

Institut für Architektur von Anwendungssystemen
Universität Stuttgart
Universitätsstraße 38
70569 Stuttgart

Diplomarbeit Nr. 3113

**A Unified Framework for Security
Visualization and Enforcement in Business
Process Driven Environments**

Thomas Karsten



Studiengang: Softwaretechnik

Prüfer: Prof. Dr. Frank Leymann
Betreuer: Dr. Andreas Schaad
Dipl.-Inf. Tobias Binz
Dipl.-Inf. Steve Strauch

begonnen am: 01.12.2010
beendet am: 23.05.2011

CR-Klassifikation D.2.3, D.3.3, K.6.5, H.4.1

Abstract

Service-oriented architecture offers a promising approach for supporting interoperability and flexibility in the context of increasingly dynamic and rapidly changing requirements in the business world. However, encapsulation of business functionalities as self-contained services, as one of the main concepts in a SOA, brings new challenges. While business experts concentrate on the domain-specific aspects, other non-functional requirements such as security remain mostly neglected, if all understood. Costs for security administration may increase, business-driven security requirements may not be addressed and security configurations may not match at all internal and external regulations and guidelines. Based on these needs, we propose a technology-independent framework that provides graphical concepts for incorporating the security demands, facilitating the handling of security requirements from the specification to their realization.

Contents:

1	Introduction	6
1.1	Motivating Example	7
1.1.1	Retailer.....	8
1.1.2	Manufacturer.....	9
1.1.3	Rating Agency	10
1.2	Problem Statement.....	10
1.3	Outline / Structure of the Thesis	11
2	Related Work	15
2.1	Security Modeling.....	15
2.2	Modeling Constraints in Workflows.....	17
2.3	Modeling Roles and Access Control.....	17
3	Fundamentals	20
3.1	Service-Oriented Architecture	20
3.1.1	SOAP	21
3.1.2	Web Services Definition Language (WSDL)	22
3.2	Security	23
3.2.1	Security Requirements	23
3.2.2	Access Control.....	23
3.2.3	Security Mechanisms	24
3.3	Business Process Management	32
3.3.1	WS Business Process Execution Language (WS-BPEL)	34
3.3.2	Business Process Model and Notation (BPMN).....	34
3.3.3	Workflow Foundation (WF)	36
4	Workflow Model Concept and Specification	39
4.1	Conceptual Terminology	39
4.2	Conceptual Workflow Model	41
4.2.1	Agent Lanes	42
4.2.2	Agent Assignment Types.....	42
4.2.3	Operations.....	43
4.2.4	Resources	44
4.3	Conceptual Access Control Model	44
4.3.1	Authentication.....	44
4.3.2	Authorization	46
4.3.3	Impersonation	46
5	Security Model Concept and Specification.....	49
5.1	General Concept.....	49

5.2	Business Process Modeling Layer	51
5.3	Security Goals.....	54
5.4	Security Requirements.....	57
5.5	Motivating Example	64
5.5.1	Agent Security Goals	64
5.5.2	Check Inventory Security Goals.....	65
5.5.3	Prepare Order Security Goals	66
5.5.4	Place Order Security Goals.....	67
5.5.5	Update Inventory Security Goals.....	68
5.5.6	Mapping Security Goals to the Security Requirements.....	69
6	Design and Implementation	72
6.1	Secure Workflow Designer Prototype	72
6.2	Development Environment.....	73
6.3	General Architecture.....	74
6.4	Prototype Implementation.....	77
6.4.1	Re-hosting Functionality.....	77
6.4.2	Workflow-related Functionality.....	78
6.4.3	Auditing Security Requirement	81
6.4.4	Infrastructure-related Functionality	81
7	Summary and Future Work	87

List of Figures:

