4 can compute their new key-paths and update their corresponding Neighbor List as (P_2, P_3, P_4) and (P_1, P_2, P_3).
- **Updating group meta data:** P_1 and P_4 update their TES/TEK with the new value of [2]/[1.1].
- **Recomputing the group key:** At this point, P_1 and P_4 can compute the new group key CK_2.

However, P_2 and P_3 are unaware of the joining of P_4 in the group at all. Now suppose that P_1 publishes a document portion associated with the same ontology *concept* and annotated with the updated TES/TEK (i.e., [2]/[1.1]). The concerned distributor delivers the document portion to P_2 and P_3 which can then perform lazy rekeying as illustrated below (see Figure 6.7).

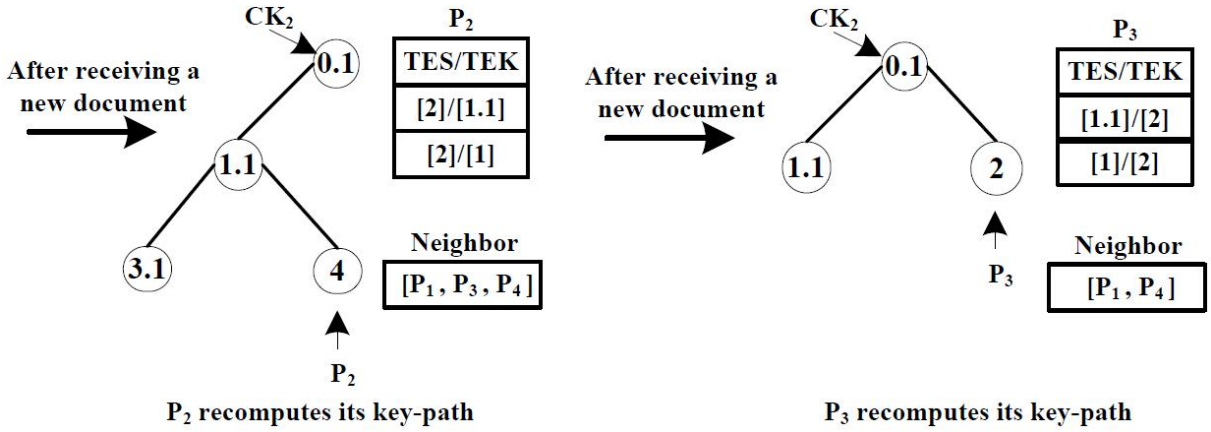


Figure 6.7: Lazy rekeying of P_2 and P_3 after receiving a new document.

- **Document delivery:** The distributor will deliver the document portion to P_2.

- **Knowledge of collaborating peers:** P_2 will notice that there has been a change in its neighbors (i.e., in its TEK value) by comparing its TES/TEK (i.e., [2]/[1]) with the received one (i.e., [2]/[1.1]). P_2 then updates its neighbor list as $[P_1, P_3, P_4]$.
- **Updating group meta data:** P_2 then updates its TES/TEK to [2]/[1.1].
- **Recomputing the group key:** P_2 computes the new group key CK_2 in order to decrypt the received document.

Similarly P_3 can update its neighbor list and TES/TEK value pair and compute the key CK_2 when it receives a document portion annotated with updated TES/TEK (i.e., [1.1]/[2]) (see Figure 6.7). Note that, the sent TES/TEK value pair is inverted for P_3 in comparison with P_2 as P_3's TEK is actually P_2's TES in P_2's key-path.

4.3.4 Lazy Rekeying after a Peer Leaves

As mentioned in Chapter 5, an unsubscription might occur due to either a provider changing its policy or a peer voluntarily leaving the group. In Chapter 5, we sticked to a simple strategy for unsubscription where distributors of the communication infrastructure simply stop delivering documents to the unsubscribed peers. This means that existing peers of the group will remain unaware of some peer leaves and continue to use the existing group key for encryption/decryption of the documents. In terms of forward secrecy, this would open a security breach if existing peers collude with leaving peers and thus unsubscribed peers may get hold of the newer documents. Recomputing a new key by existing peers using the lazy rekeying technique is the solution to this problem.

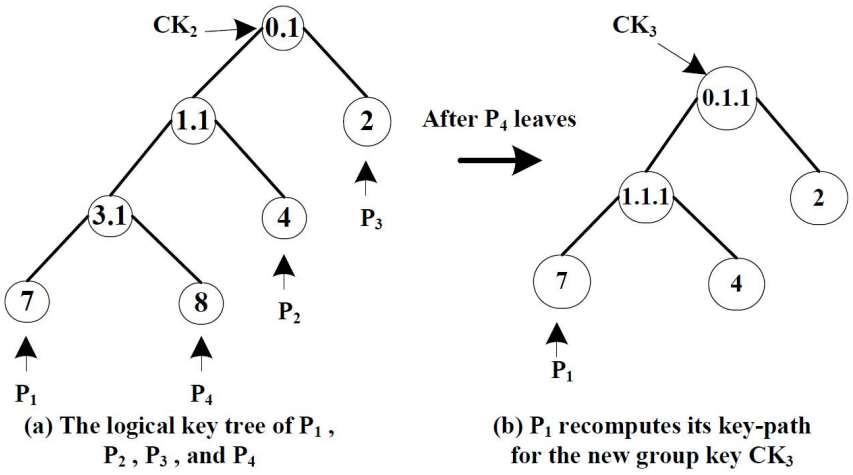


Figure 6.8: P_4 leaves. P_1 rekeys and computes the new group key CK_3.

Considering the peer P_4 of the Figure 6.6 leaves the CIG, P_1 deletes the leaving member node, i.e., 8, in Figure 6.8 (a), from its key-path and recomputes its new key-path resulting the key-tree of Figure 6.8 (b). Other existing members, i.e., P_2, P_3, still can collaborate with their previous knowledge of group key, i.e., CK_2, without stopping the collaboration as they do not even notice the peer leaving.

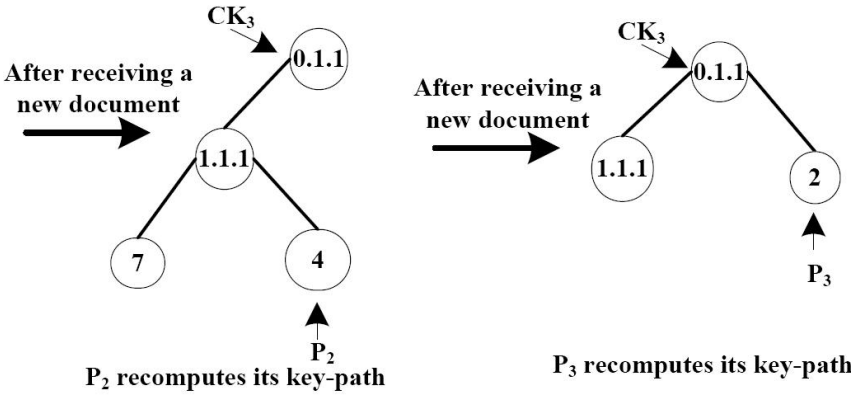


Figure 6.9: Lazy rekeying of existing peers, P_2 and P_3 recompute their key-paths after receiving a new document.

However, to enforce forward secrecy, P_1 would immediately publish a new document with the new group update meta data that will be eventually delivered to the peers P_2, and P_3. After receiving a new document with the group update meta data annotation P_2, and P_3 can recompute the new group key, CK_3, as illustrated in Figure 6.9. Note that dynamic changes in the group are neither broadcasted nor requested yet existing peers continue their collaboration with the previous group key.

4.3.5 Annotating sensitive documents with group update

To attach group update meta data for the fine grained "Enterprise XML" document, annotations of Chapter 5 can be further extended with the TES/TEK value. As described in Chapter 5, such annotations can be associated with finer grained documents, i.e., a node, or more coarse grained document portion, typically a subtree. Section 6 further describes security meta data, including the group update meta data, in order to trace document accesses in a sequence of editions in a posteriori fashion .

5 Enforcing Document Protection

Upon receipt of encrypted and annotated XML nodes based on subscriptions, a peer can perform a semantic integrity verification. Further, it can perform an EBOL-based "Enterprise XML" structural verification (see Figure 6.10) based on the *content signatures* received when subscribing to ontology *concept*s. In the following, A_U and R_U denote respectively the list of $content\ signatures$, and the received set of encrypted and annotated nodes by a peer, U. $Content\ signatures$ are the pairs of node identifier and the *concept*, i.e., (N, C), after a successful subscription and R_U also includes these pairs as meta data (c.f. Chapter 5). Let $\mathcal{N}_A$ and $\mathcal{N}_R$ be the set of node identifiers in A_U and R_U respectively.

Figure 6.10: "Enterprise XML" document protection verification schemes.

5.1 Semantic Integrity Verification

In order to detect any semantics-related authorization violation as specified in Section 2.4, the following verification steps must be performed by a recipient peer.

5.1.1 Authorization Verification (I)

The peer U verifies whether all the *concept*s of A_U it has access to are contained in R_U. The verification is as follows: $(\forall c \in A_U, \exists r \in R_U \ni (N_R, C_i) = (N_A^{'}, C_i^{'}))$, where the pairs $(N_A^{'}, C_i^{'})$ and (N_R, C_i) consist of the node identifier and the *concept* respectively of the *content signature* and the received nodes, R_U. Then for all ontology *concept*s, c, in A_U, if there is a *concept*, r, in R_U with an identical *concept*, c, then all the authorized *concept*s have been received by U (I-verified).

5.1.2 Pruning Verification (II)

U then verifies whether it has received all XML nodes from different documents. It checks a *belong-to* relation between all the document identifiers doc_{id} in the authorized node identifiers of A_U and the document identifiers $doc_{id}^{'}$ of the received node identifiers of R_U. This check is as follows: $(\forall n \in \mathcal{N}_A \exists r \in \mathcal{N}_R | (doc_{id} = doc_{id}^{'}))$; i.e., for each node in $\mathcal{N}_A$, if there is an identical document identifier in $\mathcal{N}_R$, then all the nodes of desired documents have been received by U (II-verified).

5.1.3 Mapping Verification (III)

This can be implied by the authorization verification. Let C_r be the *concept* associated with the node identifiers N_R, i.e., annotation (N_R, C_r), of the received document nodes of a peer. The peer then verifies that such an annotation of a node corresponds to an appropriate *concept*, i.e., the *concept* mapping to the node. In particular, the peer checks whether a same mapping exists in its received list of $content\ signatures$, i.e., $(\forall (N_R, C_r) \in R_U, \exists (N_A^{'}, C_i^{'}) \in A_U, (N_A^{'}, C_i^{'}) = (N_R, C_r))$. If this verification fails then the received document nodes are not associated with the appropriate *concept*s (III- verified).

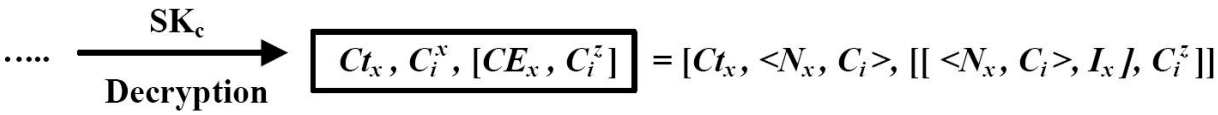


Figure 6.11: After decrypting an "Enterprise XML" node.

5.2 EBOL-based "Enterprise XML" Structural Verification

After a successful semantic verification, a peer U can verify the EBOL-based integrity violations as specified in Section 2.4. U decrypts the received XML nodes in $\mathcal{N}_R$ with the group key, CK_c, associated with the ontology *concept*, C, and traverses each document portion rooted at $x \in \mathcal{N}_R$ in breadth first order. Let x be the current visiting node. Decryption of an XML node x gives access to the annotations (as specified in Chapter 5) of the node together with its content Ct_x shown in Figure 6.11.

In Figure 6.11, CE_x is the content encoding of x. C_i^x is the *content signature* of the node x and C_i^z is the *content signature* of the parent node of x (i.e., z). N_x, C_i are respectively the node identifier of x and the ontology *concept* associated with the node x. I_x is the node integrity of x.

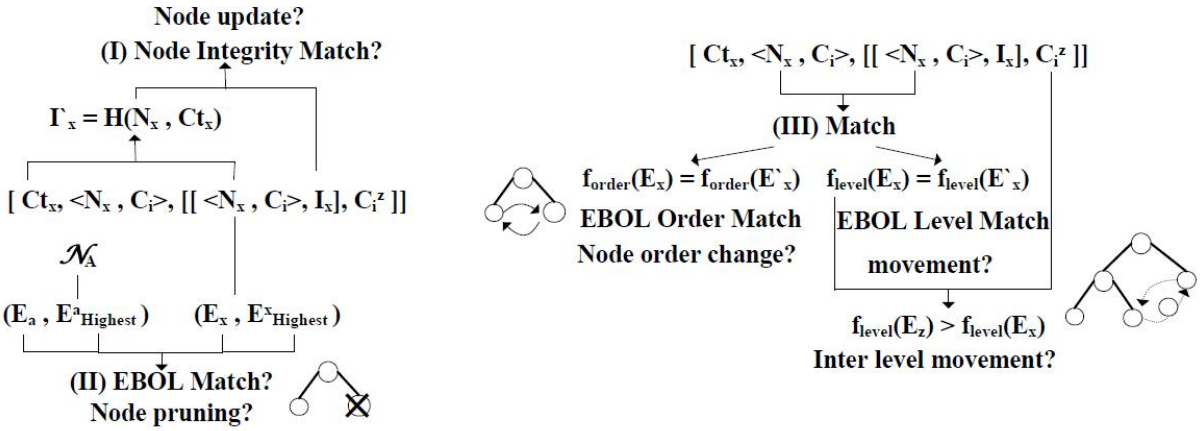


Figure 6.12: EBOL-based "Enterprise XML" structural verification schemes.

5.2.1 Illegitimate Node Update Verification (I)

U takes N_x from the outer C_i^x and x's content, Ct_x, and then computes the local hash of x as $I_x^{'} = H(N_x, Ct_x)$ which is then compared with I_x. If any mismatch is found, the node content has been changed (see Figure 6.12) (I-verified).

5.2.2 Illegitimate Node Pruning Verification (II)

U further checks the *belong-to* relation between all node identifiers of A_U and the received node identifiers of R_U. This check is as follows: $(\forall a \in \mathcal{N}_A \; \exists r \in \mathcal{N}_R | (E_r, E^r_{Highest}) = (E_a, E^a_{Highest}))$; i.e., for

each node in $\mathcal{N}_A$, if there is an identical EBOL-based node identifier in $\mathcal{N}_R$, then all the nodes have been received by U (see Figure 6.12) (II-verified).

5.2.3 Illegitimate Node Move Verification (III)

The verification process continues as the value of the node identifier N_x in the outer C_i^x must match with the inner node identifier N_x in CE_x to be sure that the annotations correspond to the appropriate node. If not, then an illegitimate node movement is detected and the node x can be discarded immediately without knowing the precise violation. To be precise, the elements of outer N_x are compared with the corresponding elements of the inner N_x. (a) if $f_{order}(E_x) \neq f_{order}(E_x^{'})$, where $E_x^{'}$ is the EBOL identifier of the node x in the inner N_x, this means a change in the order of the document nodes is detected. (b) if $f_{level}(E_x) = f_{level}(E_x^{'})$ then the depth level of x in outer N_x is compared with the depth level of the received node in the inner C_i^x. If they do not match then the node x is moved to another depth level (III-semi verified).

The success of the previous element-wise matching does not guarantee a full node moving verification. The depth level of the outer N_x must be compared with the depth level of the parent node of x (i.e., z) in the inner C_i^z. If the latter is not less than the former then it means the node x is moved to a higher level (see Figure 6.12) (III-fully verified).

5.2.4 Illegitimate Node Order Changing Verification (IV)

During the breadth-first order traversal for a current node x, if its EBOL order, $f_{order}(E_x)$, is smaller than that of any previously visited node in the same depth level, $f_{level}(E_x)$, then a node order change is detected. It means that the sibling order of the node x is changed illegitimately. No such detection implies that no order change is performed in the received XML nodes (see Figure 6.12) (IV-verified).

5.3 Document Containment Verification

As discussed in Chapter 2, a digital signature over a complete document can be verified when the whole document is received. Due to the selective routing and delivery of documents performed by distributors this verification technique is not feasible for checking the document containment property in a *DocWF* application. Instead, a mechanism is required that permits to verify the integrity of a document portion without requiring the complete document (including the pruned/filtered nodes) over which the signature was generated. The Merkle tree authentication mechanism [Mer89] to produce a Merkle signature out of a static XML document exactly suits this need as shown in [BCF04] for instance. The following therefore defines the "document containment" property similar to the one discussed in [BCF04].

Definition 5. Document Containment: Given a set of received updated nodes N of a document portion d_i where $N \subseteq d_i$, a Merkle signature of d_i rooted at node $\hat{d}_i$, denoted by $MS_i(\hat{d}_i)$ and the Merkle hash path denoted by $MP(N, \hat{d}_i)$. N is said to be contained in d_i if the locally computed Merkle hash value of the concatenated hash value of the received N and $MP(N, \hat{d}_i)$, i.e., $H(N \mid MP(N, \hat{d}_i))$ is equal to the verified signature value of $MS(\hat{d}_i)$.

A Merkle hash path $MP(N, \hat{d}_i)$ is a list of hash values of the filtered/pruned nodes, i.e., $d_i \setminus N$, that are required to compute the hash value of the root node of the original document from where nodes are filtered. According to this definition, a peer receiving a fine grained document portion from a distributor can verify the integrity of the document portion and whether the document portion is part of an original composite document without having the composite document. In relation to the technique of [BCF04], the novelty here is twofold. First, the document portion, d_i, is dynamic as different portions of a peer data model can be associated to an ontology *concept* depending on different mappings for the same *concept* (see Chapter 4). Finally, the way a Merkle signature and associated Merkle hash path are assembled with previous editions in a nested fashion so as to verify a trace of document accesses is described in the next section. To this effect, publishers must annotate their document portions with the required Merkle hash path values before publishing. Such annotations are also described in the next section.

Example 2. Consider the peer EJNM A of Figure 6.1, publishes its document portion rooted at <CMS> that is associated to the ontology *concept*, $Case$. Let NA B be an authorized subscriber for that *concept* and subsequently receives the document portion. Now, in order to allow the recipient peer, NA B to verify that the document portion rooted at <CMS> is contained at the document rooted at <EJNM>, the publishing peer, EJNM A, must attach the Merkel signature associated to the root node <EJNM> and Merkle hash path values of the nodes <ReqRecTime>, <RequestFrom>, <ReqTo>, and <ReqTime>. NA B can compute a local hash of the concatenation of the Merkle hash values of these nodes and the hash value of the received document portion. This locally computed Merkle hash value should match with the attached Merkle signature value.

6 Enforcing Traceability for a Posteriori Verification

6.1 Peer Anonymity

The anonymity of a peer in a common interest group, CIG, of Section 4.1 can be broken when that peer's identity as represented by its public DH value associated to an ontology *concept* is made public to others. This would not only make a peer identifiable to other member peers but also make her subscription interests public to other members. This might occur either during the initiation of a group key computation or during rekeying. The following discusses the preservation of peer anonymity in these two phases.

6.1.1 Preserving peer anonymity during initial group key computation

During an initial group key computation, the public DH keys would be exposed if peers would exchange their initial share (i.e., public DH values hosted at the leaves of the logical key tree). As described in Section 4.2 this exchange between peers is not required by the adapted TGDH. The communication infrastructure supports publishers in computing initial "control data blocks" by sending the shares of a group of peers. However, a malicious distributor may make the share of a peer public and thus may break peer anonymity. To address this issue, for instance, a policy-based encryption [BM05] can be used by mapping a policy to a key pair. The knowledge of an associated policy private key satisfies a policy. A peer when sending its subscription request to a distributor can encrypt the share with the policy public

key. As the publisher is the policy provider, it knows the associated policy private key and can decrypt the share when forwarded by the distributors so as to generate the "control data blocks" for a group of peers. However, distributors do not know the policy private key and cannot tell for a peer which ontology *concept* it is subscribed to.

6.1.2 Preserving peer anonymity during rekeying

As described in Section 4.3, a host initially joins with only one peer and thus can remain anonymous to other member peers until a new document exchange occurs between the host and these peers. From this point on the host is a collaborating peer to others as described in Section 4.3. Regarding a possible exposure of a peer identity during lazy rekeying by a "group update" meta data annotation which can be formed by only the TES value (in case of a balanced key tree like Figure 6.3) or public DH values of a subset of sibling-path nodes as a "control data block" for a peer to compute a key (as in Figures 6.7, and 6.9). In either case, as the TES/TEK value and the "control data block" consist of only intermediate cryptographic values of sibling-paths of the peers as opposed to their hosted DH values in the leaves, the recipient peer cannot identify associated member peers in the group.

6.2 Traceability of Document Access

In order to track consecutive editions performed by multiple peers on a document portion, it is necessary to check the latest edition with respect to the previous editions performed on the document portion. This relation can be established by a chaining mechanism performed over meta data such as Merkle signature values and Merkle hash path values, which can be attached as security annotations with the corresponding document editions by the editing peer. In that context, a chain represents a trace of document accesses that can be verified a posteriori. Regarding peer anonymity, traceability of document access can also be seen as a penalty like the knowledge of collaborating peers of Section 4 since a verification of a trace discloses the peer identity. Such an annotated document edition is called a document envelope and depicted in Figure 6.13. Essentially, document access trace checks are performed over natural XML node trees. As such, mechanisms developed in the following are with respect to original XML structure as opposed to EBOL-based structural meta data of XML nodes. In effect, agile business scenarios requiring document access traceability should implement these mechanisms for an a posteriori verification. Given that, document access tracing can also be performed after semantic verification as shown in Figure 6.10. Three kinds of checks might be performed over document access traces:

1. **Detecting document integrity and document containment violation:** Has any previous edition violated document integrity and document containment?

2. **Detecting illegitimate insertion and truncation of editions:** Has any edition been performed by an adversary?

3. **Verifying authenticity of an edition:** Have the editions been performed by an authorized collaborating peer?

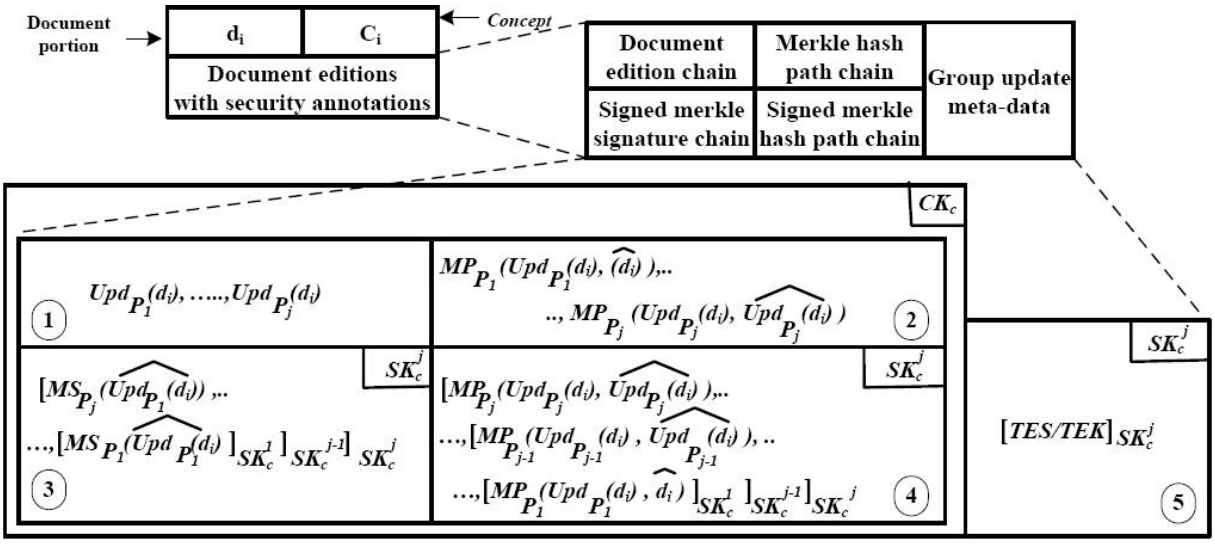


Figure 6.13: Security meta data annotated document editions (document envelope) for tracing document access.

As opposed to verifying against a set of *content signature*s like in the semantic and EBOL-based structural integrity verifications of Section 5, a peer has to verify the document access traces based on the chain of annotations received with the document. This is because any subsequent document update by a peer can not be predicted and thus a *content signature* can not be produced in a subscription phase. Every editing peer should annotate its updated portion before its publication. In the following, we first describe different annotations and then consider their a posteriori verification mechanisms.

6.2.1 Annotating Document Portions for Access Traceability

Access tracing annotations consist of plaintext data and security meta data. The plaintext data is further categorized as follows:

1. **Document editions:** The document editions performed by peers, $(Upd_{P_1}(d_i), ..., Upd_{P_j}(d_i))$ is the chain of updated document portions of d_i from the peers $P_1, ..., P_j$ respectively (see Figure 6.13).
2. **Merkle hash paths:** The Merkle hash path chain consists of multiple Merkle hash paths (i.e., $MP_{P_j}()$), each of them corresponding to the updated document portion, d_i, and computed by an editing peer P_i. The chain is denoted by $(MP_{P_1}(Upd_{P_1}(d_i), \hat{d}_i), ..., MP_{P_j}(Upd_{P_j}(d_i), \widehat{Upd_{P_j}(d_i)}))$, where $MP_{P_i}(.., ..)$ is the Merkle hash path corresponding to d_i and is computed by a peer $P_{i\in[1,j]}$ (see Figure 6.13). This list is required for a recipient peer to compute locally the corresponding Merkle hash values associated with the updated document portions.

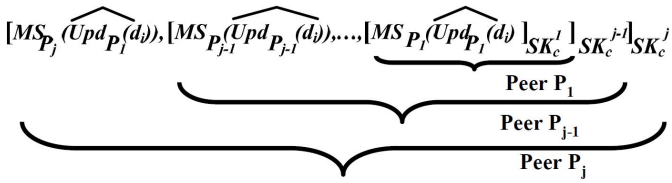
A document edition, $Upd_{P_j}(d_i)$ and an associated Merkle hash path, $MP_{P_j}(Upd_{P_j}(d_i), \widehat{Upd_{P_j}(d_i)})$, respectively represent the edition performed by a peer, P_j, and consolidated hash values of the pruned

nodes with respect to the document portion d_i. Then in Figure 6.13 the document edition chain (1) and the Merkle hash path chain (2) represent two sequences of plaintext editions and their associated Merkle hash paths respectively where the latter is a security meta data for protecting document containment. To protect document integrity and document containment with respect to the document portion each document edition and associated Merkle hash path are further signed by the editing peer and thus yielding the security meta data, i.e., signed Merkle signatures (3) and signed Merkle hash paths (4).

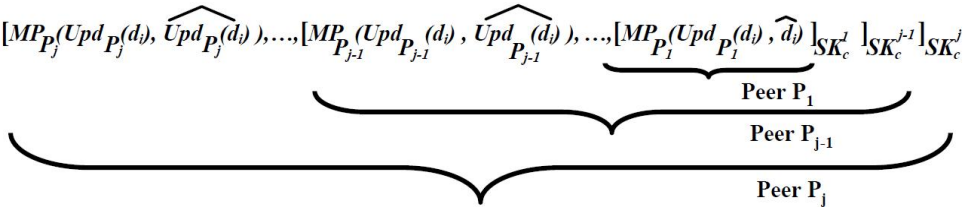
1. **Chain of Merkle signatures:** The signed chain of Merkle signatures of the document editions (Figure 6.13),
2. **Chain of Merkle hash paths:** The signed chain of Merkle hash paths is associated to the document editions.

In order to protect the document integrity of the document edition, $Upd_{P_j}(d_i)$, the editing peer, P_j, produces a Merkle signature of the edition as denoted by $MS_{P_j}(\widehat{Upd_{P_j}(d_i)})$. The Merkle signature and the Merkle hash path associated with the edition together also protect the document containment of the edition with respect to the document portion, d_i. Then to prevent an adversary from inserting invalid editions and truncating valid editions, an editing peer, P_j, signs its produced Merkle signature and Merkle hash path with its private DH key, SK_c^j in a nested fashion together with the received ones. The building processes of these security meta data including group update meta data are illustrated in the following.

Building a chain of Merkle signatures: Each editing peer, P_j, computes a Merkle signature over the root node of $Upd_{P_j}(d_i)$ which it signs together with the received Merkle signatures from the previous peers using its private DH key, SK_c^j, associated with the ontology *concept*, C. The computed signature yields the following chain value (Figure 6.13).



Building a chain of Merkle hash paths: Each peer P_j signs its Merkle hash path $MP_{P_j}(Upd_{P_j}(d_i),\widehat{Upd_{P_j}(d_i)})$ together with the received Merkle hash paths starting from the originating editor, i.e., P_1, with its private DH key, SK_c^j, yielding a similar chain as the document editions:



Annotating group update meta data: The final block contains the group update, i.e., TES/TEK, and signed by the editing peer, P_j, that results in $[TES/TEK]_{SK_c^j}$.

To prevent an attacker from accessing the document editions including the security annotations, an editing peer, P_j, encrypts all with the associated group key, CK_c and attaches an associated group update meta data before publishing it ((5) of Figure 6.13). Based on the group update, a recipient peer may perform lazy rekeying as described in Section 4.

6.2.2 Verifying a Trace of Document Access

Upon receipt of a document with the security annotations, a verification of a trace of document accesses, i.e., a document edition chain, can be performed over the chains of Merkle signatures and Merkle hash paths in an iterative fashion (see Figure 6.14).

The document editions and security annotations are only disclosed to a peer possessing the group key and thus remain confidential to others (outsiders, distributors and unauthorized peers with respect to the ontology *concept*) unless they are able to perform lazy rekeying (c.f. Section 4).

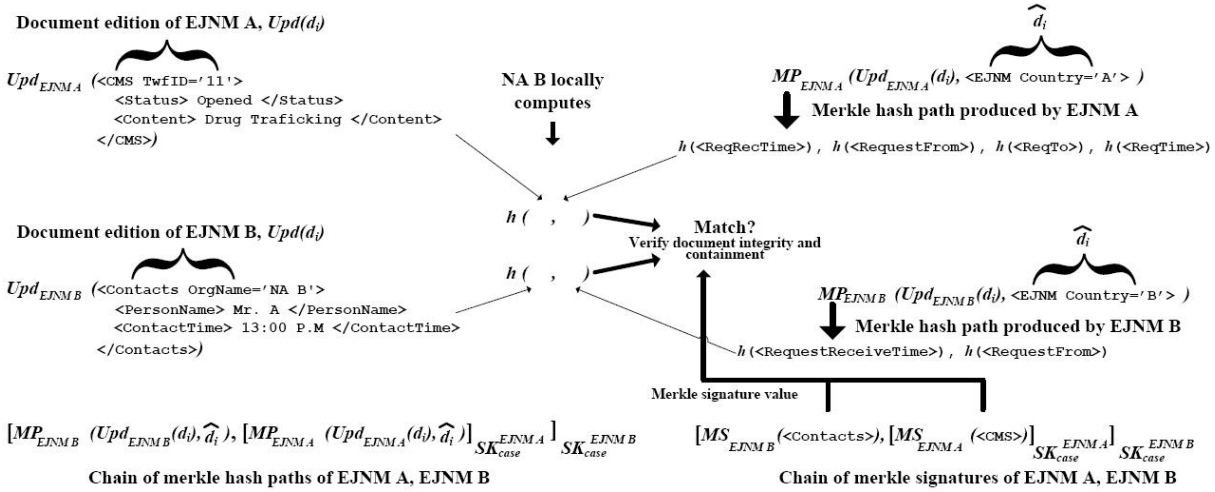


Figure 6.14: Verifying document integrity and containment in a trace of document accesses.

Verifying document integrity and containment: Upon receipt of an annotated document portion, any peer P_j can check the document integrity and authenticity by computing the hash out of the concatenated value of the received Merkle hash path, $MP_{P_1}(..), .., MP_{P_{j-1}}(..)$, and of the locally computed hash values of corresponding document editions, i.e., $Upd_{P_1}(d_i), .., Upd_{P_{j-1}}(d_i)$. Each locally computed Merkle hash must match with the corresponding Merkle signature values of the received chain:

$$[MS_{P_{j-1}}(\widehat{Upd_{P_{j-1}}(d_i)}), ..., [(\widehat{Upd_{P_1}(d_i)})]_{SK_c^1}]_{SK_c^{j-1}}.$$

Example 3. Continuing the example 2, let us assume that the peer EJNM B of Figure 6.1, publishes its document portion rooted at `<Contacts>` that is associated with the *concept* $Case$. As NA B is an authorized subscriber of that *concept* NA B subsequently receives the document portion. NA B can verify the integrity and containment of the two document portions rooted at `<CMS>` and `<Contacts>` that are published by EJNM A and EJNM B respectively. This is illustrated in Figure 6.14. The

Merkle hash path values of the document portion rooted at <CMS> are the hash values of the nodes <ReqRec>,<RequestFrom>,<ReqTo>, and <ReqTime>. NA B computes a local hash of the concatenation of these nodes and hash value of the received document edition rooted at <CMS> which should match with the associated Merkle signature in the chain. Similarly, NA B computes a local hash of the concatenation of the nodes <RequestReceiveTime>, <RequestFrom> and the hash value of the received document edition rooted at <Contacts> to match with the associated Merkle signature in the chain. In both cases, if matching is successful then document integrity and containment is protected.

Detecting illegitimate insertion and truncation of editions: A Merkle signature using the private DH key over a document edition is performed over the received Merkle signature chain and results into a nested Merkle signature chain as shown in Figure 6.13. This prevents any adversary including malicious peers in the group to include or exclude any fake edition and Merkle signature as illustrated below.

The nested Merkle signatures are verified in an iterative fashion where each successful verification pills off the outermost Merkle signature and unveils the next inner nested Merkle signature. The unveiled one in turn will be verified and this process continues until the innermost Merkle signature associated to the first edition is verified. A similar nested signature mechanism is also performed for the Merkle hash path chain by an editing peer that also prevents an adversary from including or excluding Merkle hash paths. Now any insertion of a fake edition or truncation of a valid edition involves a manipulation of the security meta data, i.e., Merkle signature chain and Merkle hash path chains. In particular, to insert a fake edition requires an insertion of an associated Merkle signature and a Merkle hash path in the corresponding chains of Figure 6.13. Such insertions and truncations are only feasible by an insider adversary that is a malicious group peer possessing the group key CK_c, who can decrypt the encrypted block for a later illegitimate insertion or truncation of editions. According to the ontology-based document modeling of Chapter 4, the insertion of an invalid edition implies a mapping that associates irrelevant document vocabularies to the ontology *concept* and results into attaching illegitimate Merkle signature and Merkle hash path in the chains (see example 4). Such an illegitimate insertion can be seen as a penalty of the ontology-based modeling. However, a truncation of a Merkle signature or a Merkle hash path from the corresponding chain makes the chains invalid which can be detected by a recipient peer during an iterative verification of the nested signatures as described before.

Example 4. The mapping of the ontology *concept*, $Case$, to the document portion rooted at <Contacts> by the peer, EJNM B, in example 3 can be seen as an irrelevant mapping as the contact information rooted at <Contacts> have no direct relation with the *concept*, $Case$. Consequently, the Merkle signature verification for the edition represented by the <Contacts> and the attached Merkle hash path values of <RequestReceiveTime> and <RequestFrom> will evaluate to a matching. Thus document integrity and containment of the edition rooted at <Contacts> are protected. Now assume the Merkle signature of the EJNM A, MS_{EJNMA}(<CMS>), and associated Merkle hash path are truncated from the Merkle signature chain and Merkle hash path chain respectively. This truncation is detected immediately by NA B when verifying the truncated chain.

Verifying authenticity of an edition: Upon decrypting the encrypted data block by the group key, CK_c, any peer in the group can verify the authenticity of the received document editions by matching a signing peer identity with an existing collaborating peer. If there is a match then the received document edition is originated from a collaborating peer of the recipient peer.

7 Security Analysis

The parameters that are relevant to the security properties offered by the mechanisms developed in this chapter are mainly twofold. First, regarding the group secret key, CK_c, associated with an ontology *concept*, C, as computed by the peers using the adapted TGDH mechanism: The independent key computation by a recipient peer relies on the "control data block" distribution during the initiation phase of a common interest group, CIG, and group update meta data attached with the delivered document during a lazy rekeying phase by a publishing peer (Section 4). The distribution of the control data blocks and the delivery of documents are controlled by the communication infrastructure nodes, which may be curious as mentioned in the attacker model (Section 2.3). A curious communication infrastructure node may distribute inappropriate control data blocks and send invalid group update meta data during a document delivery that may result in computing invalid keys. Besides, the trustworthiness of business peers can not be controlled, especially when it comes to sharing documents with unauthorized peers once the group secret key has been computed. Second, concerning the generated EBOL pairs of "Enterprise XML" nodes: the trustworthiness of business peers also can not be controlled when publishing annotated XML nodes with EBOL-based meta data, for instance, the node identifier and node integrity attributes of Chapters 4 and 5. In this context, the mechanisms presented in this chapter verify some properties that do not depend on the underpinning "control data block" distribution mechanism and EBOL-based meta data while some others do. In the security evaluation, we assume the following:

1. **Security of Diffie-Hellman Key Pairs:** The public key encryption scheme used in the specification of the Diffie-Hellman key pair (PK_c^i, SK_c^i, i.e., DH keys) that is associated with the ontology *concept* C is semantically secure against a chosen ciphertext attack. Further, the public DH keys sent during subscription (Figure 5.8 of Chapter 5) are assumed to be signed by a trusted third party so that no malicious party can subscribe to an ontology *concept*.
2. **Security of EBOL:** The encryption scheme [AKSX04] used in Chapter 4 for the specification of the EBOL, i.e., $(f_e(f_{order}(B_a)), f_e(f_{level}(B_a)))$, of an XML node a is semantically secure against a chosen ciphertext attack and the associated signature scheme achieves signature unforgeability.

Theorem 1. (**Fine-grained document authorization:**) Only authorized peers can exchange documents and thus participate in a *DocWF* execution.

Proof: This means that only authorized peers based on successful subscriptions can access the plaintext document nodes and thus get access to the published document nodes associated with the ontology *concept*.

According to the enforcement mechanism of Section 4, one can neither compute the group secret key nor rekey unless it possesses the associated "control data block" of the group. Given the initiating publisher generating "control data blocks" and under the first assumption, it is not feasible for an attacker to extract the list of public DH keys of the "control data block" if the attacker does not satisfy the policy associated with the *concept*. This is true as the "control data block" is encrypted by the individual (i.e., legitimate peer) public DH key meaning that an attacker can not possess the associated private DH key as it does not satisfy the document provider policy (see Section 4.2). It implies that an attacker can not compute the group secret key associated with the *concept*. For the same reason using a group update meta data, an attacker can not rekey as it can not decrypt any possibly intercepted "control data blocks". Therefore, no attacker can exchange valid documents and thus participate in a *DocWF* execution. □

The proofs of the theorems 2, 3, 4 and 5 are based on the security meta data of Figure 6.13 and thus apply to document access traceability. Similar proofs can be specified based on the EBOL-based XML annotations.

Theorem 2. (**Document confidentiality:**) The confidentiality of an updated document portion is ensured.

Proof: This refers to the fact that even if some encrypted document nodes with the group update meta data are in possession of an adversary, the document editions will still remain confidential to that adversary.

This property is achieved in a straightforward way: the document edition block containing document updates is encrypted by the group secret key computed by the peers of a common interest group. The exposed group update meta data of Figure 6.13 can only be exploited by an adversary who has been in the same group and thus can determine the key or can rekey based on its previous knowledge of the group key-path (see Section 4.3). As such, under the first assumption, no unintended recipient of document nodes can rekey and decrypt the nodes. Therefore, the published document nodes associated with an ontology *concept* always remain confidential to an adversary. □

Theorem 3. (**Document integrity protection:**) Upon receipt of document nodes, an authorized group peer can detect the document integrity violation of the document.

Proof: It indirectly means that any inadvertent or intentional change in the document content by any peer can be detected and that the structures of the annotated document nodes and the envelope of Figure 6.13 is unforgeable.

Assuming that a legitimate member peer of a CIG builds the document envelope based on the methodology described in Section 6, assuring the unforgeability means verifying that:

1. An original document envelope built by a peer cannot be replayed by another malicious peer under its name.
2. A document envelope cannot be built by an attacker that is not trusted by collaborating peers.

To replay the same document envelope by an attacker under its name, an attacker has to provide a Merkle signature which is to be verified with the received Merkle hash value of the document editions (see Section 6). This verification fails as the signature from the attacker will not match with the hash value (by the first assumption). In effect, the first property is enforced by the fact that a document edition chain structure in the envelope, built by a trustworthy peer is bound to its Merkle signature chain and Merkle hash path chain and thus cannot be reused during an attempt by an attacker to send a crafted document edition.

The second property is straightforward by the confidentiality claim 2 and by the facts that an attacker neither gets the group secret key nor can compute it. In effect, even though the attacker may build the edition chain it is not able to encrypt those by an appropriate group secret key in order to build the complete document envelope structure. □

Theorem 4. (**Document traceability:**) Upon receipt of a document envelope an authorized peer can trace the document updates.

Proof: As each peer signs its document edition along with the previous series of editions performed by previous editors, the recipient can trace everyone's updates by simply verifying the signatures iteratively by leveraging the methods of Section 6. □

Theorem 5. (**Document containment:**) Upon receipt of an updated document portion, a peer can verify that the updated document is contained in the original document.

Proof: This means that, after decrypting the encrypted document nodes of the document portion any peer in the group can verify that the received document portion was actually a part of the original document without receiving the complete document. This is straightforward as specified in Section 5.3.

A receiver computes a Merkle hash of the combination of the received Merkle hash path and locally computed hash of the updated document nodes. This locally computed Merkle hash value must match with the verified signature value of the received Merkle signature associated with the document portion. □

8 Related Work

Security of cross-organizational workflows and XML-based applications in both centralized and decentralized settings has been an active research field over the past years. Regarding cross-organizational workflows the main focus has been on resource access control [Coa98, KR01, BFA99] and task authorization issues [LZS09, ACM01, CLW05, KPF01]. XML-based document security hardly considers workflows and is rather centered around structure-based document protection. However, in the document-based agile workflow setting document security issues, for instance, document integrity protection is beyond the structure of a document. Moreover, key management challenges raised by the communication infrastructure such as independent group key computation associated to an ontology *concept* and lazy rekeying without a priori knowledge of peers, which are presented in this chapter have been left out.

The enforcement of access control policies within workflows has been a quite active research field over the past years both in centralized and decentralized settings. Most approaches rely on the Role Based Access Control model [SCFY96, WT04] to implement access control policies within the execution of a predefined workflow of business tasks [ASKP00, LZS09]. These mechanisms are used to control either access to resources or authorizations of business tasks as discussed in the following Sections 8.1, and 8.2.

8.1 Access Control to Resources Within Workflows

In the centralized setting, many access control infrastructures have been proposed for business processes executed in the Service Oriented Architecture paradigm [TLC05, WT04, AkH96]. In [KPF01, KM03]

centralized infrastructures are proposed which protect resources that should be accessed during the execution of a workflow based on credentials provided by business peers.

Despite significant contributions, these approaches do not however meet the requirements introduced in a *DocWF* setting as they rely on a centralized component to issue access control decisions. In fact, these are designed to protect local resources rather than workflow data that are transferred between peers without a centralized point of coordination.

8.2 Task Related authorizations Within Workflows

In [BCP06] an approach based on XACML is presented to enforce access control policies so that BPEL activities are executed by authorized users during the execution of BPEL processes. In [ACM01] a security model based on the Chinese Wall paradigm is proposed to distribute different task executions to organizations based on conflict of interests. A methodology is specified in [WKRL06] to determine the set of credentials required by a user to be able to execute some tasks of a workflow based on requirements specified in terms of security policies. In [LZS09] the authors introduce a model to dynamically assign permissions to execute tasks and in [MM07] authors propose a dynamic assignment of business peers to some predefined business tasks.

8.3 Access Control Decision Delegation Within Workflows

Few authors have addressed security issues within distributed collaborative applications. In [PH03] a solution for enforcing RBAC policies is presented to protect access to peers that are part of a peer-to-peer community. In this approach peers are able to make access control decisions autonomously without relying on an external policy decision point. In [TAK03] an access control model enabling the definition of dynamic RBAC policies enforcing dynamic separation of duty constraints is presented. The enforcement of these access control policies however relies on a dedicated coordinator which acts as a centralized entity. Such an entity knows the future requesters and as such can already issue static access control decisions. However, in a *DocWF* execution, the document providers solely provide the authorization policies that are enforced by encryption over fine grained documents. Thus the actual access control decision is realized later by the communication infrastructure which is itself a distributed system. The distributor nodes of the communication infrastructure perform this by selectively routing and delivering the documents based on their annotations.