Figure 1.1: Supply Chain Process	7
Figure 1.2: Thesis Methodology.....	12
Figure 3.1: SOA-triangle (Weerawarana, Curbera, Leymann, Storey, & Ferguson, 2005)	21
Figure 3.2: Structure of a SOAP message	21
Figure 3.3: WSDL v.1.1 Specification	22
Figure 3.4: Web Services Security Specifications (Weerawarana, Curbera, Leymann, Storey, & Ferguson, 2005)	25
Figure 3.5: Detached Structure Referencing an External Resource (Rosenberg & Remy, 2004).....	25
Figure 3.6: XKMS Services (Bertino, Martino, Paci, & Squicciarini, 2009).....	26
Figure 3.7: SAML Assertion Types	28
Figure 3.8: XACML Policy Language Model (Moses, et al., 2005).....	30
Figure 3.9: Distinction between process, workflow, and their models (Leymann & Roller, Production workflow: Concepts and Techniques, 2000).....	32
Figure 3.10: Three dimensions of a workflow (Leymann & Roller, Production workflow: Concepts and Techniques, 2000).....	33
Figure 3.11: Core BPMN elements and its categories (Weske, 2007).....	35
Figure 3.12: Windows Workflow Architecture (Microsoft, 2011)	37
Figure 4.1: Dependency hierarchy of an agent, operation, and resource	39
Figure 4.2: An activity is a combination of an agent, operation, and resource	40
Figure 4.3: A Workflow Model and its Workflow Instances	41
Figure 4.4: Agent Lane.....	42
Figure 4.5: Agent Assignment Types.....	43
Figure 4.6: Multiple Agent Lines	43
Figure 4.7: Resource Types	44
Figure 4.8: Authentication Types	44
Figure 4.9: Authentication/Authorization with Security Token Service (Bayer, 2008).....	45
Figure 4.10: Authorization Types.....	46
Figure 4.11: Impersonation Types.....	46
Figure 5.1: Conceptual Security Framework.....	50
Figure 5.2: Modeling Layer Constructs.....	51
Figure 5.3: Retailer Lane with Operation Sequence	64
Figure 5.4: Annotating Retailer Lane with Security Controls.....	64
Figure 5.5: Retailer and Check Inventory Security Controls	65
Figure 5.6: Prepare Order and PDF Resource Security Controls	66
Figure 5.7: Place Order Security Controls	67
Figure 5.8: Delegation and Related Security Controls	68
Figure 5.9: Update Inventory Security Control.....	68

Figure 5.10: Security Requirements for Retailer Agent, Check Inventory, and Update Inventory.....	69
Figure 5.11: Security Requirements for Prepare Order and PDF Resource.....	70
Figure 5.12: Security Requirements for Place Order	70
Figure 6.1: Validating Prototype	73
Figure 6.2: General Prototype Architecture	75
Figure 6.3: External Services	76
Figure 6.4: Categories of the Custom Categories.....	78
Figure 6.5: Components of the AgentLane Activity	80
Figure 6.6: Workflow Tracking in WF (Bukovics, 2010).....	81
Figure 6.7: Channels and Bindings in WCF (Cibraro, Claeys, Cozzolino, & Grabner, 2010)	82
Figure 6.8: Mappings to the Retailer Database	83
Figure 6.9: The Claim-Based Security Scenario	84

List of Tables:

Table 5.1: Modeling Constructs & Security Goals Matrix.....	53
Table 5.2: Security Goals and Related Security Requirements Matrix.....	57
Table 5.3: Security Requirements and Related Security Mechanisms Matrix	63

List of Abbreviations:

ABAC	Attribute Based Access Control
BoD	Binding of Duty
BPD	Business Process Diagram
BPM	Business Process Modeling
BPMN	Business Process Model and Notation
CSRAC	Service-Oriented Role-Based Access Control
DAC	Discretionary Access Control
H-TRBAC	Multi-Hierarchy and Task-Role Based Access Control
KDC	Kerberos Key Distribution Center
LINQ	Language Integrated Query
MDA	Model-driven Architecture
MAC	Mandatory Access Control
OMG	Object Management Group
PKI	Public Key Infrastructure
QoS	Quality of Services
RBAC	Role-Based Access Control
SAML	Security Assertions Markup Language
SOA	Service-Oriented Architecture
SoD	Separation of Duty
SSL	Secure Sockets Layer
SSO	Single Sign-On
STS	Security Token Service
T-RBAC	Task-Role Based Access Control
UDDI	Universal Description Discovery and Integration
UML	Unified Modeling Language
WCF	Windows Communication Foundation
WF	Windows Workflow Foundation
WfMC	Workflow Management Coalition
WPF	Windows Presentation Foundation
WS	Web Services
WS-BPEL	Web Service Business Process Execution Language
WSDL	Web Service Description Language
XACML	eXtensible Access Control Markup Language
X-KISS	XML Key Information Services
X-KRSS	XML Key Registration Service
XML	eXtensible Markup Language

1 Introduction

Business environments are undergoing a fundamental change (Friedman, 2007). The complexity of the value chains is dramatically increasing, spreading more and more over organizations and continents. To succeed in the face of fierce competition the enterprises have to be fast and flexible.

Service-oriented architecture (SOA), as a new architectural style in information technology, has evolved to provide the required agility and flexibility. The concept of SOA focuses on building loosely-coupled applications made out of self-contained business functionality, also known as services (Weerawarana, Curbera, Leymann, Storey, & Ferguson, 2005).

Typical business process languages foster the implementation of services and provide a suitable modeling notation for business functionality. The associated runtime semantics provide support for the execution order and orchestration of the included business logic, such as services, backend transaction calls or human interactions. The actual business process models, however, remain independent of the realizing functionality. In contrast, as business processes become more complex, analysis of the realized business logic and associated constraints becomes significantly harder.

While the business expert concentrates on the specification of the business aspects of a process, other aspects, such as security, remain mostly neglected (if at all understood). The reasons for this are often lack of security knowledge on the business side as well as the unwillingness to “pollute” the business process with additional information perceived as non-business related. Security, and in a wider sense, compliance, of a business process, however, are critical to any organization, as for example an unauthorized modification of a node (which could e.g. represent a purchase order) may have major implications on the process outcome.

Accordingly, the visualization of security artifacts on the business process model level appears to be a promising approach regarding the communication between business and security experts, system-independent definition of security constraints, and model-driven propagation through a system stack.

In this thesis we present an approach that allows a business user to easily identify and specify security controls on the business process level. This approach is based on a prior analysis of core elements of business modeling languages common to business process dialects used today. Finally, in relation to these elements we identify possible matching types of security constraints that in turn enable the enforcement of related mechanisms at

runtime. A prototypical implementation is based on the motivating example that is introduced in the following section.

1.1 Motivating Example

This section presents a motivating example typically found in supply chains. In our scenario we analyze a basic supply chain including the involved participants, process steps, and data exchange. We focus on typical constraints in business process relationships and their implications on the technical implementation.