8.4 Document Access Control

In this section we review a few enforcement solutions for XML document access control and highlight the differences with our work.

8.4.1 Encryption-based enforcement

Encryption as an enforcement mechanism for access control decisions made at a server has been discussed in the literature for a while [BF02, MS03]: the server encrypts the data it stores with secret keys; the client can access these only if it possesses the right decryption keys. This technique supports dynamic change only through the use of the server as a centralized point of enforcement that computes and distributes keys and therefore constitutes a single point of failure. Scalability and performance are central issues growing with the number of clients accesses, notably regarding the need for the partial reencryption of data because of changes in the access control rules. This technique also does not address the need for distributed sources of data. It should also be mentioned that these approaches altogether do not address traceability issues with respect to document updates.

8.4.2 Tamper-resistance for access control

The use of tamper-resistant modules as described in [BNP08] makes it possible to alleviate the limitations of the latest approach regarding policy dynamicity, both in terms of the access control decision and its enforcement. This approach however requires the difficult and expensive deployment of a trusted infrastructure. We instead think delegation might be enough to adapt the access control policy in most scenarios.

8.4.3 Centralized server-based decision and enforcement

Despite significant improvements in the area of fine grained access control on XML documents, for instance, in [DdVPS02], the enforcement relies on a centralized client-server framework. Clients request the server for accessing a document. The server, which is responsible for designing the document schema, decides about the authorizations and at the same time enforces access control on the document.

In [DdVPS01] the authors describe a fine grained access control technique for SOAP based communication among Web Services. However, the aforementioned work does not address the particular security requirements of collaborative XML document edition as described in this chapter.

From the enforcement perspective, these approaches are known also as the view based XML access control. However, the view based approach inherently contains two significant limitations [BLL04]: computation and storage scalability. As an increasingly large number of requesters is involved, the management of views does not scale up and the increasing number of documents and clients demands more storage and processing cost on the server side. The view based approach also does not consider the issue of document updates where documents are dynamically exchanged among several participants and in particular document protection aspects.

The most dominating assumption of those approaches is the fact that all the accesses are decided and enforced by a central entity in charge of the XML data sources.

8.4.4 A posteriori verification

In [MFBK06] some mechanisms and algorithms are introduced for the cooperative updates of XML documents in a distributed environment. While this work relies on the use of cryptography to support controlled document edition similarly to our work, it does not consider distributed sources of documents. This approach is more tailored to the a posteriori verification of the correct execution of a document edition process. The authors in [EW07] propose an a posteriori policy enforcement approach for data objects that move from one system to another. This approach is similar to our document access traceability mechanisms in that both approaches provide audit and accountability, which strongly encourages the trustworthy behavior of interacting peers. However, the approach of the authors remains only in formal level without specifying a solution for specific data objects, for instance, documents in document-based workflow applications.

9 Conclusion

This chapter presented the mechanisms necessary towards meeting the security requirements raised by the execution of a *DocWF*. Our solution is based on a group-based cryptography and specialized security annotations. We adapted the TGDH solution to enforce fine-grained and dynamic document access control according to an ontology-based policy of the document providers. These policies are checked by the distributors of the communication infrastructure. Annotations including document security meta data are exploited by peers for verifying various semantic and structural integrity properties of the documents. Besides, the adapted TGDH solution provides peer anonymity which can however be compromised when tracing the access of peers to the document, for instance, in sensitive workflow instances. The document access traceability is assured by the adapted TGDH combined with security annotations for a posteriori verification that can easily be integrated into an execution of a *DocWF*.

The work done in this chapter has been published in the proceedings of The EDOC '08: Proceedings of the 12th International IEEE Enterprise Distributed Object Computing Conference, September 18-20, 2008, Munich, Germany [RRS08].

Chapter 7

Conclusions and Perspectives

The best way to predict the future is to invent it.
- Alan Kay -

In this thesis we introduced the concept of a document-based agile workflow (*DocWF*) system for agile business processes. We have specified *DocWF* models that are based on business goals and associated business rules and therefore the *DocWF* models are independent of any business task specification. Business peers then proactively determine suitable business tasks and their binding to IT components, e.g., Web Services. We have provided adequate semantic solutions to achieve the non-disruptive interoperability of the document exchanges required by a priori unknown peers. We designed a decentralized communication infrastructure enabling the selective distribution of semantically enriched documents to legitimate peers to support the decentralized execution of a *DocWF* application. We also specified security solutions to offer the guarantees required by the execution of *DocWF* applications in terms of document security and for addressing the vulnerabilities raised by the communication infrastructure. This concluding chapter summarizes the contributions described in this report and is organized as follows. We first summarize our contributions and their applications. The possible execution modes supported by the document-based agile workflows are then specified and we finally present some possible research directions.

1 Summary

The contributions of this thesis encompass the four following research areas.

Design of a document-based agile workflow system for agile business processes: As opposed to modeling a routine business scenario by typical task-based workflows, agile business scenarios that cannot be generalized by predefined flows of pre-specified business tasks can neither be modeled by tasks nor the tasks can be bound to pre-configured IT components for latter executions. To that effect, we suggested in Chapter 3 a high level business process model supporting a loosely coupled execution that is triggered by exchanging semantically enriched documents between *DocWF* peers. The model can be checked both at design time and at runtime using a set of constraints defined on the business logic. A loosely coupled execution is realized by the dynamic task enactment of the models that features a fully decentralized execution and supports a late binding of goals to suitable tasks and the runtime determination of corresponding services. In short, the developed *DocWF* models and their loosely coupled executions assist a peer of a *DocWF* execution in the following ways:

1. **Loosely coupled models:** *DocWF* modeling technique does not rely on existing services as opposed to typical task-based workflow modeling.
2. **Reuse:** Reuse is enabled not only during runtime task enactment but also during a *DocWF* modeling by allowing a peer to determine suitable tasks that may vary for the same goal for instance.
3. **Distributed control:** A peer can determine suitable tasks and their concrete binding independently and thus there exists no centralized point of control during a *DocWF* execution.
4. **Dynamic task enactment:** Task enactment is dynamic for the actual binding of tasks to concrete services by, for instance, enabling semantic-based service discovery at runtime.
5. **Modeling error detection:** *DocWF* models enable the early detection of design and execution errors and their fixing without restarting a process.

Specification of interoperable documents for a *DocWF* execution: We presented in Chapter 4 a semantic enabled document specification and a document comparison technique to implement interoperable documents. Interoperable documents support the interoperability required for document exchanges during an execution of document-based agile workflows. These approaches meet the requirements of non-disruptive document exchanges and distributed document handling by a priori unknown business peers as specified in Chapter 4. Relying on a business domain ontology as a stable interface as opposed to relying on a syntactical interface (e.g., XML schema), distributed document handling enables the following:

1. **Semantic enabled enterprise XML:** Exchanged documents are annotated with adequate semantic information so that varying vocabularies of "enterprise XML" can be understood beyond business boundaries.
2. **Fine-grained document access:** Document authorizations are specified at the semantic level, and are then enforced at the document instance level. The right to access multiple document instances or multiple document nodes are specified just based on such simple authorization patterns thanks to the decoupling of our semantic based definition.
3. **Document convergence:** All legitimate peers in a *DocWF* execution can access consistent document portions.
4. **Enabling document comparison:** Peers are able to distinguish documents and their content (i.e., both structurally and semantically).

Decentralized communication infrastructure for document-based agile workflow executions: Interoperability mechanisms of a *DocWF* include annotations of "enterprise XML" and their comparison solutions to enable fine grained document exchanges beyond business boundaries. Yet, these methods and techniques do not provide any guarantee on non-disruptive exchanges of fine granular documents amongst peers which further process those documents in order to achieve their goals. Considering the absence of a centralized coordinator akin to distributed environments, a decentralized communication infrastructure for the execution of a *DocWF* is developed in Chapter 5 to enable fine grained document exchanges amongst a priori unknown peers. Relying on an ontology-based stable interface amongst peers, the communication infrastructure offers the following:

1. **Loosely coupled document exchanges:** Business peers are able to exchange fine granular documents without a priori knowledge of each other.
2. **Selective document routing and delivery:** Multiple fine grained documents associated with the same ontology *concept* may be originated from multiple providers. Given that, those semantically equivalent document portions can be routed and delivered to appropriate peers according to providers policies.
3. **Fully distributed communication:** Communication including document exchanges amongst a priori unknown peers are supported by the distributors as opposed to a dedicated coordinator.

Security of document-based agile workflows: Security solutions to support a secure execution of document-based agile workflows were developed in Chapter 6. As mentioned in Chapter 6 a *DocWF* execution raises new security concerns and challenges with respect to documents and the communication infrastructure as opposed to typical distributed workflow management systems. Document security solutions include the integrity protection of fine grained documents and the enforcement of authorizations over fine grained documents. While encryption over fine grained documents is used as an enforcement technique, a special key management solution is developed that includes the computation of a group key associated with an ontology *concept*. Using this solution peers from different origins yet with the same authorization for an ontology *concept* can compute the group key independently and recompute the group key asynchronously when a peer joins or leaves. Such an independent and asynchronous group key computation by peers involved in a *DocWF* execution limits disruptions in the execution of the *DocWF*. In addition, traceability solutions protect a series of related document updates by peers from various integrity violations of adversaries, for instance, document containment violations, and illegitimate insertions and truncations of editions. These security solutions mainly comprise:

1. **Document authorization**: As opposed to task-based authorizations the main security solution associated to documents is the enforcement of fine grained document authorizations according to an agreed document semantics.
2. **Document protection**: This consists in the enforcement of fine grained document authorizations and in the protection of the integrity of fine grained documents according to the ontology-based policy specification.
3. **Traceability for a posteriori verification**: Peer anonymity in a *DocWF* execution is also assured when required which however can be compromised if document access of peers needs to be traced.

These solutions capitalize on the group-based cryptographic technique called tree-based group Diffie-Hellman (TGDH) as detailed in Chapter 6. We adapted the TGDH technique to enable a group of peers to compute the group key independently where peers have the same authorized subscriptions for an ontology *concept*. In particular, it restricts document access through the encryption of a document portion with a key computed by a group of peers with similar access rights. This independent key computation allows a legitimate peer to encrypt/decrypt documents autonomously in a fine grained manner and to prevent malicious business actors from performing unauthorized access and actions in documents, for instance, illegitimate updates, deletions, insertions and even displacements of nodes. The adapted technique limits the scope of rekeying when peers dynamically join or leave in an agile environment so as to enable non-disruptive document exchanges for an extended period of time as required for the execution of a *DocWF*. Traceability for an posteriori verification is also enabled by this mechanism with some special security meta data consisting in particular of the chain of Merkle signatures and the chain of Merkle hash paths. The design of the suggested mechanisms is strongly coupled with the communication infrastructure.

2 Implementation

The theoretical results presented in thesis have been implemented using Java technologies. The implementation work we pursued is summarized in this section.

Document distributor implementation for the communication infrastructure: We designed a document distributor service (appendix A) that implements the semantic enabled policy checking mechanisms of Chapter 5 that enables selective document routing and delivery during an execution of a *DocWF*. Such a distributor service can be deployed on a communication infrastructure node, for instance, an ISP provider or a telecom provider that may want to support the decentralized communication infrastructure of a *DocWF*.

Security libraries for adapted TGDH and traceability in a document-based agile workflow execution: We developed security libraries (appendix B) implementing the adapted TGDH for key management and Merkle hashing for enterprise XML to support a secure execution of a *DocWF*. The implementation of the adapted TGDH is based on a Java Cryptography Extension provider that supports the basic Diffie Hellman cryptosystem. To support document traceability for a posteriori verification we implemented Merkle hash tree mechanisms for enterprise XML. The security libraries can be integrated into a peer node and distributor nodes depending on the execution modes as described in the following section.

3 Execution modes supported by document-based agile workflows

Agile application scenarios have their individual peculiarities as illustrated in Chapters 1 and 2. In effect, some scenarios may not need all semantic-based interoperability solutions or some other scenarios

may not be concerned with any internal adversary for instance. Our *DocWF* solutions comprise multiple mechanisms regarding interoperability and security as described in Chapters 4 and 6. The developed mechanisms when implemented together with the communication infrastructure of Chapter 5, provide enough flexibility and can be customized according to scenarios. Based on the mechanisms we introduced in Chapters 3, 4, 5 and 6, various customizations that we term as execution modes can possibly be supported by the document-based agile workflow infrastructure. These execution modes provide the modularity that makes it possible to deploy from a very lightweight *DocWF* system to a fully supported *DocWF* system. The execution modes are illustrated in Figure 7.1 and summarized below.

1. **Lightweight document-based agile workflow modeling**: In this mode, there is no support for the LTL representation of business rules and their finite state automaton of Chapter 3. Instead some static decision tables to represent business rules can be checked before and after achieving a goal. Such a lightweight document-based agile workflow modeling can be used when *DocWF* models and business rules are assumed to be correct and static respectively in the involved business scenario and therefore no modeling error may occur due to a faulty *DocWF* model.

2. **Lightweight document-based agile workflow infrastructure**: In this case, neither the semantic enabled document modeling with annotations of Chapter 4 nor the security mechanisms of Chapter 6 are implemented for the executions of a *DocWF*. The assumption is a business process scenario involves only trusted known peers and their data models are static. As such, XML schema-based data modeling and XML-structure-based security solutions as described in Chapter 2 can be used. However, the EBOL-based structural integrity solution needs to be deployed for document integrity protection as described in Chapter 6.

3. **Interoperable *DocWF* infrastructure without document comparisons**: The semantic enabled document modeling with document annotations is implemented and supported by the ontology driven communication infrastructure. In this mode, one portion of a composite document (i.e., a mapped document portion of a *concept*) is updated only once by a peer, resulting into only one version of a document. Thus, the document comparison solution of Chapter 4 is not required.

4. **Interoperable *DocWF* infrastructure with document comparisons**: The document comparison solution is implemented in addition to the previous mode.

5. **Lightweight decentralized communication infrastructure for a *DocWF* execution**: It considers a business process in a closed domain with an increasing number of peers. In this case, authorization decision delegation is required but the ontology-based document modeling of Chapter 4 and semantic enabled policy checking of Chapter 5 may not be required as peers are in a closed boundary and thus they may share their document mapping. Therefore, light distributors implementing only XML structure-based access control as mentioned in Chapter 2 would be sufficient.

6. **Decentralized communication infrastructure with semantic enabled policy checking**: As opposed to the previous mode the business process involves peers from other organizations and thus the semantic enabled policy checking is also implemented for distributors.

7. **Secure *DocWF* infrastructure without semantic verification**: This considers peers of various organizations with agreed XML structure based interfaces. Peers therefore do not require semantic annotations before publishing as the published XML node structure is agreed between peers. Therefore, in this mode, the semantic verification of receipt documents of Chapter 6 is not required.

	DocWF Model	Interoperability of *DocWFs*		Security of *DocWFs*				Communication Infrastructure of *DocWFs*
Execution Mode	LTL-based Business Rules for *DocWF* Models	Ontology-based Document Modeling	Document Comparison	Semantic Integrity	EBOL-based Structural Integrity	Document Containment	Document Traceability for A Posteriori Verification	Semantic Enabled Policy Checking
1. Lightweight document-based agile workflow modeling	×	✓	✓	✓	✓	✓	✓	✓
2. Lightweight document-based agile workflow infrastructure	✓	×	×	×	✓	✓	✓	×
3. Interoperable *DocWF* infrastructure without document comparisons	✓	✓	×	✓	✓	✓	✓	✓
4. Interoperable *DocWF* infrastructure with document comparisons	✓	✓	✓	✓	✓	✓	✓	✓
5. Lightweight decentralized communication infrastructure for a *DocWF* execution	✓	×	✓	×	✓	✓	✓	×
6. Decentralized communication infrastructure with semantic enabled policy checking	✓	✓	✓	✓	✓	✓	✓	✓
7. Secure *DocWF* infrastructure without semantic verification	✓	×	✓	×	✓	✓	✓	×
8. Secure *DocWF* infrastructure without structural verification	✓	✓	✓	✓	×	×	✓	✓
9. Secure *DocWF* infrastructure without document containment	✓	✓	✓	✓	✓	×	✓	✓
10. Secure *DocWF* infrastructure without traceability	✓	✓	✓	✓	✓	✓	×	✓
11. Secure *DocWF* infrastructure	✓	✓	✓	✓	✓	✓	✓	✓

✓ = implemented/enabled

× = not implemented/not enabled

Figure 7.1: Execution modes supported by the document-based agile workflow infrastructure.

8. **Secure *DocWF* infrastructure without structural verification**: In this mode, peers from various organizations exchange documents through a trusted communication infrastructure. As such, adversaries including malicious peers and distributors do not perform any malicious activities on encrypted nodes, for instance, updates, deletes, inserts, moves. For the same reason document containment is not required. However, semantic verification is still needed as to verify the receipt of XML nodes according to subscriptions.

9. **Secure *DocWF* infrastructure without document containment**: In this mode, peers and distributors are trusted to send document portions without any illegitimate pruning and as such, the document containment property may not be required.

10. **Secure *DocWF* infrastructure without traceability**: In this mode, the document access traceability mechanisms of Chapter 6 are not deployed assuming the scenario does not require a peer to trace any previous document access.

11. **Secure *DocWF* infrastructure**: The security mechanisms of Chapter 6 are fully implemented in peer sites and in the communication infrastructure nodes.

An execution mode can be selected based on the features required to implement an agile business process. For instance, in Figure 7.1, the execution mode of (5) can be selected when *DocWF* peers work in a closed community where peers agree on XML schema-based interfaces for data exchanges and their data models also do not change.

4 Perspectives

Finally, we give an overview of the possible lines of research that could be carried out based on the results presented in this thesis.

Business task determination during task enactment: The suitable business task determination process in the scope of this thesis is limited to a simple string matching of business goals and goal annotations of BPMN tasks for instance. It could integrate some complex match-making processes such as matching combination of goals, matching goals or business rules with pre- or post-conditions of a business task.

Modeling platform tool for Document-based agile workflows: The design of a *DocWF* model in the scope of this thesis is limited to a specification of the models in terms of goals, associated business rules and LTL-based rule transformation. As mentioned in Chapter 3, a modeling platform extending, for instance, the BPMN modeler of [EBP] for business users to hide the details of the LTL-based rule representation and their deadlocks and conflicts detection would be helpful. As a *DocWF* model is supposed to be communicated with the stakeholders and involved peers residing in varying business boundaries a web-based modeling platform like Oryx [ORY] would be appropriate. In that context, an extension of Oryx realizing the *DocWF* design time entities of Figure 3.2 (i.e., modeling *DocWF*s with goals and rules) can be implemented.

Service binding for a task enactment: In agile business scenarios like any pervasive environment, the availability of specific service instances cannot be guaranteed over time as actors move and services leave and join a collaboration. For tackling this issue, the task state modeling of Chapter 3 assumes that alternative services are semantically equivalent to a substitutable service which may not be true in a pervasive setting as shown in [FGIZ08, FGIZ]. The task state models are maintained by peers locally and thus may not always get the global view of suitable services. However, implementing the *DocWF* execution snap model, i.e., $DocWFexec$, of Chapter 3 as an annotation would provide a global view in terms of goals achieved and to be achieved. These limitations of the task state model may also be addressed by investigating how distributors can take over the substitution activities in order to get a global view of the tasks states. In that context, a semantic-based service substitution mechanisms can be sought in order to determine suitable services as in [FGIZ08] for instance.

Document-based agile workflow execution conformance: For critical business scenarios, conformance with a pre-specified sequence of tasks can be helpful for business actors in order to analyze, for instance, compliance regulations. As mentioned in Chapter 3, there exists no such specification for a *DocWF*, no verification of execution against such a plan is possible. Instead of such a deterministic assessment, a *DocWF* execution conformance rather can be goal directed and based on business rule enforcement verification which could be a complementary security solution to the Chapter 6.

Business rule conflict resolution at runtime: As mentioned in Chapter 3, the problem with a rule-based system is possible conflicts, in particular when rules are introduced by peers from various organizations. Conflicts may result into the violation of separation of duty constraints for instance. In a typical task-based workflow a conflict of interest amongst peers can be avoided by integrating complex security policy models at workflow design time and thus the separation of duty constraints can be satisfied [ACM01, CLW05, BN89]. However, such design time policy models are not sufficient for *DocWF*s as an exhaustive set of pre-defined business tasks may not exist in design time and as such further research is required.

Appendix A

Document distributor implementation for the communication infrastructure

We present in this appendix an implementation of the semantic enabled policy checking of a distributor that we developed as a proof of concept as specified in Chapter 5. The implementation work we pursued consists of the development of a core functionality (i.e., semantic enabled policy checking) of a distributor as designed in Figure 5.8 of Chapter 5. The main goal of this work is to show a proof of concept of fine grained document authorizations (i.e., Fine grained document access for a *concept* and access control decision delegation) of Chapter 6 leveraging the decentralized communication infrastructure of Chapter 5.

1 Distributor service and UML [UML] diagrams

We first outline the overall classes that are implemented in order to realize the semantic enabled policy checking of a distributor. Interaction between those classes are then illustrated in a sequence diagram.

1.1 Class and sequence diagrams

Figure A.1 depicts the distributor class diagram. The central class of the distributor is *DistributorService*. The *DistributorService* class coordinates with other classes to build the business logic of a distributor. This is described as follows:

- *DistributorService* class instantiates other classes (i.e., *DomainOntology*, *PolicyOntology*, *SPARQLengine*, *XacmlToSPARQL*, *NodeRouter* and *NodeDeliverer*).