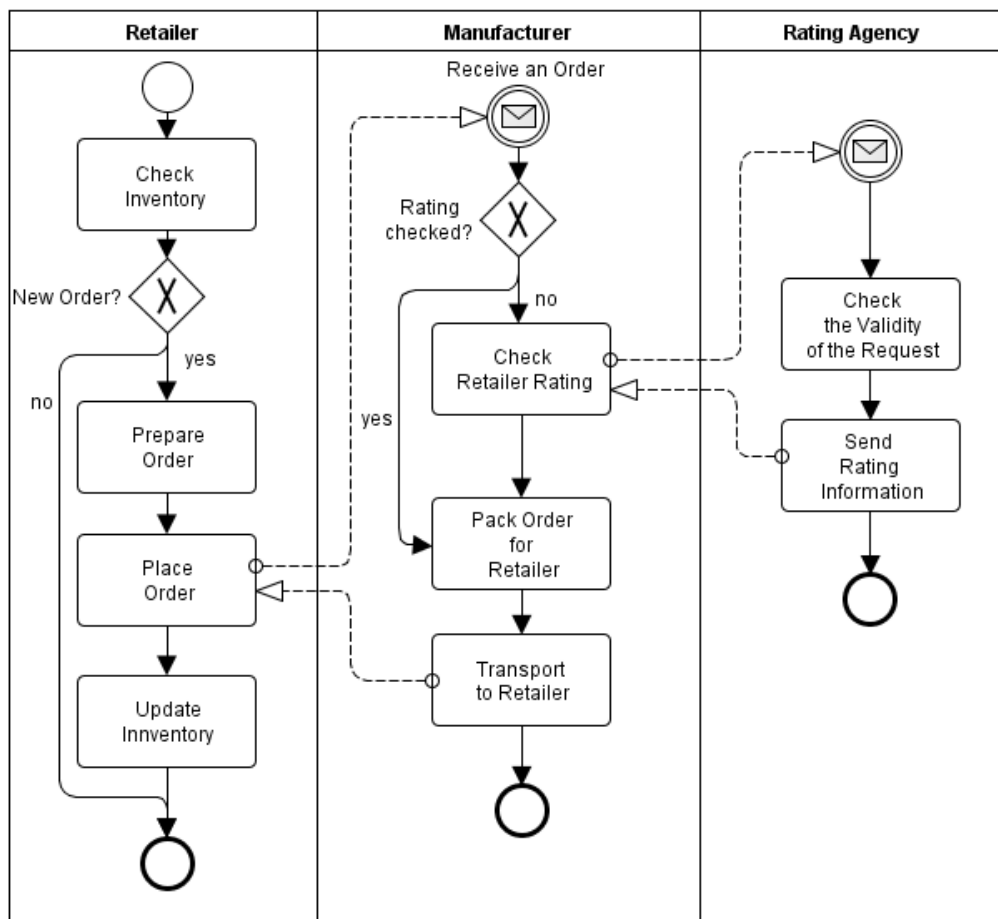


Figure 1.1: Motivating Example - Supply Chain Process

The retailer is a medium-size business that operates hundreds of grocery stores across several regions. The manufacturer is a typical large enterprise that has many business relationships with retailers. The manufacturer has its own logistics service that transports ordered goods to retailers. A retailer acting as a requesting party stores the goods temporarily before selling them to the end-consumers.

Facing the increasing competition on the regional market, our retailer is interested in efficient management of all processes relevant for business. The elaborated supply chain process, as shown in Figure 1.1, promises to reduce the administrative costs. However, the retailer is concerned about the exchange of sensitive information and the expected overhead in managing the security relevant aspects.

The business case involves the roles and constraints as detailed in the following Sections 1.1.1 – 1.1.3:

1.1.1 Retailer

The retailer submits a purchase request. As computer generated documents without a signature are not legally binding for all business-to-business transactions, some purchasing requests may contain a separate digitally signed purchasing order copy, which has to be in compliance with the Commercial Code.

Retailer Tasks:

- Check Inventory.
- Prepare Order.
- Place Order.
- Update Inventory.

Retailer Constraints:

- Only managers and employees from the purchasing department are allowed to place purchase requests.
- The purchase request is a piece of sensible information that must be kept confidential by manufacturers.
- The retailer must be able to authorize the manufacturer for placing a request to the rating agency.
- An audit of all placed orders must be ensured.
- All illegal access order placements must be prohibited and traced.

Technical Implications:

- The concept of roles at least on the level of the enterprise must be available.
- Execution of retailer tasks should require prior authentication.
- The document resource of the “Prepare Order” must be signed with a valid signature.
- The “Place an Order” task must be encrypted and require remote authorization.
- The “Check Inventory” and “Update Inventory” tasks require a local authorization.
- The retailer infrastructure must support logging and tracing to facilitate later audit.

1.1.2 Manufacturer

The manufacturer receives a purchase request. If the received order comes from an associated retailer, the manufacturer has to check the credit rating of the requestor. Only if the retailer identity is verified and the rating checks are all passed, the order is forwarded to the internal logistics service. Finally the order is packed and transported to the retailer.

Manufacturer tasks:

- Check Retailer Rating
- Pack Order for Retailer.
- Transport to Retailer.

Manufacturer constraints:

- The received order is a piece of sensitive information and must be protected from improper and unauthorized disclosure.
- The received order must be correct and complete.
- The received order must have contractual accountability. The identity associated with the placed order must stem from a valid retailer at the time of the request.
- The customer must have an acceptable rating.
- The passing of information to the rating agency is only allowed with an explicit permission of the retailer¹.

Technical implications:

- The retailer should be able to open a secure session with the manufacturer. The signature of the requestor must be valid.
- The privacy of retailer data must be ensured. Only allowed attributes of the retailer’s business data are allowed to be forwarded to the rating agency.
- The Check Retailer task must support impersonation².

¹ This concept is known as downstream usage control. It describes how the data should be treated after it was released. (Bussard, Neven, & Preiss, 2010)

- “Pack an Order for Retailer” and “Transport to Retailer” tasks may have context-based threats on involved resources.

1.1.3 Rating Agency

The Rating Agency is an organization that maintains a database of previous deficits in payments of the registered business partners. All registered business partners may place rating enquires about their business partners as long as they are in the database.

Rating Agency Tasks:

- Check the Validity of the Request.
- Send Rating Information.

Rating Agency Constraints:

- The identity associated with the rating information request must stem from a registered partner of the rating agency.
- The rating agency needs to be able to verify that the presented identity belongs to the associated requestor.
- The information about the rating of the business partner is confidential and should be shared only on explicit permission of the examined party.

Technical Implications:

- Unauthorized access to the rating’s agency database must be prevented.
- The rating agency should be able to trust the requesting identity.
- “Check the Validity of the Request” task must support impersonation.

1.2 Problem Statement

Business partners want services to be available when they are needed. The requested information must be correct and its integrity has to be assured. Furthermore, the interactions with the services must be transparent, allowing confidentiality, guaranteeing privacy of their personal information when communicating with underlying services and components. The orchestration of services must allow flexible access control policies for resources reflecting the dynamic nature of legitimated parties. In addition, despite the heterogeneity of underlying platforms, the autonomy and loosely-coupled manner of composed services, the ubiquitous accessibility of resources must be always ensured to a

² The concept of impersonation allows one party to transfer the right to use the own identity to another party.

mutually satisfying degree. However, dynamic and flexible binding of services conflict with conventional security mechanisms and models, making modeling and enforcing of security constraints a serious challenge.

Accordingly, there still remains a need for an efficient method that can:

- Assist the business expert by providing a unified well-understandable visualization of security relevant aspects;
- Encompassing several workflow-related security mechanisms and their variations;
- From convenient modeling up to the actual enforcement and implementation;
- Enabling flexible and highly customizable multi-domain access control policies;
- And reflecting the dynamic nature of legitimated parties in orchestrated scenarios.

Based on these observations, the main goals of this thesis are:

- An in-depth identification of security goals, related enforcing security requirements, and implementing security mechanisms;
- A conceptual workflow model, access control model, and a general security framework with several levels of abstraction, including detailed transition matrices to each layer;
- A unified visualization of the previous concepts;
- An architecture, design, and implementation of a validating prototype based on the motivating example and the elaborated framework.

1.3 Outline / Structure of the Thesis

As shown in Figure 1.2 this thesis is divided into seven chapters:

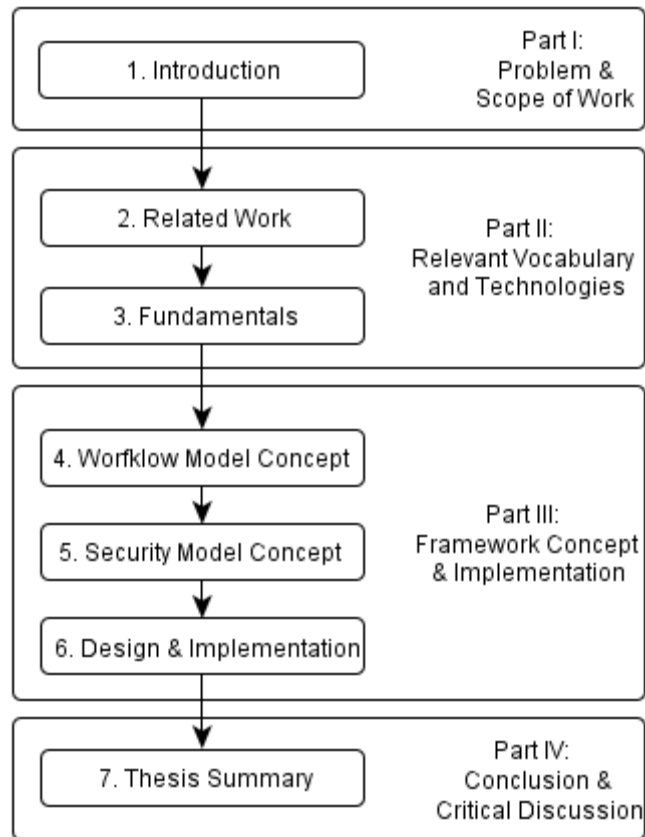


Figure 1.2: Thesis Methodology

Chapter 1 – Introduction: The introduction did provide a motivating example leading to a problem statement covered in this diploma thesis. Furthermore, we summarized the scope of our work and provide some useful definitions and conventions.