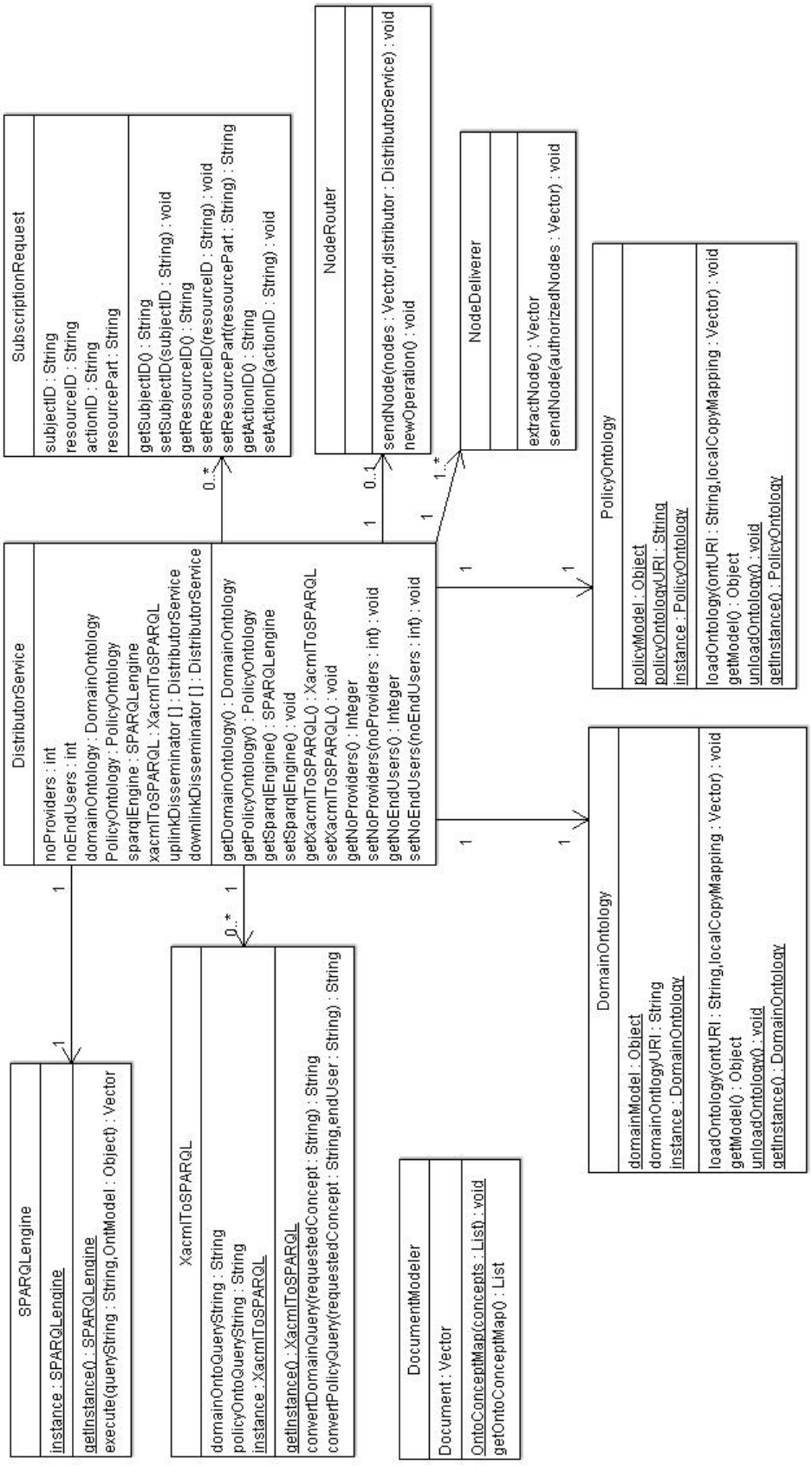


Figure A.1: Ontology-based fine grained XML distribution: Class diagram.

- An XACML request from a peer for a business domain *concept* is represented by the *SubscriptionRequest* class. The semantic enabled policy checking for a *concept* for a given business peer is initiated upon receipt of a *SubscriptionRequest*.
- *DistributorService* class forwards the request to the *XacmlToSPARQL* class which then converts the request into an SPARQL query over the *DomainOntology*.
- *SPARQLengine* class interprets this query using Joseki [JOS] apis.
- The result of the query of the previous step is then applied over the *PolicyOntology* using the *SPARQLengine* class as described in Chapter 5.
- Based on the result of the previous step legitimate XML nodes are delivered to the authorized peers through the *NodeDeliverer*.

A distributor also performs selective routing of fine grained documents as described in Chapter 5 which would be realized by the *NodeRouter* class. However, in this implementation as mentioned above we focus on the semantic enabled policy checking of a distributor. Figure A.2 further depicts the above interactions in a sequence diagram.

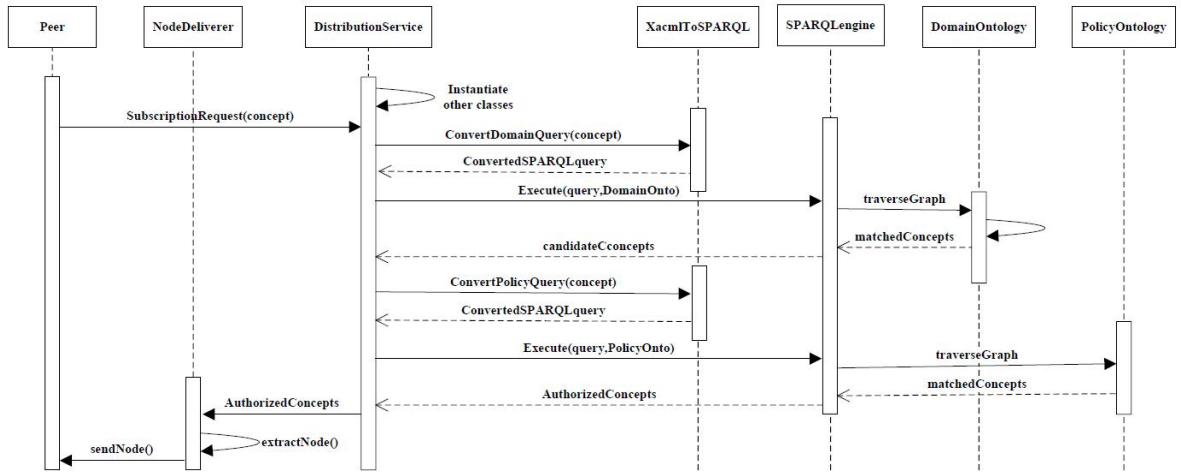


Figure A.2: Interaction of the classes of Figure A.1.

2 Document distribution visualization tool

In order to visualize document publishing and subscriptions by a peer and business logic of a distributor, we developed a graphical tool leveraging above mentioned classes that are controlled by several JSP [JSP] pages. The JSP pages get inputs from some HTML pages. In the following, first we discuss the principles of the tool and followed by an illustration of interactions of JSPs and HTMLs.

The principles of the visualization application are depicted in Figure A.3. The visualization application is composed of the three following components:

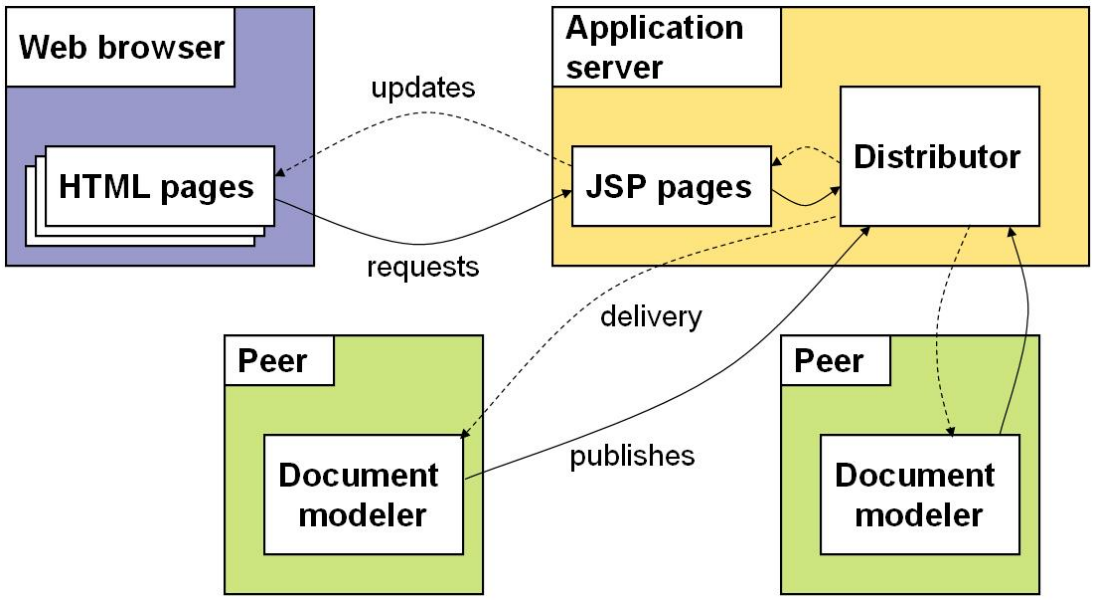


Figure A.3: Document distribution visualization application principles.

- **Application server**: The application server is the core of the visualization application. It hosts the distributor logic as specified in section 1 and the JSP pages (i.e., *subscription.jsp*, *distributorProcess.jsp*, *receive.jsp*, *provider.jsp* and *map.jsp*). Some JSP pages forward *concept* requests to the distributor which in turn replies by sending dynamic HTML pages as updates through some other JSPs (detailed shortly).
- **Document modeler**: The document modeler will allow a peer to specify the mapping document portions to *concept*s as specified in Chapter 4, performs document publishing and receiving of documents to and from distributors (see Figure A.3). Intuitively, the document modeler is hosted in a peer site.
- **Web browser**: This is the client side that visualizes the events during document dissemination using several HTML pages (i.e., *enduser.html* and *provider.html*). The HTML pages for a recipient peer shows the last received XML nodes. This can be done in dynamic fashion by using, for instance, Ajax script [AJA] in the peer site so that pages can be updated in regular interval. In order to simulate the implemented functionalities of the visualization there are separated HTML pages for a distributor and publishing peers.

Figure A.4 shows the client side main screen shot of the visualization tool.

At runtime, the provider peers involved in a *DocWF* execution use HTML forms in order to publish updated XML nodes to the distributors. Based on the providers policies distributors filter nodes before sending those to legitimate peers through JSP pages. The subscribing peers can also visualize their received XML nodes in HTML pages. This is illustrated in the following.

1. *subscription.jsp*: It receives subscription requests from the *enduser.html* and forwards the request to the *distributorProcess.jsp*.

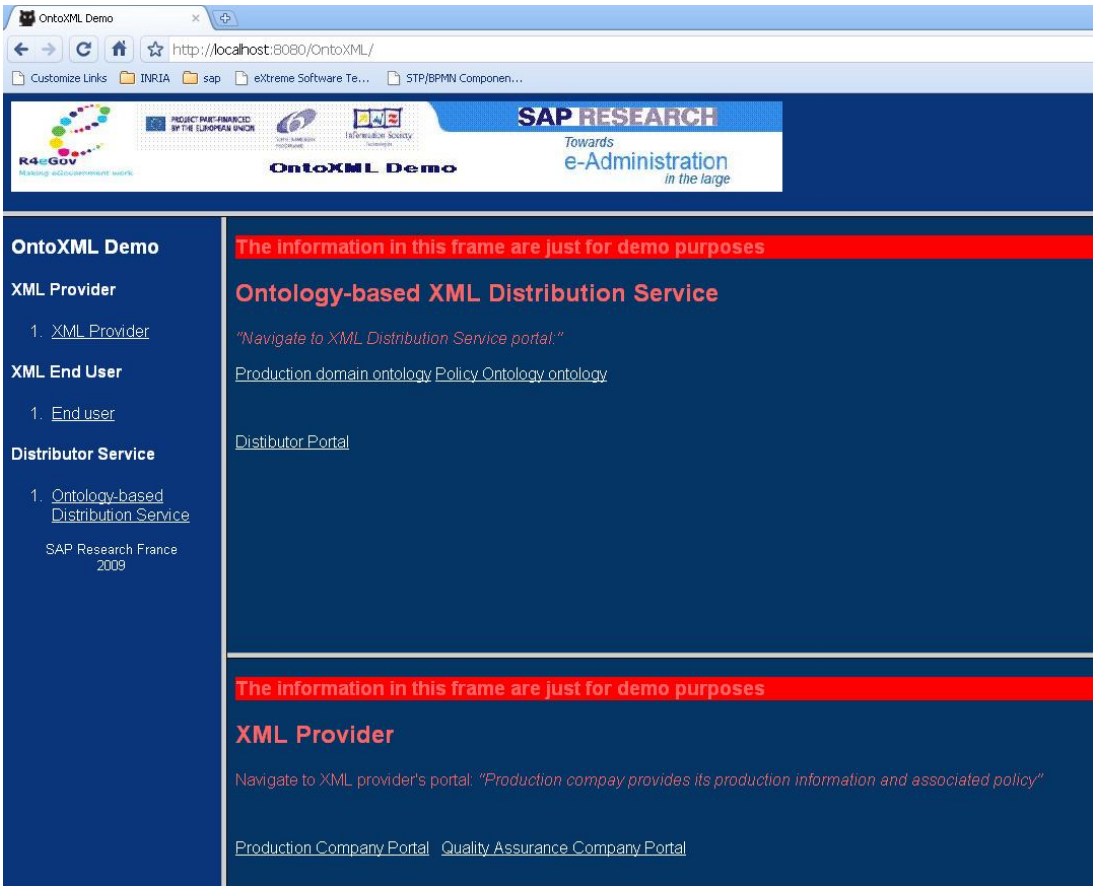


Figure A.4: Document distribution visualization tool main screen.

2. *distributorProcess.jsp*: It invokes the business logic of a distributor and sends back the fine grained documents to *receive.jsp*.

3. *receive.jsp*: It receives fine grained documents from the *distributorProcess.jsp* and shows those to the peers.

4. *provider.jsp*: A provider peer selects its profile in the *provider.html* and publishes its documents through the *provider.jsp*.

5. *map.jsp*: It gets the documents of the provider to be associated with the domain ontology *concept*s from the *provider.jsp* and provides an interface to the provider to perform the mapping.

3 Deployment

The implementation presented in this appendix has been developed based on the following technologies:

- **J2EE**: The prototype has been implemented in Java.

- **Tomcat**: Application server [TOM].
- **JOSEKI**: XPARQL engine interpreting SPARQL queries [SPQ].
- **Sun's XACML**: XACML engine interpreting XACML request [XAC].
- **JSP**: Java server pages [JSP].

Appendix B

Security library implementation for the document protection

We present in this appendix the implementation of the security mechanisms for document protection specified in Chapter 6 to secure an execution of a *DocWF*. The implemented document protection is based on the document envelope of Chapter 6 (see Figure 6.13). This implementation is bundled into a standalone demonstrator demonstrating the cross border crime scenario of Chapter 6. This demonstrator was part of the advance security work package deliverable of the EU IST r4egov project [R4E] and was called "Distributed Access Control Framework for Document Centric Collaboration". The novelties of this implementation are the Java libraries for the adapted TGDH and document containment (i.e., merkle hashing) as specified in Chapter 6. In what follows, the description of the demonstrator in terms of implemented Java packages and several screen shots depicting few interactions of the scenario. Then we describe the libraries that we have implemented according to the specification of Chapter 6.

1 Distributed access control framework for document centric collaboration and UML [UML] diagrams

We first outline the overall packages that are implemented in order to realize the demonstrator. The relations amongst those packages are then depicted.

1.1 Demonstrator packages

Figure B.1 depicts the full package diagrams of the demonstrator.

- **CollaborationsManager:** It manages collaborations between peers by maintaining a global collaboration context.

- **KeyManager:** It is the entry class for the key management (i.e., compute, recompute).
- **Listeners:** They dispatch requests and documents to appropriate peers.
- **DeligationManager:** It implements the delegation functionality of a peer.
- **TGDH:** It represents the implementation of the adapted TGDH (Tree-based group Diffie-hellman) protocol for multiple peers. It is further described in Section 2.
- **MerkleHash:** It is the Merkle hashing implementation of "Enterprise XML". It is further described in Section 3.

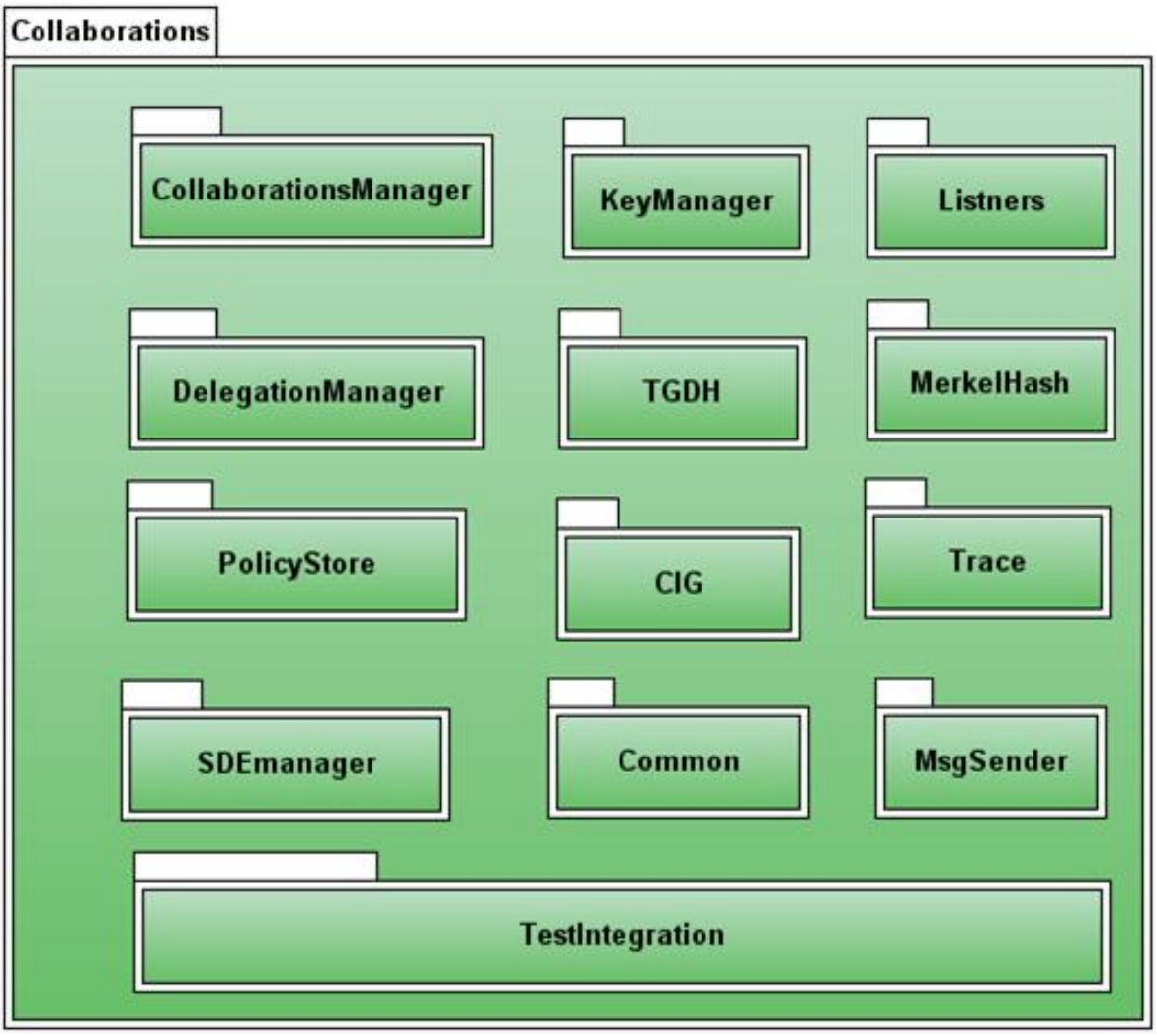


Figure B.1: Distributed XML access control framework: package diagram

- **PolicyStore:** It represents the policy store of a peer.
- **CIG:** It implements the common access interest group specification of Chapter 6.
- **Trace:** It represents a document access trace as specified in Chapter 6.

- **SDEmanager:** This is document envelope manager that builds the complete document envelope as specified in in Chapter 6.
- **MsgSender:** This is responsible for publishing a document envelope.
- **TestIntegration:** This package contains the test classes.

Some packages in Figure B.1 are not implemented so far as these are not important for a proof of concept of the adapted TGDH and Merkle hashing for enterprise XML. These includes 'DeligationManager', 'PolicyStore','MsgSender', and 'Common'. Figure B.2 shows interactions amongst the packages.

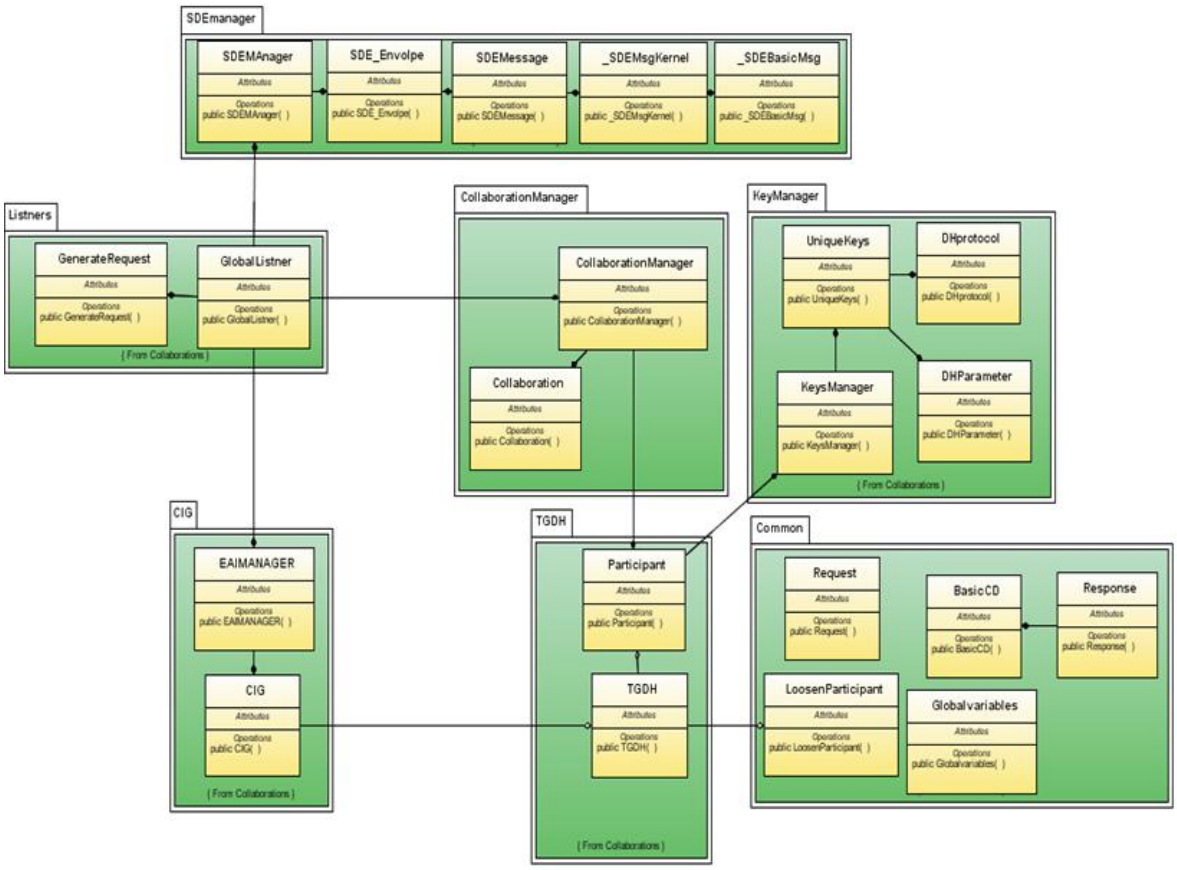


Figure B.2: Distributed XML access control framework: package interactions.