Chapter 2 – Related Work: This chapter introduces and discusses the current state of the art concerning the modeling and enforcement of security in workflows.

Chapter 3 – Fundamentals: In order to support further analysis of the problem statement possible, this chapter provides an overview of the main concepts and technologies covered in this thesis.

Chapter 4 – Workflow Model Concept and Specification: Based on the business modeling constructs discussed in Chapter 3 we present a general concept of a workflow model. The elaborated model elements are used as a basis for the security framework introduced in the next chapter.

Chapter 5 – Security Model Concept and Specification: This chapter investigates the core problems in security modeling and presents a general framework for modeling security in business driven environments. The presented security framework encompasses four layers: business modeling layer, security goals layer, security requirements, and security

mechanisms layer. For each layer a set of suitable components is discussed. All combinations to the underlying layer are elaborated and summarized in transition matrices. The chapter closes with a discussion on how the elaborated security framework can be applied to the motivating example introduced in the first chapter.

Chapter 6 – Design and Implementation: This chapter presents an overview of the architecture, design, and implementation details of the validating prototype that is based on the framework presented within the previous chapter.

Chapter 7 – Summary and Future Work: The final chapter summarizes the achieved results and introduces some critical discussions on this topic. Finally, the chapter ends with the future work in the context of this thesis.

2 Related Work

Besides available general introduction to computer security (Pfleeger, 2006), Web service security (Bertino, Martino, Paci, & Squicciarini, 2009), (Rosenberg & Remy, 2004), role-based access control (Ferraiolo, Kuhn, & Chandramouli, 2007) and business process management (Leymann & Roller, Production workflow: Concepts and Techniques, 2000), (Harvey, 2005), (Weske, 2007), we decided to review the most relevant papers on modeling security constraints at the business process level.

The relevant work is organized in three sections: security modeling, modeling constraints in workflows, and modeling roles and access control. The first section provides an overview of papers with a focus on security in workflows and general security modeling. As none of the investigated work presents a solid and integrated approach to the previously defined problem statement, our goals are to investigate theoretical contributions in security constraint modeling and the alternatives of role-based access control applicable to workflow management systems.

2.1 Security Modeling

(Jurjens, 2002) presents a formalized approach based on UML for expressing security-relevant information within the diagrams in a system specification. The model allows MDA-like formal analysis and verification of security mechanisms and protocols. The presented approach focuses on stereotypes, tagged values and constraints of the UML extensions mechanisms without any systematical identification and description of the security goals that can be expressed with the proposed model.

(Artelsmair, Essmayr, Lang, & Weippl, 2002) provide a comprehensive overview of available security modeling methodologies. Authentication as a security goal can take three forms: general user authentication, authentication of message content, and message origin authentication. Integrity as a concept to ensure consistency of data is divided into semantic integrity for being correct and complete and access control integrity for authorizing which users can modify which resources. The same applies to confidentiality, which denotes the protection of data from improper and unauthorized disclose. It is divided into semantic confidentiality and access control confidentiality. The first constraint defines different levels of secrecy of data, e.g. public or secret and the last specifies which users are able access which data. Further identified security goals are: non-repudiation which prevents an individual from denying having performed a particular action, auditing, anonymity for the absence of identity, and validity for digital contracts and signatures. After that they present

several security requirements that can enforce the identified security goals. The presented approach concentrates on UML without any systematical visualization on the introduced concepts and provides no information on how derived security requirements can be applied.

(Herrmann & Herrmann, 2006) present workflow related security goals and study their possible assignment to main categories of business process elements such as agents, roles, artifacts, and activities. Further they present a tool which allows domain experts to define the inspected security requirements expressed through UML activity diagrams. Further, they discuss how the abstract security goals can be checked for syntactical and semantical correctness. However, they do not provide any description of the enforcing and realizing mechanisms.

(Rodriguez, Fernandez-Medina, & Piattini, 2007) introduce graphical extensions, which allow the specification of auditing, integrity, access control, non-repudiation, privacy and attack harm detection. They also present to which Business Process Diagram (BPD) these security properties can be applied. The presented approach focuses only on BPMN and takes only graphical aspects into consideration.