2 Adapted TGDH library for independent key computation by a peer

The adapted TGDH library is implemented by the following Java classes as depicted in Figure B.3:

- *TGDHlinkedList*: This is the main class of this library. The logical binary key tree of the adapted

TGDH is represented as a linked list. Thus adding/leaving a peer node is simply a manipulation of the pointers as opposed to manipulating the whole tree as in a classical TGDH.

Figure B.3: The adapted TGDH package.

- *Participant*: This class represents a peer in the logical tree of the adapted TGDH. It maintains the Diffie Hellman key pairs associated to the peer's subscription of *concept*s. It also keeps track of its TEK and TES associated to its key path and sibling path as specified in Chapter 6.
- *DHprotocol*: This class represents the Diffie Hellman protocol which is used by a peer to generate the Diffie Hellman key pairs in its key path.
- *DHParameter*: It represents the parameters, for instance, prime number needed to generate the Diffie Hellman keys.

3 Merkle hash library for document containment of "Enterprise XML"

The implementation of generating Merkle hashing over an XML document and its verification are depicted in Figure B.4 and illustrated in the following:

1. **Canonicalize:** Canonicalize the document for which a Merkle hash needs to be generated.
2. **Copath determination:** Determine the hash values of the pruned portion of the document (i.e., determining the copath).
3. **Hash computation:** Generate a hash value of the concatenated values of the root node of the legitimate portion and the copath. This is a recursive process as hashing is performed in a bottom up fashion over an XML document.
4. **Verification:** Perform previous two steps out of the received document portions. If the computed hash matches with the existing one then the received pruned document portion is originated from the actual composite document.

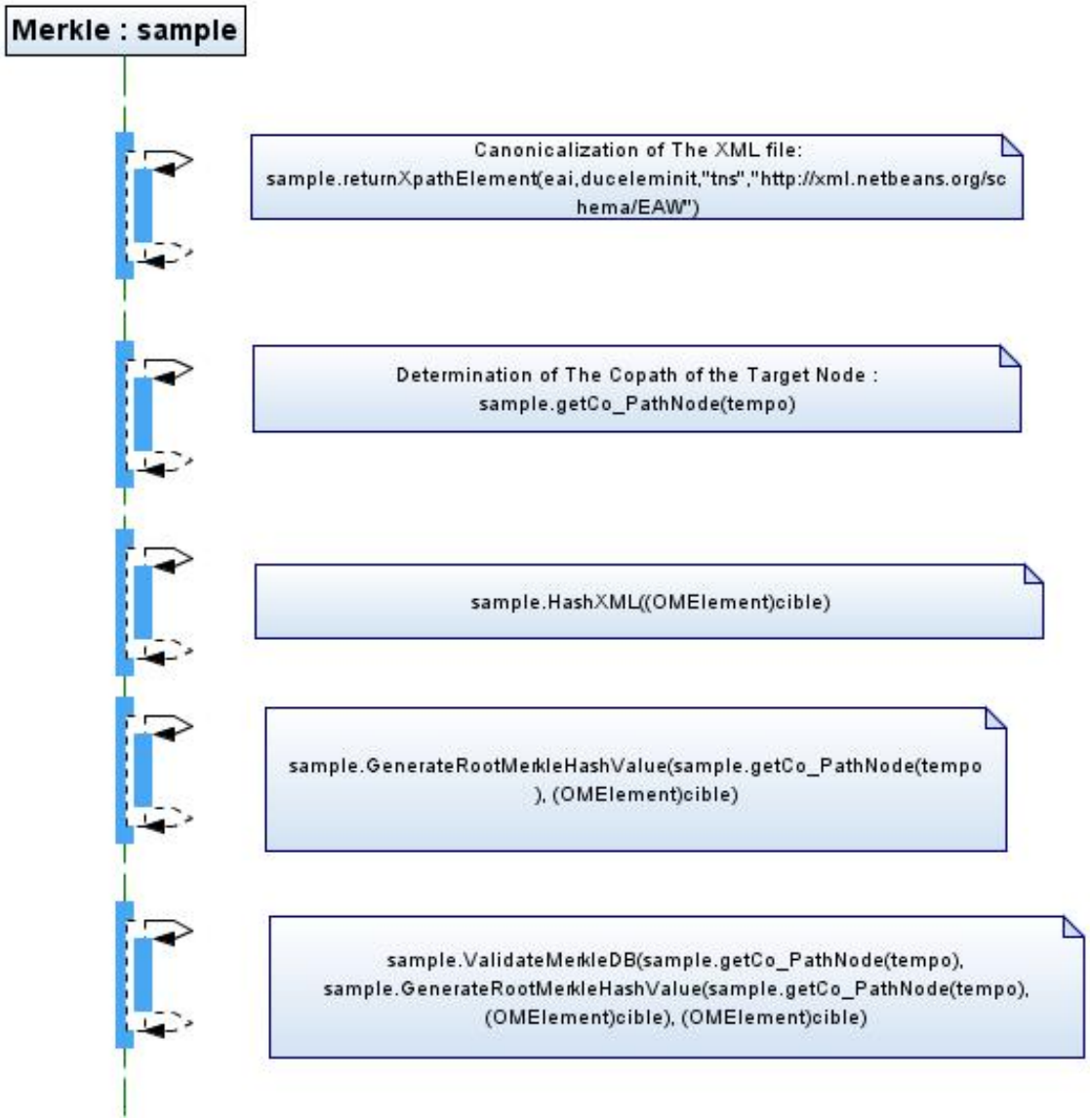


Figure B.4: Merkle hash implementation for "Enterprise XML".

4 Commandline Demonstrator

In this section, various screenshots, i.e., Figures B.5,B.6,B.7,B.8,B.9,B.10 based on eclipse IDE are shown illustrating an execution of the *DocWF* demonstrator. For demonstration purpose it is assumed that ontology *concept*s are already mapped to document portions. The execution is depicted in the two following phases:

- **Interest specification phase**: In this phase a distributor collects multiple subscription requests and initiate the computation of common interest groups as specified in Chapter 6.
- **Collaboration phase**: In this phase peers edit documents, verify document integrity, traces of

received documents as specified in Chapter 6.

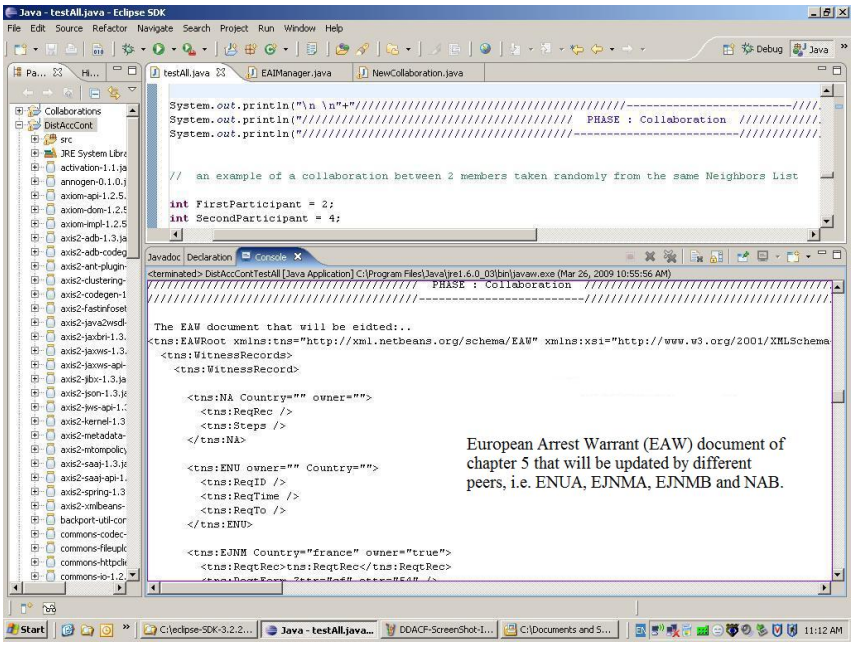


Figure B.5: Building the European Arrest Warrent (EAW) document by ENU A, EJNM A, EJNM B and NA B.

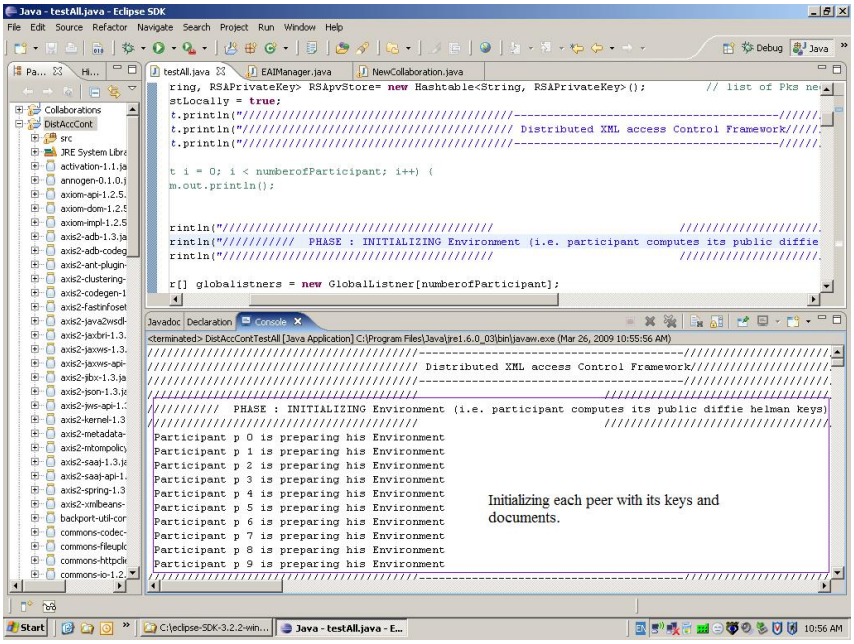


Figure B.6: Initialization of peers.

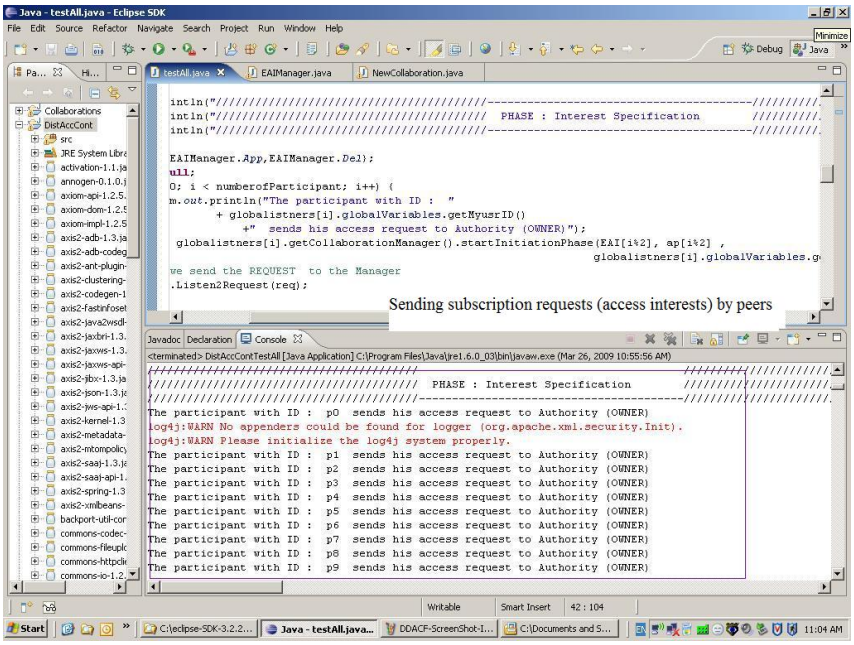


Figure B.7: Subscription requests of peers.

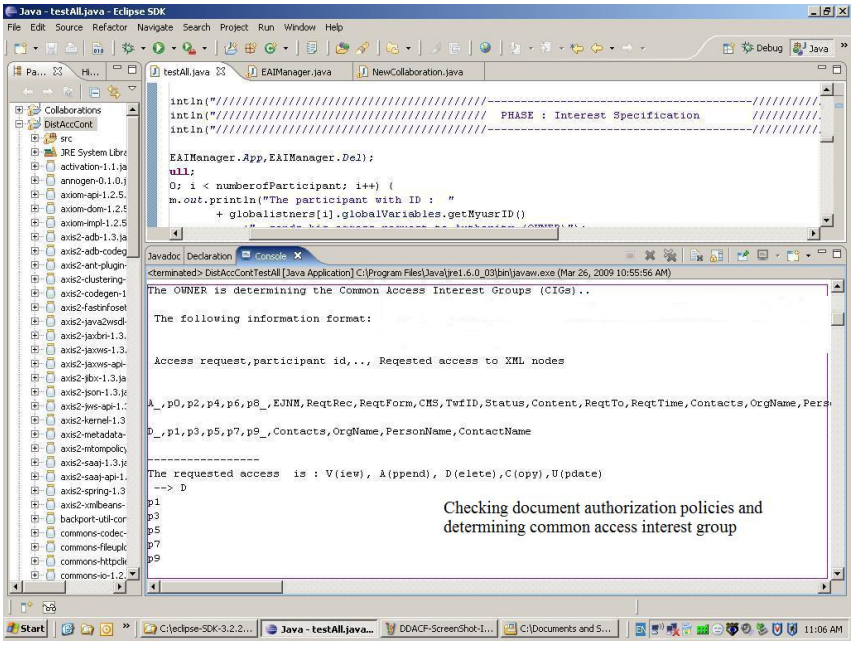


Figure B.8: Checking authorization policies and determining a common access interest group (CIG) by a distributor.

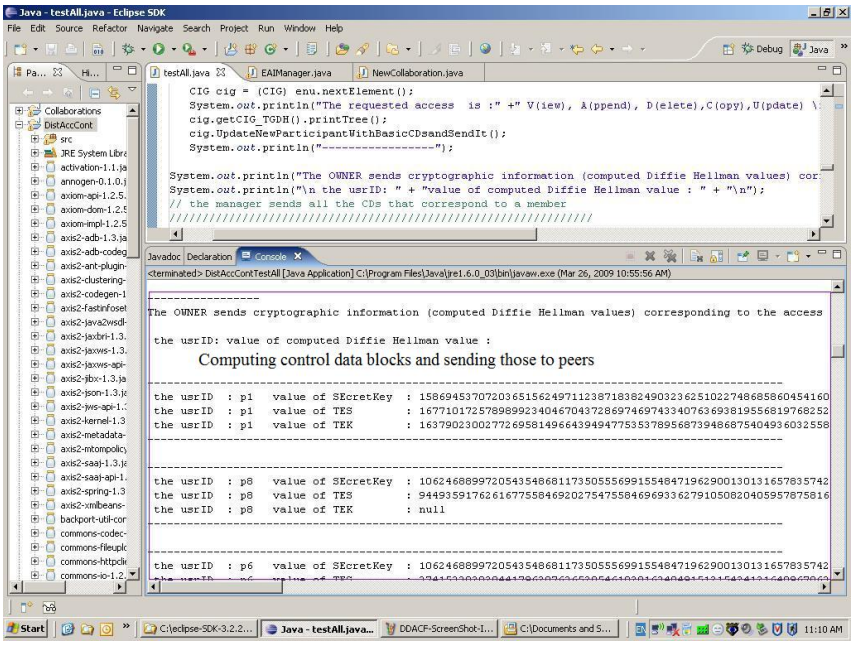


Figure B.9: Computing control data blocks and sending them to peers by a distributor.

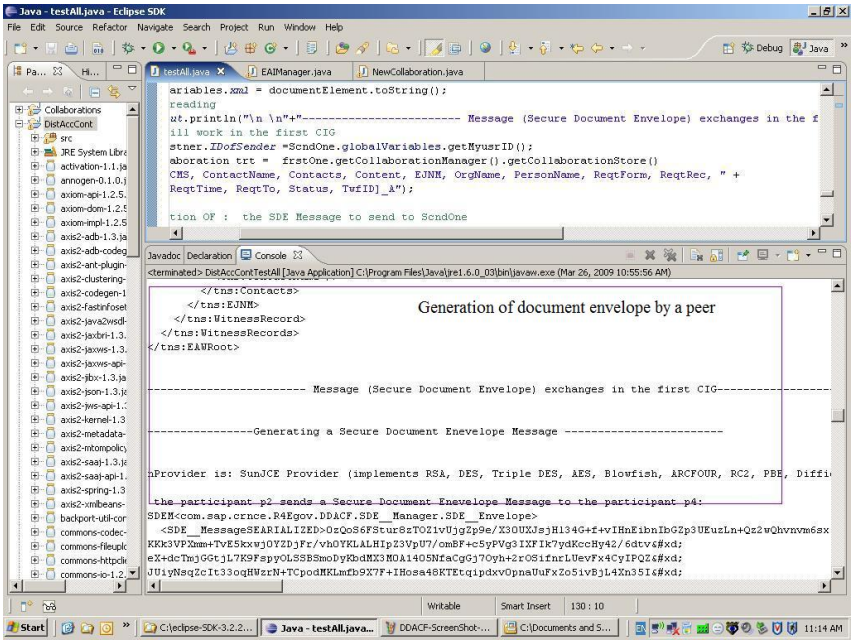


Figure B.10: Generating a document envelope by a peer.

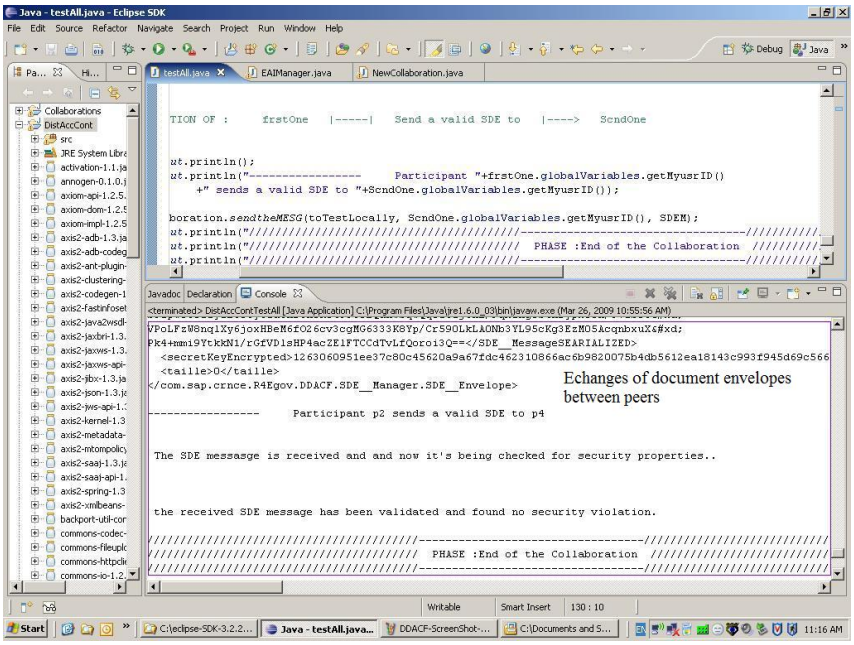


Figure B.11: An exchange of a document envelope between two peers.

5 Demonstrator visualization tool

In order to visualize the demonstrator, we developed a graphical UI tool using Openlaszlo [OPEa] framework.

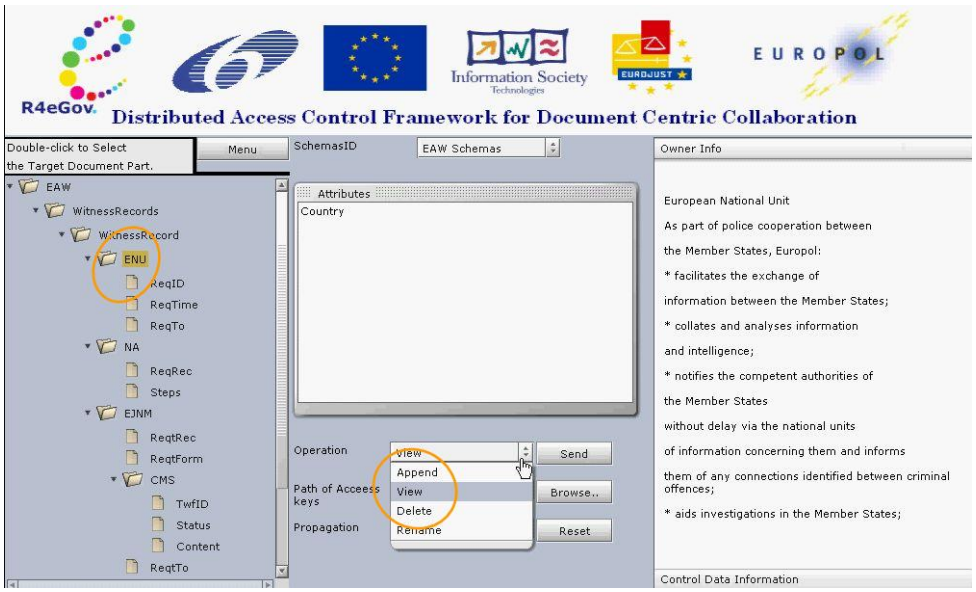


Figure B.12: Requesting for view access over the <ENU> document portion by a peer.

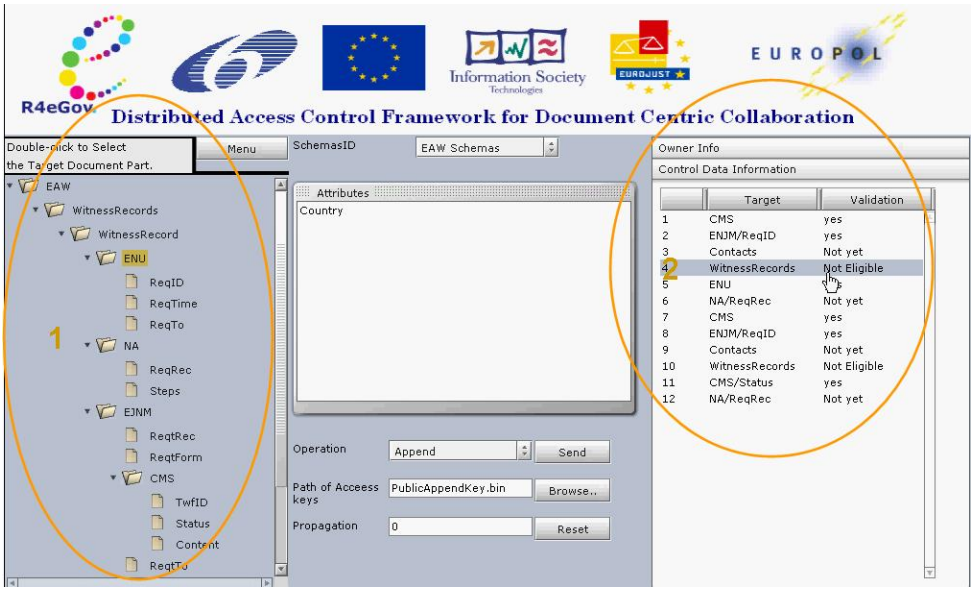


Figure B.13: (1) The European Arrest Warrant (EAW) document tree of Chapter 6. (2) Fine grained access rights of the peer of Figure B.12 over different XML nodes of the EAW document as determined by a distributor.

6 Deployment

The implementation presented in this appendix has been developed based on the following technologies:

- **J2EE**: The prototype has been implemented in Java.
- **JCE**: Java cryptographic extension apis for basic Diffie Hellman protocol.
- **Adapted TGDH**: Adapted TGDH library as described in section 2.
- **Merkle hash for XML**: Merkle hash for XML as described in section 3.
- **Openlaszlo**: User interfaces for peers are designed using Openlaszlo [OPEa] framework.

Appendix C

Illustration of a formal *DocWF* execution snap

We illustrate in this appendix a formal *DocWF* execution snap by describing the transition rules based on the task state model specified in Chapter 3. We also illustrate the basic principles of a *DocWF* execution conformance checking as mentioned in the perspectives.

1 *DocWF Status* Transition Rules

Any chosen task (i.e., recipe or free) has an initial state value of 0. Let T_i be a potential task and there is no task T_k such that $T_i > T_k$ then the state value of T_i is $S_a(T_i) = 1$ (Rule A). If the new status resulted from a successful execution of T_i is S_b, then the execution of T_i is denoted by $S_a(T_i)S_b$. This implies $S_b(T_i) \in \{2, 3\}$ (Rule B).

Now, $\forall T_j$ (i.e., potential tasks) such that $T_j > T_i$, the state value of T_j at *DocWF* status S_b is determined by (1) If $T_j = T_i$ then $f_{ij} = 1$ and the state value of T_j at new status is: $S_b(T_j) = 3$; (Rule C) (2) If $T_j \neq T_i$ then the state value of T_j at new *DocWF* status S_b depends on the state value of T_i at the status S_b. There are four possibilities depending on the policy violation and feasible condition:

- Rule I - $S_b(T_i) = 2$ and $v_{ij} = v_{ji} = 1$:
  (a) If $\exists C' \in C(T_j)$ such that $S_b(T_k) \in \{2, 3\}$ for any $T_k \in C'$, then $f_{ij} = 1$ and $S_b(T_j) = 1$;
  (b) Else if $\exists k, j\ T_{k=j} \in C(T_j)$ such that $S_b(T_{k=j}) = 2$ then $f_{ij} = 1$ and $S_b(T_j) = 2$;
  (c) Else $f_{ij} = 0$ and $S_b(T_j) = 0$.

- Rule II - $S_b(T_i) = 2$ and $v_{ij} = v_{ji} = 0$:
  (a) If $S_a(T_j) = 1$ then $S_b(T_j) = 2$;
  (b) Else if $\exists k, j\ T_{k=j} \in C(T_j)$ such that $S_b(T_{k=j}) = 2$ then $f_{ij} = 1$ and $S_b(T_j) = 3$;
  (c) Else $f_{ij} = 1$ and $S_b(T_j) = 1$.

- Rule III - $S_b(T_i) = 3$ and $v_{ij} = v_{ji} = 1$:
  (a) If $\exists C' \in C(T_j)$ such that $S_b(T_k) \in \{2, 3\}$ for any $T_k \in C'$, then $f_{ij} = 1$ and $S_b(T_j) = 1$;
  (b) Else if $\exists k, j\ T_{k=j} \in C(T_j)$ such that $S_b(T_{k=j}) = 2$ then $f_{ij} = 1$ and $S_b(T_j) = 2$;
  (c) Else $f_{ij} = 0$ and $S_b(T_j) = 0$.

- Rule IV - $S_b(T_i) = 3$ and $v_{ij} = v_{ji} = 0$:
  (a) If $S_a(T_j) = 1$ then $S_b(T_j) = 2$;
  (b) Else if $\exists k, j\ T_{k=j} \in C(T_j)$ such that $S_b(T_{k=j}) = 2$ then $f_{ij} = 1$ and $S_b(T_j) = 3$;
  (c) Else $f_{ij} = 1$ and $S_b(T_j) = 1$.

According to the above transition rules, a potential task T_j's state value at a new *DocWF status* S_b is 0 iff one of the following holds:

- T_j is just chosen as a recipe or a free task.
- If the state value of the task T_i in S_b is 2, and the violation condition $v_{ij} = v_{ji}$ is 1; meaning the associated documents/document portions can not be handled as feasible condition is false in the current *DocWF status* (Rule I(c)).
- If the state value of the task T_i in S_b is 3, and the violation condition v_{ij} is 1; meaning the associated document/document portion can not be handled as feasible condition is false in the current *DocWF status* (Rule III(c)).

A potential task T_j's state value at a new *DocWF status* S_b is 1 iff one of the following holds:

- If the state value of the task T_j in S_a was 0; meaning it is ready to be executed. (Rule A)
- If the state value of the task T_i in S_b is 2, and the policy violation condition v_{ij} is 0; meaning the associated document/document portion can be handled immediately (Rule II (c)).
- If the state value of the task T_i in S_b is 2, and the policy violation condition $v_{ij} = v_{ji}$ is 1 and at least all tasks in one of T_j's feasible condition sets is successfully executed; meaning if some of the preceding goals are achieved if not all then a subsequent document/document portion can be handled for an enabled goal (Rule I(a)).
- If the state value of the task T_i in S_b is 3, and the policy violation condition v_{ij} is 0; meaning the associated document/document portion can be handled immediately (Rule IV (c)).
- If the state value of the task T_i in S_b is 3, and the policy violation condition $v_{ij} = v_{ji}$ is 1 and at least all tasks in one of T_j's feasible condition sets is successfully executed; meaning if some of the preceding goals are achieved if not all then a subsequent documents/document portions can be handled for an enabled goal (Rule III(a)).

A potential task T_j's state value at a new *DocWF status* S_b is 2 iff one of the following holds:

- If the state value of the task T_i in S_b is 1, and the goal is achieved after its execution; meaning associated document/document portions are handled completely (Rule B).

- If the state value of the task T_i in S_b is 2 or 3, and the policy violation condition $v_{ij} = v_{ji}$ is 1 but the task T_j is executed before; implies the goal is achieved by handling associated documents/document portions (Rule I(b) and Rule III(b) respectively).
- If the state value of the task T_i in S_b is 2 or 3, and the policy violation condition v_{ij} is 0 and previously the state value of T_j was 0; meaning the execution of T_j should handle the associated documents/document portions and the goal should be achieved (Rule II(a) and IV(a) respectively).

A potential task T_j's state value at a new *DocWF status* S_b is 3 if the following holds:

- If the state value of the task T_i in S_b is 1, and the goal is not achieved yet after its execution; meaning associated document/document portions are handled partially and the same task may need to be executed later (Rule B).
- If the state value of the task T_i in S_b is 2, and the policy violation condition $v_{ij=v_{ji}}$ is 1 and the task T_j is executed before; but the goal is not achieved yet; implies the same task may need to be executed later to handle associated document/document portions (Rule II(b)).
- If the state value of the task T_i in S_b is 3, and the policy violation condition v_{ij} is 0 and the task T_j is executed before; but the goal is not achieved yet; implies the same task may need to be executed later to handle associated document/document portions (Rule IV(b)).
- If the same task T_i is potentially executable leading to a self loop (Rule C).

The transition rules take uncertainty into consideration during *DocWF* execution to decide upon a potential task to execute. For example, in the electronic health care record generation workflow of Figure 3.5 it is possible to perform additional diagnosis tests, i.e., T_4 while doctors are doing treatment, i.e., T_5; even doctors can postpone the treatment and asks for additional diagnosis tests for further treatment (explained in the following).

2 Execution Conformance for a *DocWF*

Recall from Chapter 3, the *DocWF* model-based verification allows an actor to detect design and run time errors of models formulated as deadlocks and conflicts. As opposed to checking the models, a peer may check an execution snapshot as specified as $DocWFexec$ tuple in Chapter 3. Another way is to check the affects of business rule enforcement on documents. These two mechanisms are illustrated below:

- **Verifying execution snap shot:** For any violation of a rule in the $DocWFexec$ implies an execution violation. So, ideally for no rule violation detection of the final $DocWFexec$ tuple implies that the complete execution of the *DocWF* model is conformed to the business rules associated to the goals.
- **Verifying business rule enforcement:** As business rules are enforced by updating document portions associated to authorized concepts by peers, verifying document access by peers during an execution.

Regarding execution snap shot verification, the $DocWFexec$ tuple computed by a peer must be made public so as to be verifiable by a peer. For verifying business rule enforcement, a peer first needs to identify business ontology concepts that are related to the business rules it wants to verify. Based on *DocWF* modeling technique of section 5 of Chapter 3, the related concepts can be identified by LTL operands representing concepts of a business rule. The peer must have subscriptions to receive mapped document portions of the concepts. Then actual verification must be performed on the business ontology level (i.e., semantic) which can be further rigorously performed on the document structure level. These are described in Chapter 6 as part of document protection.

Appendix D

Curriculum Vitae

Mohammad Ashiqur Rahaman
5, Square du Retiro
78150 Le Chesnay, France

Phone: +33 (0)1 39 63 58 01
Mobile: +33 (0)6 23 14 91 74
rahaman.mohammadashiqur@gmail.com

Date of birth: December, 13^{th} 1978
Citizenship: Bangladesh

Education

- Sept. 2006 - March. 2010. **Ph.D. in Computer Science**, Telecom Paris, Paris. joint programme with Institut Eurecom, Sophia-Antipolis, France.
  Supervisor: Pr. Yves Roudier, Institut Eurecom.
  Thesis contributions: design of a document-based agile workflow system, design of solutions to provide interoperability and security within a document-based agile workflow execution.

- Sept. 2004 - Sept. 2005. **M.Sc IT with specialization in Software Engineering of Distributed Systems**, Royal Institute of Technology (KTH), Stockholm, Sweden.
  Relevant courses: Distributed Systems, Programming Web Services, Java Network Programming, Software Engineering, Formal Methods.

- Oct. 1997 - Feb. 2003. **B.Sc in Computer Science**, University of Dhaka, Bangladesh.
  Relevant courses: Computer Architecture, Digital Systems, Data structure and Algorithms, C/C++, Discrete Mathematics.

Research Experience

- Sept. 2006 - Aug. 2009. **Research Associate at SAP Labs France SAS** Sophia-Antipolis, France.
  I was part of the Security and Trust Research program at SAP Research, as a PhD fellow funded by the French CIFRE programme. My research activities focused on topics related to the execution of document-based decentralized workflows including architecture design, interoperability and security.

- Mar. 2004 - Aug. 2004. **Master's Student, Intern at SAP Labs France SAS** Sophia-Antipolis, France.
  I wrote my Master's thesis during first six-month of eleven month internship within the security and trust research group at SAP on the topic "Web Services Security and XML Re-writing Attacks". I was involved in a EU IST project called MOBIUS. In the scope of this Master thesis I designed an algorithm for detecting XML re-writing attacks against SOAP messages. The result of this work was implemented as an extension WS-security plugin for a SAP Research web service test-bed tool called "JOHNSON".

Research Projects

- From Sept. 2009. **ITEmIS**: *IT and Embedded Integrated Systems*. French ANR project involving seven organizations. ITEmIS aims at easing the evolution from today's world of separate lightweight embedded applications and IT services to the future world of seamlessly integrated services, thus qualifying and defining a new generation SOA enabling IT and Embedded Integrated Systems. I am involved in the designing the architecture of ITEmIS accommodating various architectural paradigm such as SOA, SCA and JBI.
  http://itemis-anr.org/

- Sept. 2006 - Aug. 2009. **R4eGov**: *Towards e-Administration in the large*. EU IP project involving twenty institutions. This project consists in designing a secure collaborative framework supporting the execution of business processes involving different European institutions. This project capitalizes on the results of my thesis with respect to fine grained document access control and document protection solutions.
  http://www.r4egov.info/

- Oct. 2005 - Aug. 2006. **MOBIUS**: *Mobility, Ubiquity and Security*. EU IP project involving eighteen institutions. This project aims at giving users independent guarantees of the safety and security of Java applications for their mobile phones and PDAs. I was involved in this project as master student doing formal verification of XML re-writing attack detections.
  http://mobius.inria.fr/

Academic Teaching and Supervision

- Teaching courses on Data structure, Automata Theory, C programming, Digital Systems and Discrete mathematics at Daffodil International University, Bangladesh

- Supervision of one master's students at SAP Labs France.

Software Development Activities

- 2006-2009. In the scope of my PhD thesis, I implemented two prototypes presenting my research results. The first prototype (i.e. document distributor) is an implementation of the ontology-driven communication infrastructure. This supports a document-based agile workflow execution by routing and delivering of fine grained documents based on document provider's policy to legitimate peers. The other prototype implements two Java security libraries for the adapted TGDH solution and Merkle hashing mechanism for enterprise XML documents respectively. The adapted TGDH implements the independent group secret key computation and the Merkle hashing for enterprise XML documents supports the mechanisms of document access traceability. These development tasks used the following technologies: Java / Tomcat, Joseki, Sun's XACML implementation, JSP, JCE: Java cryptographic extension apis for basic Diffie Hellman protocol.

- 2007-2008. In the scope of the **R4eGov** project, I implemented a Java log web service as part of a messaging framework (e.g. ESB) to record SOAP message exchanges between interacting parties (Law and enforcement authorities of EU countries).

- 2005. In the scope of my master's thesis, I extended a SOAP messaging framework by implementing WS-Security features based on Java. The extended framework demonstrates XML rewriting attacks against SOAP messages and my solution "SOAP Account" for early detection of such attacks.

Work Experiences

- From Sept. 2009. **Expert Engineer at INRIA-Paris** France.
  I am involved in a French ANR project called **ITEmIS** in which i am designing the architecture for the ITEmIS system.

- Sept. 2006 - Aug. 2009. **Research Associate at SAP Labs France SAS** Sophia-Antipolis, France.
  I was mainly involved in a European IST Research project namely R4eGov to which I contributed with my PhD thesis results. My work also involved presentations to SAP internal and external events as well as in contributions in the intellectual property portfolio of SAP by innovative patent applications.

- Oct. 2005 - Aug. 2006. **Intern at SAP Labs France SAS** Sophia-Antipolis, France.
  Eleven-month internship in the corporate research group at SAP.

- Mar. 2003 - Jul. 2004. **Lecturer at Daffodil International University** Dhaka, Bangladesh.
  Teaching computer science courses in undergraduate level and conducting labs.

Professional Certificates

- IBM Certified XML Solution Developer.
- IBM Certified SOA Associate.

Conference Papers

"**Towards Secure Content Based Dissemination of XML Documents**", Rahaman, Mohammad Ashiqur and Roudier, Yves and Schaad, Andreas, in the proceedings of the Fifth International Conference on Information Assurance and Security (IAS 09), August 18-20, 2009, X'ian, China.

"**A Secure Comparison Technique for Tree Structured Data**", Rahaman, Mohammad Ashiqur and Roudier, Yves and Schaad, Andreas, The Fourth International Conference on Internet and Web Applications and Services (ICIW 2009), May 24-28, 2009 - Venice/Mestre, Italy.

"**Ontology-based Secure XML Content Distribution**", Rahaman, Mohammad Ashiqur and Roudier, Yves and Schaad, Andreas, IFIP SEC 2009, 24th International Information Security Conference, May 18-20, 2009, Pafos, Cyprus.

"**Distributed Access Control For XML Document Centric Collaborations**", Rahaman, Mohammad Ashiqur and Roudier, Yves and Schaad, Andreas, EDOC '08: Proceedings of the 2008 12th International IEEE Enterprise Distributed Object Computing Conference.

"**SOAP-based Secure Conversation and Collaboration**", Rahaman, Mohammad Ashiqur and Schaad, Andreas, ICWS '07: Proceedings of the International Conference on Web Services 2007, IEEE Computer Society.

"**Towards Secure SOAP Message Exchange in a SOA**", Rahaman, Mohammad Ashiqur and Schaad, Andreas and Marteen, Rits, SWS '06: Proceedings of the 3rd ACM workshop on Secure web services.

"**An Inline Approach for Secure SOAP Requests and Early Validation**", Rahaman, Mohammad Ashiqur and Marten, Rits and Schaad, Andreas, OWASP 2006. European Conference on Open Web Application Security Project. May 30-31, 2006, Leuven, Belgium.

Journal Papers

"**A Publish/Subscribe Model for Secure Content-Driven XML Dissemination**", Rahaman, Mohammad Ashiqur and Roudier, Yves and Schaad, Andreas, in the special issue of Journal of Information Assurance and Security (JIAS), volume 6, http://www.mirlabs.org/jias/.

Workshop Papers

"**XML Secure Views Using Semantic Access Control**", Rota, Andrea and Short, Stuart and Rahaman, Mohammad Ashiqur, 1st International Workshop on Data Semantics (DataSem 2010), in cooperation with the Journal on Data Semantics, March 22, 2010 - Lausanne, Switzerland.

"**Document-based Dynamic Workflows: Towards Flexible and Stateful Services**", Rahaman, Mohammad Ashiqur and Roudier, Yves and Schaad, Andreas, in the proceedings of SCC 2009, 2009 IEEE International Conference on Services Computing, September 21-25, 2009, Bangalore, India.

"**Towards Secure SOAP Message Exchange in a SOA**", Rahaman, Mohammad Ashiqur and Schaad, Andreas and Marteen, Rits, SWS '06: Proceedings of the 3rd ACM workshop on Secure web services.

Patents (Filed by SAP)

- 2009 Agile Workflow Modeling and Execution based on Document
- 2009 A Comparing Method for Large Sanitized XML Trees
- 2009 A Memory Efficient and Secure Enterprise XML Processing
- 2008 A System and Method for Fully Decentralized Fine-grained Access Control on XML Data Structures
- 2006 Methods of Preventing XML Rewriting Attacks Against SOAP Messages
- 2006 Secure Time-based Auction Protocol

Languages

Native Bengali, Fluent English, business communication of French.

Technical Skills and Tools Literacy

- "Enterprise Architecture and Development methodology": TOGAF, SAP EAF, IBM SOA, Zachman, ITIL, SCRUM.
- "SOA, Web Service and XML Standards": ESB, SOAP, WSDL, UDDI, SCA, OSGi, WS-Discovery, WS-Security, WS-Policy, WS-Security Policy, WS-Trust, WS-SecureConversation, XACML, X-Path, XSLT, X-Query, XBRL, X-Form, X-Link, X-Pointer.
- "Business Process Languages": BPEL, BPEL4People, BPMN, WS-Orchestration, WS-Choreography.

- "Web Technology/Tools": Php, MySQL, Joomla, Axis2, REST, JSON, JavaScript, MS Visio.
- "Programming Languages and IDEs": C/C++, Java, Php, XHTML, SQL, EJB, Servlet, JSP, JDBC, JSTL, Tomcat, Eclipse, Spring, JBoss, Maven, Eclipse, Aptana, Ms Office, Open Office, Latex.
- "SAP Technology and tools": SAP BPM, XI / PI, ABAP Objects, ABAP WebDynpro, Netweaver Developer Studio, ABAP Workbench, SAP Workflow.

Extracurricular Activities

- Sports: Badminton, Cricket, Football.
- Traveling, Listening radios, Music (Original Soundtracks), Reading, Cinema, Internet

Bibliography

[AAM+95] Gustavo Alonso, G. Alonso, C. Mohan, Divyakant Agrawal, Amr El Abbadi, Roger Gunthor, D. Agrawal, A. El Abbadi, and Mohan Kamath. Exotica/FMQM: A Persistent Message-based Architecture for Distributed Workflow Management. pages 1–18, 1995.

[AAM97] Gustavo Alonso, G. Alonso, and C. Mohan. Workflow Management Systems: The Next Generation of Distributed Processing Tools, 1997.

[AB00] Mark Klein Abraham Bernstein, Chrysanthos Dellarocas. Adaptive Workflow Systems. volume 9, 2000.

[ABM05] Yuan An, Alex Borgida, and John Mylopoulos. Constructing Complex Semantic Mappings Between XML Data and Ontologies. In *The Semantic Web ISWC 2005*, pages 563–574. Springer Berlin / Heidelberg, 2005.

[ABP] Agile Business Process Modelling Framework and Enterprise architecture, http://www.agilealliance.org/system/article/file/740/file.pdf.

[ACM01] Vijayalakshmi Atluri, Soon Ae Chun, and Pietro Mazzoleni. A Chinese Wall Security Model for Decentralized Workflow Systems. In *CCS '01: Proceedings of the 8th ACM conference on Computer and Communications Security*, pages 48–57, New York, NY, USA, 2001. ACM.

[ADK+05] Vikas Agarwal, Koustuv Dasgupta, Neeran Karnik, Arun Kumar, Ashish Kundu, Sumit Mittal, and Biplav Srivastava. A Service Creation Environment based on End to End Composition of Web Services. In *WWW '05: Proceedings of the 14th international conference on World Wide Web*, pages 128–137, New York, NY, USA, 2005. ACM.

[ADO00] Wil M. P. van der Aalst, Jörg Desel, and Andreas Oberweis. Business Process Management, Models, Techniques, and Empirical Studies. London, UK, 2000. Springer-Verlag.

[AJA] AJAX: Asynchronous Javascript and XML.

[AkH96] Vijayalakshmi Atluri and Wei kuang Huang. An authorization model for workflows. In *In Proceedings of the 4th European Symposium on Research in Computer Security*, pages 44–64. Springer-Verlag, 1996.