(Wolter, Menzel, Schaad, Miseldine, & Meinel, 2009) address typical security goals, such as authentication or confidentiality. They propose a model-driven transformation approach from security requirements up to concrete security implementations. They further discuss a translation of security annotated business processes into XACML (eXtensible Access Control Markup Language) and AXIS2 security configurations. However, the graphical notation and the prototypical implementation are limited to the Separation of Duty (SoD) and the Binding of Duty (BoD) constraints. The authors do not provide any explicit graphical notations for the rest of the security constraints.

(Riesner & Pernul, 2010) analyze Business Process Modeling (BPM) with the focus on sensitive areas. After considering and comparing different sources in literature they derive the following security goals: integrity, availability, privacy, confidentiality, access control for enforcement of CIA (confidentiality, integrity, and availability) requirements, authentication, separation of duties for achieving compliance and as a measure to avoid abuse of privileges, audit, and adherence to business sequence flow. Further, they propose “security functions” and “security mechanisms” layers for enforcing and implementing the identified security goals. However, without any formal description of the containing elements and transition matrices, the presented approach is of little help to our problem statements.

(Wöhrle, 2008) presents an interesting extension for applying XML-Signature and XML-Encryption to different parts of a WS-BPEL process. The presented approach is especially practical in collaborative scenarios, but without focusing on related access control mechanisms the intended results could be very difficult to achieve.

2.2 Modeling Constraints in Workflows

(Bertino, Ferrari, & Atluri, 1999) divide constraints on role and assignment in workflow into three main categories: static, dynamic and hybrid constraints. While static constraints can be evaluated at design time, dynamic, as the name supposes, can be only evaluated at run time. Their activation depends on the execution history of the workflow. Hybrid constraints can be partially verified at the design time. However, they do not further investigate on the different types of SoD constraints.

(Schaad, 2003) extends the role concept by constraining the role membership, role activation and role use. He specifies the separation controls in two main dimensions static separation of duties (strong exclusion) and dynamic separation of duties (weak exclusion), where a further distinction between several kinds of dynamic separation is made.

(van der Aalst & Pesic, 2006) state that most business modeling languages are rather procedural and do not fit well with the autonomous nature of services. Thus, they propose a declarative language named DecSerFlow standing for Declarative Service Flow language, that should facilitate a better enactment and monitoring of the service flows. In their paper, they distinguish three groups of constraints: existence constraints, relation constraints and negation constraints.

2.3 Modeling Roles and Access Control

(Sandhu, Coyne, Feinstein, & Youman, 1996) present role-based access control (RBAC) model concept which is based on roles and roles hierarchies. They introduce some basic constraints such as: mutually exclusive roles also known as SoD constraints, cardinality constraints e.g. a maximum number of members in a role, and prerequisite constraints based on a role hierarchy e.g. the user A can be assigned to role A only if the user is already assigned to a role B.

(Oh & Park, 1999) propose a task-role based access control (T-RBAC) model that enhance the RBAC (Role Based Access Control) model through the integration of task and role.

The characteristics of tasks are the basis of access control and have a separate meaning from role.

(Feng, Jun, Hao, & Li, 2004) present a context-aware access control system service-oriented role-based access control model (CSRAC). In their model, access control can make its access control decisions by capturing security relevant environmental context, such as time, location, operation state, or other environmental information. Thus, it allows an access control which dynamically can grant and adapt permissions to users based on their current context.

(Yuan & Tong, 2005) introduce an attribute based access control (ABAC) model, which is based on subject, object, and environment attributes. Further, it addresses both mandatory (MAC) and discretionary access control (DAC) needs. The goal of the approach is to completely decouple the subject-object relation by independently defining attributes of subjects, objects and environment state.

(Emig, Brandt, Abeck, Biermann, & Klarl, 2007) combine the traditional role based access control (RBAC) approach with the ABAC model and introduce a new WSOA-aware access control metamodel. A problem of ABAC as the authors state is the indirection between objects and their access permissions. This indirection increases the complexity of appropriate policies. Thus, their model inherits from ABAC only the way service requestors are identified. The identification is based on a set of attributes and combined with RBAC: the role hierarchy and a set of permissions.

(Li, Li, Xie, Chen, Liu, & Pan, 2008) present a more fine-grained multi-hierarchy and task-role based access model named H-TRBAC. They apply the idea of hierarchies in roles on the task set, creating two multi-hierarchical sets. This model allows creating constraints where a task cannot be scheduled until its entire ancestors are completed.