[AKSX04] Rakesh Agrawal, Jerry Kiernan, Ramakrishnan Srikant, and Yirong Xu. Order preserving encryption for numeric data. In *SIGMOD '04: Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 563–574, New York, NY, USA, 2004. ACM.

[AMAA97] G. Alonso, C. Mohan, D. Agrawal, and D. Agrawal A. El Abbadi. Functionality and Limitations of Current Workflow Management Systems, 1997.

[AS96] Gustavo Alonso and Hans-Jorg Schek. Database Technology in Workflow Environments, 1996.

[ASKP00] Gail-Joon Ahn, Ravi Sandhu, Myong Kang, and Joon Park. Injecting rbac to secure a web-based workflow system. In *RBAC '00: Proceedings of the fifth ACM workshop on Role-based access control*, pages 1–10, New York, NY, USA, 2000. ACM.

[ASTK06] Ashraful Alam, Ganesh Subbiah, Bhavani Thuraisingam, and Latifur Khan. Reasoning with Semantics-aware Access Control Policies for Geospatial Web Services. In *SWS '06: Proceedings of the 3rd ACM workshop on Secure web services*, pages 69–76, New York, NY, USA, 2006. ACM.

[Ba05] S. Bowers and B. ascher. Actor-oriented Design of Scientific Workflows, 2005.

[BAA] Bringing Agility to Architecture, and Architecture to Agility, http://www.agilearchitect.org.

[BBB06a] Thomas Bauer Birgit Burmeister, Hans-Peter Steiert and Hartwig Baumgärtel. Agile Processes Through Goal- and Context-Oriented Business Process Modeling. In *Business Process Management Workshops, Workshop on Dynamic Process Management (DPM 2006)*, pages 217–228. Springer Berlin / Heidelberg, 2006.

[BBB06b] Thomas Bauer Birgit Burmeister, Hans-Peter Steiert and Hartwig Baumgärtel. Agile Processes Through Goal- and Context-Oriented Business Process Modeling. In *BUSINESS PROCESS MANAGEMENT WORKSHOPS*, pages 217–228. Springer, 2006.

[BBF+] Mark Bartel, John Boyer, Barb Fox, Brian LaMacchia, and Ed Simon. XML Signature Syntax and Processing (second edition), http://www.w3.org/tr/xmldsig-core/.

[BCF+] Scott Boag, Don Chamberlin, Mary F. Fernández, Daniela Florescu, Jonathan Robie, and Jérôme Siméon. XQuery 1.0: An XML Query Language. Technical report.

[BCF04] E. Bertino, B. Carminati, and E. Ferrari. Merkle Tree Authentication in UDDI Registries. Idea Group Inc, International Journal of Web Services Research, 1(2):37-57, 2004.

[BCP06] Elisa Bertino, Jason Crampton, and Federica Paci. Access control and authorization constraints for ws-bpel. In *ICWS '06: Proceedings of the IEEE International Conference on Web Services*, pages 275–284, Washington, DC, USA, 2006. IEEE Computer Society.

[BDW07] Matthias Born, Florian Dörr, and Ingo Weber. User Friendly Semantic Annotation in Business Process Modeling. *Lecture Notes in Computer Science : Web Information Systems Engineering WISE 2007 Workshops*, pages 260–271, 2007.

[BESP08] Jean Bacon, David M. Eyers, Jatinder Singh, and Peter R. Pietzuch. Access Control in Publish/Subscribe Systems. In *DEBS '08: Proceedings of the second international conference on Distributed event-based systems*, pages 23–34, New York, NY, USA, 2008. ACM.

[BF02] Elisa Bertino and Elena Ferrari. Secure and Selective Dissemination of XML Documents. *ACM Trans. Inf. Syst. Secur.*, 5(3):290–331, 2002.

[BFA99] Elisa Bertino, Elena Ferrari, and Vijay Atluri. The specification and enforcement of authorization constraints in workflow management systems. *ACM Trans. Inf. Syst. Secur.*, 2(1):65–104, 1999.

[BFG04] Karthikeyan Bhargavan, Cédric Fournet, and Andrew D. Gordon. Verifying Policy-based Security for Web Services. In *CCS '04: Proceedings of the 11th ACM conference on Computer and communications security*, pages 268–277, New York, NY, USA, 2004. ACM.

[Bil05] Philip Bille. A Survey on Tree Edit Distance and Related Problems. *Theoretical Computer Science*, 337:217–239, 2005.

[BLL04] Wang-Chien Lee Bo Luo, Dongwon Lee and Peng Liu. *A Flexible Framework for Architecting XML Access Control Enforcement Mechanisms*, volume Volume 3178/2004 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, December 2004.

[BM05] Walid Bagga and Refik Molva. Policy-based cryptography and applications. In *Financial Cryptography and Data Security*, pages 72–87. Springer Berlin / Heidelberg, 2005.

[BMR94] Daniel Barbara, Sharad Mehrotra, and Marek Rusinkiewicz. INCAS: A Computation Model for Dynamic Workflows in Autonomous Distributed Environments. Technical report, Matsushita Information Technology Labratory, 2 Research Way, 3rd Floor, Princeton, N.J. 08540, 1994.

[BN] A. D. A. Boujraf and M. Noble. Towards e-Administration in the Large (R4eGov). Deliverable WP3-D7, 2007. EU IP R4eGov.

[BN89] D.F.C. Brewer and M.J. Nash. The chinese wall security policy. In *Security and Privacy, 1989. Proceedings., 1989 IEEE Symposium on*, pages 206–214, May 1989.

[BNP08] Luc Bouganim, Francois Dang Ngoc, and Philippe Pucheral. Dynamic Access-Control Policies on XML Encrypted Data. *ACM Trans. Inf. Syst. Secur.*, 10(4):1–37, 2008.

[BP06] Changwoo Byun and Seog Park. An Efficient Yet Secure XML Access control Enforcement by Safe and Correct Query Modification. In *In Proc. of the 17th International Conference on Database and Expert Systems Applications (DEXA), 2006*, pages 276–285, 2006.

[BPE] Web Services Business Process Execution Language version 2.0, oasis standard.

[BPM] Business Process Modeling Notation, Object Management Group / Business Process Management Initiative, http://www.bpmn.org/.

[BPS] The Business of SOA: Evolving An Agile Enterprise with Service Oriented Architecture, http://www.bptrends.com/publicationfiles/03-06wp-soa-proforma-huchzemeier.pdf.

[BPT] Case Management: Combining Knowledge with process, bptrends july 2009, http://www.bptrends.com/.

[BR] Business Rules, http://www.agilemodeling.com/artifacts/businessrule.htm.

[CCPP96] Fabio Casati, Stefano Ceri, Barbara Pernici, and Giuseppe Pozzi. Workflow Evolution. In *ER '96: Proceedings of the 15th International Conference on Conceptual Modeling*, pages 438–455, London, UK, 1996. Springer-Verlag.

[CD] James Clark and Steve DeRose. XML Path Language (XPath), http://www.w3.org/tr/xpath.

[CGM97] Sudarshan S. Chawathe and Hector Garcia-Molina. Meaningful Change Detection in Structured Data. In *SIGMOD '97: Proceedings of the 1997 ACM SIGMOD international conference on Management of data*, pages 26–37, New York, NY, USA, 1997. ACM.

[CLW05] Shih-Chien Chou, An-Feng Liu, and Chien-Jung Wu. Preventing Information Leakage within Workflows that Execute among Competing Organizations. *J. Syst. Softw.*, 75(1-2):109–123, 2005.

[Coa98] Workflow Management Coalition. Workflow security considerations - white paper, technical report wfmc-tc-1019. Technical report, Workflow Management Coalition, 1998.

[CRGMW96] Sudarshan S. Chawathe, Anand Rajaraman, Hector Garcia-Molina, and Jennifer Widom. Change Detection in Hierarchically Structured Information. In *SIGMOD '96: Proceedings of the 1996 ACM SIGMOD international conference on Management of data*, pages 493–504, New York, NY, USA, 1996. ACM.

[DAM] The DARPA Agent Markup Language (DAML), http://www.daml.org/.

[DBL08] *Proceedings of the 12th International Conference on CSCW in Design, CSCWD 2008, April 16-18, 2008, Nanyang Hotel, Xi'an Jiaotong University, Xi'an, China*. IEEE, 2008.

[DCdVMS05] Sabrina De Capitani di Vimercati, Stefania Marrara, and Pierangela Samarati. An Access Control Model for Querying XML Data. In *SWS '05: Proceedings of the 2005 workshop on Secure web services*, pages 36–42, New York, NY, USA, 2005. ACM.

[DDdV+00] E. Damiani, S. De, S. De Capitani di Vimercati, S. Paraboschi, and P. Sarnarati. Securing XML Documents. In *In Proc. of the 2000 International Conference on Extending Database Technology (EDBT2000*, pages 121–135. Springer, 2000.

[DdVPS01] Ernesto Damiani, Sabrina De Capitani di Vimercati, Stefano Paraboschi, and Pierangela Samarati. Fine Grained Access Control for Soap E-services. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, pages 504–513, New York, NY, USA, 2001. ACM.

[DdVPS02] Ernesto Damiani, Sabrina De Capitani di Vimercati, Stefano Paraboschi, and Pierangela Samarati. A Fine-grained Access Control System for XML Documents. *ACM Trans. Inf. Syst. Secur.*, 5(2):169–202, 2002.

[DEL+00] Paul Dourish, W. Keith Edwards, Anthony LaMarca, John Lamping, Karin Petersen, Michael Salisbury, Douglas B. Terry, and James Thornton. Extending Document Management Systems with User-Specific Active Properties. *ACM Trans. Inf. Syst.*, 18(2):140–170, 2000.

[DFKO02] Ying Ding, Dieter Fensel, Michel Klein, and Borys Omelayenko. The Semantic Web: Yet Another Hip? *Data and Knowledge Engineering*, 41:205–227, 2002.

[DH76] Whitfield Diffie and Martin E. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.

[DIT] The OASIS DITA Pharmaceutical Content SC, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=dita-pharma.

[Dog97] *Workflow Management Systems and Interoperability, Proceedings of the NATO Advanced Study Institute on Workflow Management Systems (WFMS), held in Istanbul, Turkey, August 12 - 21, 1997*, volume 164 of *NATO ASI Series*, 1997.

[EAB+03] L. van Elst, F. R. Aschoff, A. Bernardi, H. Maus, and S. Schwarz. Weakly-structured Workflows for Knowledge-intensive Tasks: An Experimental Evaluation. In *WETICE '03: Proceedings of the Twelfth International Workshop on Enabling Technologies*, page 340, Washington, DC, USA, 2003. IEEE Computer Society.

[EBP] SOA Tools BPMN Modeler, http://www.eclipse.org/bpmn/.

[EBX] ebXML (Electronic Business using eXtensible Markup Language), http://www.ebxml.org/geninfo.htm.

[EC01] Greg Meredith Sanjiva Weerawarana Erik Christensen, Francisco Curbera. Web Services Description Language (WSDL) 1.1, http://www.w3.org/tr/wsdl. 2001.

[EFGK03] Patrick Th. Eugster, Pascal A. Felber, Rachid Guerraoui, and Anne-Marie Kermarrec. The Many Faces of Publish/Subscribe. *ACM Comput. Surv.*, 35(2):114–131, 2003.

[EK00] Clarence A. Ellis and Karim Keddara. A Workflow Change Is a Workflow. In *Business Process Management, Models, Techniques, and Empirical Studies*, pages 201–217, London, UK, 2000. Springer-Verlag.

[EP99] Johann Eder and Euthimios Panagos. Towards Distributed Workflow Process Management, 1999.

[EW07] Sandro Etalle and William H. Winsborough. A Posteriori Compliance Control. In *SACMAT '07: Proceedings of the 12th ACM symposium on Access control models and technologies*, pages 11–20, New York, NY, USA, 2007. ACM.

[FBD+02] D. Fensel, C. Bussler, Y. Ding, V. Kartseva, M. Klein, M. Korotkiy, B. Omelayenko, and R. Siebes. Semantic Web Application Areas. In *In Proc. 7th Int. Workshop on Applications of Natural Language to Information Systems (NLDB 2002*, pages 27–28, 2002.

[FCG04] Wenfei Fan, Chee-Yong Chan, and Minos Garofalakis. Secure XML Querying With Security Views. In *SIGMOD '04: Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 587–598, New York, NY, USA, 2004. ACM Press.

[Fea] Michael Feathers. Agile Workflow, Object Mentor, http://www.objectmentor.com/resources/articles/agileworkflow.pdf.

[FGIZ] Manel Fredj, Nikolaos Georgantas, Valérie Issarny, and Apostolos Zarras. Adaptation to connectivity loss in pervasive computing environments.

[FGIZ08] Manel Fredj, Nikolaos Georgantas, Valerie Issarny, and Apostolos Zarras. Dynamic Service Substitution in Service-Oriented Architectures. In *SERVICES '08: Proceedings of the 2008 IEEE Congress on Services - Part I*, pages 101–104, Washington, DC, USA, 2008. IEEE Computer Society.

[FJK+08] T. Finin, A. Joshi, L. Kagal, J. Niu, R. Sandhu, W. Winsborough, and B. Thuraisingham. Rowlbac: Representing Role Based Access Control in OWL. In NY ACM, New York, editor, *Proceedings of the 13th ACM Symposium on Access Control Models and Technologies (SACMAT '08), Estes Park, CO, USA.*, pages 73–82, June 2008.

[FZT04] Matthias Ferdinand, Christian Zirpins, and David Trastour. Lifting XML Schema to OWL. In *Web Engineering*, pages 563–574. Springer Berlin / Heidelberg, 2004.

[Gab05] Alban Gabillon. A Formal Access Control Model for XML Databases. In *In Proc. of the 2005 VLDB Workshop on Secure Data Management (SDM)*, pages 86–103, 2005.

[GAC+97] Esin Gokkoca, Mehmet Altinel, Ibrahim Cingil, Nesime Tatbul, Pinar Koksal, and Asuman Dogac. Design and Implementation of a Distributed Workflow Enactment Service. In *COOPIS '97: Proceedings of the Second IFCIS International Conference on Cooperative Information Systems*, pages 89–98, Washington, DC, USA, 1997. IEEE Computer Society.

[GHM+] Martin Gudgin, Marc Hadley, Noah Mendelsohn, Henrik Frystyk Nielsen, Anish Karmarkar, and Yves Lafon. SOAP version 1.2 part 1: Messaging framework (second edition), http://www.w3.org/tr/soap12-part1/.

[GM] Simon Godik and Tim Moses. extensible Access Control Markup Language (xacml), http://www.oasis-open.org/committees/xacml/repository/cs-xacml-specification-1.1.pdf.

[HMNS] Harald Holz, Heiko Maus, Naoyuki Nomura, and Martin Schaaf. Second workshop on Knowledge Management for Distributed Agile Processes: Models, Techniques, and Infrastructure (KMDAP 2005),in professional knowledge management. third biennial conference, wm 2005, kaiserslautern, germany, april 2005. revised selected papers.

[Hola] David Hollingsworth. Microsoft Biztalk, http://www.microsoft.com/biztalk/.

[Holb] David Hollingsworth. TIBCO iProcess Suite, http://www.tibco.com/software/business-process-management/iprocess-suite/default.jsp.

[Holc] David Hollingsworth. The Workflow Reference Model, Technical Report WFMC-TC-1003, The Workflow Management Coalition, 1995.

[Hua] Dong Huang. Semantic Policy-based Security Framework for b.

[HWW] Philippe Le Hégaret, Ray Whitmer, and Lauren Wood. Document Object Model (DOM), http://www.w3.org/dom/.

[hZyLL05] Zhao hui ZHANG, Da you LIU, and Wei-Jiang LIU. Using Ontologies for Distributed Workflow Management. *Services Computing, IEEE International Conference on*, 2:263–264, 2005.

[IBMa] IBM BPM suite, http://www-01.ibm.com/software/info/bpm/offerings.html.

[IBMb] IBM WebSphere, Enterprise Service Bus, http://www-01.ibm.com/software/integration/wsesb/.

[IPT] International Press Telecommunication Council (iptc), http://www.iptc.org/.

[JDO] Java Document Object Model, http://www.jdom.org/.

[JMM03] Daniel J., Mandell1, and Sheila A. McIlraith. Adapting BPEL4WS for the Semantic Web: The Bottom-Up Approach to Web Service Interoperation. In *The SemanticWeb - ISWC 2003, Proceedings of International Semantic Web Conference, October 2003*, pages 227–241. Springer Berlin / Heidelberg, 2003.

[JMS] Java Message Service (JMS), http://java.sun.com/products/jms/.

[JOS] Joseki - A SPARQL Server for Jena, http://www.joseki.org/.

[JSP] Java Server Pages, http://java.sun.com/products/jsp/.

[JWST] Amit Jain, Duminda Wijesekera, Anoop Singhal, and Bhavani Thuraisingham. Semantic-Aware Data Protection in Web Services, http://www.cse.sc.edu/ jain/files/semantic-awarexmlsecurity.pdf.

[JWZ94] Tao Jiang, Lusheng Wang, and Kaizhong Zhang. Alignment of Trees - An Alternative to Tree Edit. In *CPM '94: Proceedings of the 5th Annual Symposium on Combinatorial Pattern Matching*, pages 75–86, London, UK, 1994. Springer-Verlag.

[Kaya] Michael Kay. An Anatomy of an XSLT Processor, http://www.ibm.com/developerworks/xml/library/x-xslt2/.

[Kayb] Michael Kay. Building Workflow Applications with XML and XQuery, http://www.stylusstudio.com/xml/workflow.html.

[KB08] A. Kundu and E. Bertino. A New Model for Secure Dissemination of XML Content. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 38(3):292–301, May 2008.

[KE06] Ashish Kundu and Bertino Elisa. Secure Dissemination of XML Content Using Structure-based Routing. In *EDOC '06: Proceedings of the 10th IEEE International Enterprise Distributed Object Computing Conference*, pages 153–164, Washington, DC, USA, 2006. IEEE Computer Society.

[KFJ03] Lalana Kagal, Tim Finin, and Anupam Joshi. A Policy Language for a Pervasive Computing Environment. In *POLICY '03: Proceedings of the 4th IEEE International Workshop on Policies for Distributed Systems and Networks*, page 63, Washington, DC, USA, 2003. IEEE Computer Society.

[KH00] Michiharu Kudo and Satoshi Hada. Xml Document Security based on Provisional Authorization. In *CCS '00: Proceedings of the 7th ACM conference on Computer and communications security*, pages 87–96, New York, NY, USA, 2000. ACM.

[KHB00] B. Kiepuszewski, A. H. M. Ter Hofstede, and C. Bussler. On Structured Workfow Modelling, 2000.

[KM03] Hristo Koshutanski and Fabio Massacci. An access control framework for business processes for web services. In *XMLSEC '03: Proceedings of the 2003 ACM workshop on XML security*, pages 15–24, New York, NY, USA, 2003. ACM.

[KMK02] Rupa Krishnan, Lalitha Munaga, and Kamalakar Karlapalem. XDoC-WFMS: A Framework for Document Centric Workflow Management System. In *Revised Papers from the HUMACS, DASWIS, ECOMO, and DAMA on ER 2001 Workshops*, pages 348–362, London, UK, 2002. Springer-Verlag.

[KMR05] Gabriel Kuper, Fabio Massacci, and Nataliya Rassadko. Generalized XML Security Views. In *SACMAT '05: Proceedings of the tenth ACM symposium on Access control models and technologies*, pages 77–84, New York, NY, USA, 2005. ACM Press.

[KPF01] Myong H. Kang, Joon S. Park, and Judith N. Froscher. Access Control Mechanisms for Inter-Organizational Workflow. In *SACMAT '01: Proceedings of the sixth ACM symposium on Access control models and technologies*, pages 66–74, New York, NY, USA, 2001. ACM.

[KPS+04] Lalana Kagal, Massimo Paolucci, Naveen Srinivasan, Grit Denker, Tim Finin, and Katia Sycara. Authorization and Privacy for Semantic Web Services. *IEEE Intelligent Systems*, 19(4):50–56, 2004.

[KPT00] Yongdae Kim, Adrian Perrig, and Gene Tsudik. Simple and Fault-Tolerant Key Agreement for Dynamic Collaborative Groups. In *CCS '00: Proceedings of the 7th ACM conference on Computer and communications security*, pages 235–244, New York, NY, USA, 2000. ACM Press.

[KR01] Konstantin Knorr and Susanne Röhrig. Security requirements of e-business processes. In *I3E '01: Proceedings of the IFIP Conference on Towards The E-Society*, pages 73–86, Deventer, The Netherlands, The Netherlands, 2001. Kluwer, B.V.

[Kun] Akiomi Kunisa. Digital Watermarking based on Guided Scrambling and Its Robustness Evaluation to JPEG Compression. In *IEICE Trans Fundam Electron Commun Comput Sci (Inst Electron Inf Commun Eng),VOL.E86-A;NO.9;PAGE.2366-2375(2003)*.

[LLLL04] Bo Luo, Dongwon Lee, Wang-Chien Lee, and Peng Liu. QFilter: Fine-grained Run-time XML Access Control via NFA-based Query Rewriting. In *CIKM '04: Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 543–552, New York, NY, USA, 2004. ACM Press.

[LLY02] P. Lee, J. Lui, and D. Yau. Distributed Collaborative Key Agreement Protocols for Dynamic Peer Groups, 2002.

[LNS07] Hongrae Lee, Raymond T. Ng, and Kyuseok Shim. Extending q-grams to Estimate Selectivity of String Matching with Low Edit Distance. In *VLDB '07: Proceedings of the 33rd international conference on Very large data bases*, pages 195–206. VLDB Endowment, 2007.

[LTL] A Genealogy of LTL-to-Büchi Translators, http://spot.lip6.fr/wiki/ltltranslationalgorithms.

[LYGL01] Xiaozhou Steve Li, Yang Richard Yang, Mohamed G. Gouda, and Simon S. Lam. Batch Rekeying for Secure Group Communications. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, pages 525–534, New York, NY, USA, 2001. ACM Press.

[LZS09] Yahui Lu, Li Zhang, and Jiaguang Sun. Task-Activity based Access Control for Process Collaboration Environments. *Comput. Ind.*, 60(6):403–415, 2009.

[MBW07] Florian Dörr Matthias Born and Ingo Weber. User-friendly semantic annotation in business process modeling. In *Web Information Systems Engineering Ű WISE 2007 Workshops*, pages 260–271. Springer Berlin / Heidelberg, 2007.

[Mer89] R.C. Merkle. A Certified Digital Signature. *In Advances in Cryptology - CRYPTO '89 Proceedings, Lecture Notes in Computer Science*, 435:218–238, 1989.

[MFBK06] Giovanni Mella, Elena Ferrari, Elisa Bertino, and Yunhua Koglin. Controlled and Cooperative Updates of XML Documents in Byzantine and Failure-Prone Distributed Systems. *ACM Trans. Inf. Syst. Secur.*, 9(4):421–460, 2006.

[MK] Hatsukazu Tanaka Minoru Kuribayashi. A robust Watermarking System based on the Properties of Low Frequency in Perceptual Audio Coding. In *IEICE Trans Fundam Electron Commun Comput Sci (Inst Electron Inf Commun Eng), vol. E86-A, no. 12, pp. 3267-3275, December 2003*.

[MM04] Nikola Milanovic and Miroslaw Malek. Current Solutions for Web Service Composition. *IEEE Internet Computing*, 8(6):51–59, 2004.

[MM05] F. Montagut and R. Molva. Enabling Pervasive Execution of Workflows. *International Conference on Collaborative Computing: Networking, Applications and Worksharing*, 0:10 pp., 2005.

[MM07] Frederic Montagut and Refik Molva. Enforcing Integrity of Execution in Distributed Workflow Management Systems. *Services Computing, IEEE International Conference on*, 0:170–177, 2007.

[MPvdA07] N. Sidorova M. Pesic, M. H. Schonenberg and W. M. P. van der Aalst. Constraint Based Workflow Models: Change Made Easy. *Lecture Notes in Computer Science : On the Move to Meaningful Internet Systems 2007*, pages 77–94, 2007.

[MRW96] Stefan Morschheuser, Heinz Raufer, and Christoph Wargitsch. Challenges and Solutions of Document and Workflow Management in a Manufacturing Enterprise: A Case Study. In *HICSS '96: Proceedings of the 29th Hawaii International Conference on System Sciences Volume 5: Digital Documents*, page 4, Washington, DC, USA, 1996. IEEE Computer Society.

[MS03] Gerome Miklau and Dan Suciu. Controlling Access to Published Data Using Cryptography. In *VLDB*, pages 898–909, 2003.

[MSB08] Mirjam Minor, Daniel Schmalen, and Ralph Bergmann. XML-based Representation of Agile Workflows. In Martin Bichler, Thomas Hess, Helmut Krcmar, Ulrike Lechner, Florian Matthes, Arnold Picot, Benjamin Speitkamp, and Petra Wolf, editors, *Multikonferenz Wirtschaftsinformatik*. GITO-Verlag, Berlin, 2008.

[MTKH03] Makoto Murata, Akihiko Tozawa, Michiharu Kudo, and Satoshi Hada. XML Access Control Using Static Analysis. In *CCS '03: Proceedings of the 10th ACM conference on Computer and communications security*, pages 73–84, New York, NY, USA, 2003. ACM Press.

[Mye86] Eugene W. Myers. An O(ND) Difference Algorithm and its Variations. *Algorithmica*, 1:251–266, 1986.

[NJ02] Andrew Nierman and H. V. Jagadish. Evaluating Structural Similarity in XML Documents. pages 61–66, 2002.

[NKMHB] Anthony Nadalin, Chris Kaler, Ronald Monzillo, and Phillip Hallam-Baker. Web Services Security:SOAP Message Security 1.1 (WS-Security 2004), http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-soapmessagesecurity.pdf.

[NlL05] Wilfred Ng and Ho lam Lau. Effective Approaches for Watermarking XML Data. In *In DASFAA*, pages 68–80, 2005.

[NM] Natalya F. Noy and Deborah L. McGuinness. Ontology Development 101: A Guide to Creating Your First Ontology.

[ODF] OASIS Open Document Format for Office Applications (OpenDocument),http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=office.

[OPEa] Openlaszlo, a framework for generating user interfaces, http://www.openlaszlo.org/.

[OPEb] OpenXML Document Viewer, http://www.codeplex.com/openxmlviewer.

[ORY] The Oryx Editor for BPMN, http://bpt.hpi.uni-potsdam.de/oryx.

[OWL] OWL Web Ontology Language Overview, http://www.w3.org/tr/owl-features/.

[PE] Joe Sventek Patrick Eugster, Rachid Guerraoui. Type-Based Publish/Subscribe, Technical Report, EPFL, http://ic2.epfl.ch/publications/documents/ic_tech_report_200029.pdf.

[PET] PEtALS, An Enterprise Service Bus, http://www.petalslink.com/lang-en.

[PFJ+01] ao Pereira, Jo, Franccoise Fabret, Hans-Arno Jacobsen, François Llirbat, and Dennis Shasha. WebFilter: A High-throughput XML-based Publish and Subscribe System. In *VLDB '01: Proceedings of the 27th International Conference on Very Large Data Bases*, pages 723–724, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[PH03] Joon S. Park and Junseok Hwang. Role-based access control for collaborative enterprise in peer-to-peer computing environments. In *SACMAT '03: Proceedings of the eighth ACM symposium on Access control models and technologies*, pages 93–99, New York, NY, USA, 2003. ACM.

[Pnu77] Amir Pnueli. The Temporal Logic of Programs. In *SFCS '77: Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, pages 46–57, Washington, DC, USA, 1977. IEEE Computer Society.

[Pod08] Marius Ioan Podean. Document and Workflow Management in Collaborative Systems, Babes-Bolyai University of Cluj-Napoca. In *Economy Informatics, 1-4/2008, Working paper*, 2008.

[Pow96] David Powell. Group Communication (Introduction to the Special Section). *Commun. ACM*, 39(4):50–53, 1996.

[Pra] Atul Prakash. Group Editors, University of Michigan, http://www.lri.fr/ mbl/trends-cscw/.

[PS07] Jeff Pollock and Susie Stephens. Enterprise Semantic Web, Semantic Technologies 2007, http://me.jtpollock.us/pubs/2007.05-pollock.stc.2007.pdf. 2007.

[PSC03] V. Parmar, Hongchi Shi, and Su-Shing Chen. XML Access Control for Semantically Related XML Documents. *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on*, pages 10 pp.–, Jan. 2003.

[R4E] R4eGov: Making eGovernment Work, http://www.r4egov.eu/.

[RB01] Erhard Rahm and Philip A. Bernstein. A Survey of Approaches to Automatic Schema Matching. *VLDB Journal: Very Large Data Bases*, 10(4):334–350, 2001.

[RDF] Resource Description Framework (RDF), http://www.w3.org/rdf/.

[RMM93] P Flores R Medina-Mora, H K T Wong. ActionWorkflow as the Enterprise Integration Technology. *Bulletin of the Technical Committee on Data Engineering 16 (2) (1993)*, pages 49–52, 1993.

[RMS06] Mohammad Ashiqur Rahaman, Rits Marten, and Andreas Schaad. An Inline Approach for Secure SOAP Requests and Early Validation. In *OWASP 2006. European Conference on Open Web Application Security Project. May 30-31, 2006, Leuven, Belgium*, May 2006.

[Rob96] William N. Robinson. Goal-Oriented Workflow Analysis and Infrastructure. In *Georgia State University, Working paper*, pages 96–07, 1996.

[Ros] RosettaNet, http://www.rosettanet.org/cms/sites/rosettanet/.

[Ros04] Davide Rossi. Orchestrating Document-based Workflows with X-Folders. In *SAC '04: Proceedings of the 2004 ACM symposium on Applied computing*, pages 503–507, New York, NY, USA, 2004. ACM.

[RPRS09] Mohammad Ashiqur Rahaman, Henrik Plate, Yves Roudier, and Andreas Schaad. Content Driven Secure and Selective XML Dissemination. Technical Report RR-09-219, Eurécom, 05 2009.

[RRL08] Cristiano G. Rezende, Bruno P. S. Rocha, and Antêonio A. F. Loureiro. Publish/Subscribe Architecture for Mobile Adhoc Networks. In *SAC '08: Proceedings of the 2008 ACM symposium on Applied computing*, pages 1913–1917, New York, NY, USA, 2008. ACM.

[RRMS09] Mohammad Ashiqur Rahaman, Yves Roudier, Philip Miseldine, and Andreas Schaad. Ontology-based Secure XML Content Distribution. In *IFIP SEC 2009, 24th International Information Security Conference, May 18-20, 2009, Pafos, Cyprus*, May 2009.

[RRS] Mohammad Ashiqur Rahaman, Yves Roudier, and Andreas Schaad. A Publish/Subscribe Model for Secure Content-Driven XML Dissemination. In *the special issue of Journal of Information Assurance and Security (JIAS), volume 6, http://www.mirlabs.org/jias/.*

[RRS08] Mohammad Ashiqur Rahaman, Yves Roudier, and Andreas Schaad. Distributed Access Control for XML Document Centric Collaborations. In *EDOC '08: Proceedings of the 2008 12th International IEEE Enterprise Distributed Object Computing Conference*, pages 267–276, Washington, DC, USA, 2008. IEEE Computer Society.

[RRS09a] Mohammad Ashiqur Rahaman, Yves Roudier, and Andreas Schaad. Document-based Dynamic Workflows: Towards Flexible and Stateful Services. In *Proceedings of the IEEE SCC 2009 International Conference on Services Computing, September 21-25, 2009, Bangalore, India*. IEEE Computer Society, 2009.

[RRS09b] Mohammad Ashiqur Rahaman, Yves Roudier, and Andreas Schaad. A secure comparison technique for tree structured data. In *ICIW '09: Proceedings of the 2009 Fourth International Conference on Internet and Web Applications and Services*, pages 304–309, Washington, DC, USA, 2009. IEEE Computer Society.

[RRS09c] Mohammad Ashiqur Rahaman, Yves Roudier, and Andreas Schaad. Towards Secure Content Based Dissemination of XML Documents. In *The Fifth International Conference on Information Assurance and Security (IAS 09), X'ian, China*, August 18-20 2009.

[RS04] Jinghai Rao and Xiaomeng Su. A Survey of Automated Web Service Composition Methods. In *In Proceedings of the First International Workshop on Semantic Web Services and Web Process Composition, SWSWPC 2004*, pages 43–54, 2004.

[RS07a] Mohammad Ashiqur Rahaman and Andreas Schaad. SOAP-based Secure Conversation and Collaboration. In *ICWS '07: Proceedings of the IEEE International Conference on web services*, pages 471–480, Los Alamitos, CA, USA, 2007. IEEE Computer Society.

[RS07b] Mohammad Ashiqur Rahaman and Andreas Schaad. SOAP-based Secure Conversation and Collaboration. *Web Services, IEEE International Conference on*, 0:471–480, 2007.

[RSR06] Mohammad Ashiqur Rahaman, Andreas Schaad, and Maarten Rits. Towards Secure SOAP Message Exchange in a SOA. In *SWS '06: Proceedings of the 3rd ACM workshop on Secure web services*, pages 77–84, New York, NY, USA, 2006. ACM.

[SAPa] A Purchase Order WSDL Document, SAP Enterprise Services Workplace, http://esoadocu.sap.com/socoview(bd1lbizjptgwmczkpw1pbg==)/get_wsdl.xml?packageid=dbbb6d8aa3b382f191e0000f20f64781&id=0afcbb068cee3d59a67b420bc73f2f1b.

[SAPb] SAP NETWEAVER BPM White Paper, https://www.sdn.sap.com/irj/scn/go/-portal/prtroot/docs/library/uuid/d014cef6-37cf-2b10-e8ae-871324d54d8d.

[SAPc] SAP NetWeaver Process Integration (SAP PI), http://www.sdn.sap.com/irj/sdn/nw-soa.

[SAX] About Sax, http://www.saxproject.org/.

[SBP] Semantic Business Process Management Working Group, http://www.sbpm.org/.

[SCFY96] Ravi S. Sandhu, Edward J. Coyne, Hal L. Feinstein, and Charles E. Youman. Role-Based Access Control Models. *Computer*, 29(2):38–47, 1996.

[Sel77] Stanley M. Selkow. The Tree-to-Tree Editing Problem. *Inf. Process. Lett.*, 6(6):184–186, 1977.

[SHM08] Hisashi Shimosaka, Tomoyuki Hiroyasu, and Mitsunori Miki. Distributed Workflow Management System based on Publish-Subscribe Notification for Web Services. *New Generation Computing, Friday, April 04, 2008*, pages 395–408, 2008.

[SHO] Simple HTML Ontology Extensions (SHOE), http://www.cs.umd.edu/projects/plus/shoe/.

[Sida] Bilal Siddiqui. Design XML Schmas for Enterprise Data, http://www.ibm.com/developerworks/edu/x-dw-x-schemadata.html.

[Sidb] Bilal Siddiqui. Understanding DOM, http://www.ibm.com/developerworks/edu/x-dw-xudom-i.html.

[SIE] SIENA, A Publish/Subscribe Wide Area Event Notification, http://www.inf.usi.ch/carzaniga/siena/.

[SMLP05] Yuqing Sun, Xiangxu Meng, Shijun Liu, and Peng Pan. An Approach for Flexible RBAC Workflow System. *International Conference on Computer Supported Cooperative Work in Design*, 1:524–529 Vol. 1, 2005.

[SOX] Sarbanes-Oxley, Financial and Accounting Disclosure Information, http://www.sarbanes-oxley.com/.

[SPM] Semantic Process Modelling Environment, Semantics Utilised for Pprocess Management withing and between Enterprises (SUPER). deliverable d 5.1, 2007. EU IP SUPER, http://www.ip-super.org.

[SPQ] SPARQL Query Language for RDF, http://www.w3.org/tr/rdf-sparql-query/.

[SS] Six Sigma, http://www.isixsigma.com.

[STA] The Streaming API for XML, http://stax.codehaus.org/.

[Sto] Klaas-Jan Stol. A Framework for Document-oriented, Workflow-enabled Applications, Computing Science. University of Groningen,www.cs.rug.nl/ aiellom/tesi/stol.pdf. Technical report.

[SWS] Semantic Web Services - Use Cases, http://www.daml.org/services/use-cases/architecture/.

[SWT] Semantic Web Tools, http://esw.w3.org/topic/semanticwebtools.

[SWU] Semantic Web Use Cases: http://www.w3.org/2001/sw/sweo/public/usecases/.

[SYY+08] Weiming Shen, Jianming Yong, Yun Yang, Jean-Paul A. Barthès, and Junzhou Luo, editors. *Computer Supported Cooperative Work in Design IV, 11th International Conference, CSCWD 2007, Melbourne, Australia, April 26-28, 2007. Revised Selected Papers*, volume 5236 of *Lecture Notes in Computer Science*. Springer, 2008.

[SZ89] D. Shasha and K. Zhang. Fast Parallel Algorithms for the Unit Cost Editing Distance Between Trees. In *SPAA '89: Proceedings of the first annual ACM symposium on Parallel algorithms and architectures*, pages 117–126, New York, NY, USA, 1989. ACM.

[SZ97] Dennis Shasha and Kaizhong Zhang. Approximate Tree Pattern Matching. In *In Pattern Matching Algorithms*, pages 341–371. Oxford University Press, 1997.

[Tag01] Roger Tagg. Document-Oriented and Process-Oriented Views in Lightweight Workflow, http://www.cis.unisa.edu.au/ cisrmt/unpublished/taggdocprocwf01.doc. School of Computer and Information Science, University of South Australia, Mawson Lakes, SA 5095, 2001.

[TAK03] Anand R. Tripathi, Tanvir Ahmed, and Richa Kumar. Specification of secure distributed collaboration systems. In *ISADS '03: Proceedings of the The Sixth International Symposium on Autonomous Decentralized Systems (ISADS'03)*, page 149, Washington, DC, USA, 2003. IEEE Computer Society.

[TBLL01] James Hendler Tim Berners-Lee and Ora Lassila. The Semantic Web, Scientific American, May 2001. 2001.

[TLC05] W. T. Tsai, Xinxin Liu, and Yinong Chen. Distributed policy specification and enforcement in service-oriented business systems. In *ICEBE '05: Proceedings of the IEEE International Conference on e-Business Engineering*, pages 10–17, Washington, DC, USA, 2005. IEEE Computer Society.

[TOM] Apache Software Foundation - tomcat application server, http://tomcat.apache.org/.

[UDRS05] Octavian Udrea, Yu Deng, Edna Ruckhaus, and V.S. Subrahmanian. A Graph Theoretical Foundation for Integrating RDF Ontologies. *Proccedings of the Twentieth National Conference on Artificial Intelligence*, pages 1442–1447, 2005.

[Ukk91] Esko Ukkonen. Approximate String Matching with q-grams and Maximal Matches. Technical report, 1991.

[UML] Unified Modeling Language, Object Management group, http://www.uml.org/.

[VBS04] José M. Vidal, Paul Buhler, and Christian Stahl. Multiagent Systems with Workflows. *IEEE Internet Computing*, 8(1):76–82, 2004.

[VCFS00] Nicolas Vidot, Michelle Cart, Jean Ferrié, and Maher Suleiman. Copies Convergence in a Distributed Real-time Collaborative Environment. In *CSCW '00: Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pages 171–180, New York, NY, USA, 2000. ACM.

[vdAJ00] W. M. P. van der Aalst and S. Jablonski. Dealing with Workflow Change: Identification of Issues and Solutions. *International Journal of Computer Systems Science and Engineering*, 15(5):267–276, September 2000.

[vdAP06] W. M. P. van der Aalst and M. Pesic. DecSerFlow: Towards a Truly Declarative Service Flow Language. *Lecture Notes in Computer Science : Web Services and Formal Methods, Volume 4184, 2006*, pages 1–23, 2006.

[vdAW05] Wil M. P. van der Aalst and Mathias Weske. Case Handling: A New Paradigm for Business Process Support. *Data Knowl. Eng.*, 53(2):129–162, 2005.

[Vig08] Pierre Vigneras. Why BPEL is not the holy grail for BPM, http://www.infoq.com/articles/bpelbpm. 2008.

[W3C] The World Wide Web Consortium (W3C), http://www.w3.org/.

[Wag75] Robert A. Wagner. On the complexity of the extended string-to-string correction problem. In *STOC '75: Proceedings of seventh annual ACM symposium on Theory of computing*, pages 218–223, New York, NY, USA, 1975. ACM.

[Wan00] Alf Inge Wang. Using Software Agents to Support Evolution of Distributed Workflow Models. In *In In Proc. International ICSC Symposium on Interactive and Collaborative Computing (ICCŠ2000*, page page, 2000.

[WCX] Ching-Te Wang, Tung-Shou Chen, and Zhen-Ming Xu. Video Watermarking of Which Embedded Information Depends on the Distance between Two Signal Positions. In *IEICE Trans Fundam Electron Commun Comput Sci (Inst Electron Inf Commun Eng) VOL.E87-A;NO.8;PAGE.2152-2159(2004)*.

[Wes01] M. Weske. Formal Foundation and Conceptual Design of Dynamic Adaptations in a Workflow Management System. In *HICSS '01: Proceedings of the 34th Annual Hawaii International Conference on System Sciences (HICSS-34)-Volume 7*, page 7051, Washington, DC, USA, 2001. IEEE Computer Society.

[WF74] Robert A. Wagner and Michael J. Fischer. The String-to-String Correction Problem. *J. ACM*, 21(1):168–173, 1974.

[WJL04] Jinling Wang, Beihong Jin, and Jing Li. An Ontology-based Publish/Subscribe System. In *In Middleware*, pages 232–253, 2004.

[WK05] Jianrui Wang and Akhil Kumar. A Framework for Document-Driven Workflow Systems. In *Business Process Managemant*, pages 285–301, New York, NY, USA, 2005. Springer Berlin / Heidelberg, Lecture Notes in Computer Science.

[WKRL06] Martin Wimmer, Alfons Kemper, Maarten Rits, and Volkmar Lotz. Consolidating the access control of composite applications and workflows. In *In Proceedings of DBSec, 06, volume 4127 of LNCS, pages 44 to 59, Sophia Antipolis*, 2006.

[WMD07] Ingo Weber, Ivan Markovic, and Christian Drumm. A Conceptual Framework for Composition in Business Process Management. In *Business Information Systems*, pages 54–66. Springer Berlin / Heidelberg, 2007.

[WMMM90] S. Wu, U. Manber, G. Myers, and W. Miller. An O(NP) Sequence Comparison Algorithm. *Inf. Process. Lett.*, 35(6):317–323, 1990.

[WSN] Web Services Notification, http://www.oasis-open.org/committees/tchome.php?wgabbrev=wsn.

[WSO] WSO2, Enterprise Service Bus, http://wso2.org/projects/esb/java.

[WSP] Web Services Policy 1.2 - Framework (WS-Policy), http://www.w3.org/submission/ws-policy/.

[WT04] Roosdiana Wonohoesodo and Zahir Tari. A role based access control for web services. *Services Computing, IEEE International Conference on*, 0:49–56, 2004.

[WW98] T. Wewers and C. Wargitsch. Four Dimensions of Interorganizational, Document-oriented Workflow: A Case Study of the Approval of Hazardous-waste Disposal. volume 4, pages 332–341 vol.4, Jan 1998.

[WW05] Barbara Weber and Werner Wild. Towards the Agile Management of Business Processes. In Klaus-Dieter Althoff, Andreas Dengel, Ralph Bergmann, Markus Nick, and Thomas Roth-Berghofer, editors, *Wissensmanagement*, pages 375–382. DFKI, Kaiserslautern, 2005.

[XAC] Sun's XACML Implementation, http://sunxacml.sourceforge.net/.

[XBR] XBRL, eXtensible Business Reporting Language, http://www.xbrl.org/whatisxbrl/.

[XSL99] XSL Transformations (XSLT), Version 1.0, http://www.w3.org/tr/xslt. 1999.

[YdmGM05] Mariemma I. Yagüe, María del-mar Gallardo, and Antonio Maña. Semantic Access Control Model: A Formal Specification, european symposium on research in computer security, ESORICS 2005 pp.24-43, lncs 3679, springer-verlag, milano, italy,. 2005.

[YKT05] Rui Yang, Panos Kalnis, and Anthony K. H. Tung. Similarity Evaluation on Tree-Structured Data. In *SIGMOD '05: Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 754–765, New York, NY, USA, 2005. ACM.

[YZL06] Ronghua Yao, Qijun Zhao, and Hongtao Lu. A novel Watermark Algorithm for Integrity Protection of XML Documents. In *IJCSNS International Journal of Computer Science and Network Security, VOL.6 No.2B, February 2006*, 2006.

[ZS89] K. Zhang and D. Shasha. Simple Fast Algorithms for the Editing Distance Between Trees and Related Problems. *SIAM J. Comput.*, 18(6):1245–1262, 1989.