3 Fundamentals

This chapter focuses on main technologies and services that are crucial for a better understanding of the concepts and principles discussed in the following chapters. This chapter is not supposed to give a complete introduction of all topics, but rather a general overview of the relevant topics such as security, workflow management and their related subtopics.

The first section presents SOA as the implementing architectural style. The methodology of SOA centers on service-oriented solutions and promises besides the flexibility and agility in definition of services, the facilitation in incorporating of security in technology-independent and loosely-coupled manner (Weerawarana, Curbera, Leymann, Storey, & Ferguson, 2005). The next section is dedicated to the main concepts and technologies in security. The evaluated standards are part of the implementation layer in the proposed security framework in Chapter 5. Finally, the Business Process Management Section gives the reader a synopsis of recent workflow related concepts and technologies that have been taken into account for the development of the conceptual workflow model introduced in Chapter 4.

3.1 Service-Oriented Architecture

Service-oriented architecture is a specific architectural style that allows exchanging data and participating in business processes regardless of the operating systems or programming languages underlying those applications (Newcomer & Lomow, 2004). The main concept behind SOA is the concept of services (Bertino, Martino, Paci, & Squicciarini, 2009). Services hide implementation details and make functionality available through standardized interfaces in a loosely coupled manner. Services usually provide additional metadata that describes such information in terms of business functionality, security requirements, and quality of services (QoS). Through the use of machine readable formats such as UDDI (Universal Description Discovery and Integration) for metadata, this information is discoverable by other systems. Being discoverable and available through public interfaces, services can be composed into new business-oriented services, allowing a new level of abstraction (Weerawarana, Curbera, Leymann, Storey, & Ferguson, 2005).

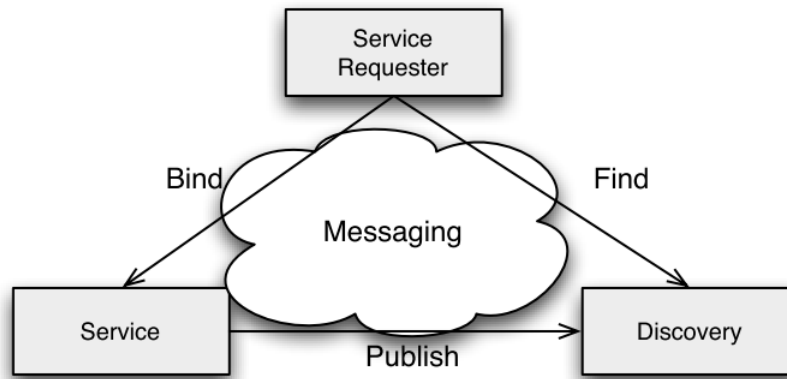


Figure 3.1: SOA-triangle (Weerawarana, Curbera, Leymann, Storey, & Ferguson, 2005)

The SOA Triangle as shown in Figure 3.1 summarizes the SOA cycle described in the following. A service provider publishes its services based on the definition schema of the discovery facility. The discovery facility aggregates the metadata making it available for a requestor. After a successful search the service requestor retrieves metadata for binding and consuming the service.

3.1.1 SOAP

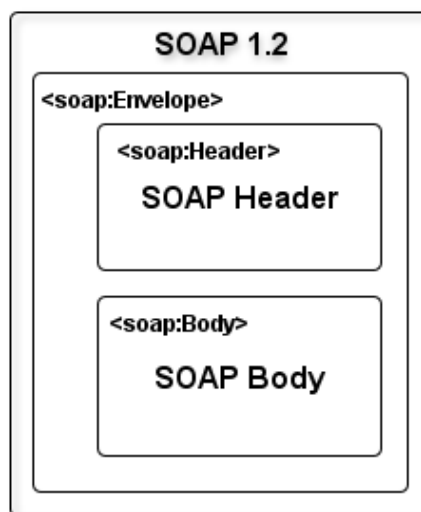


Figure 3.2: Structure of a SOAP message

The SOA paradigm makes an exhaustive usage of messages for inter-service and inter-vendor sharing and exchange of information. SOAP³ (Gudgin, et al., 2007) is a standard messaging protocol standardized by W3C. SOAP defines an extensible mechanism for message exchange between Web services. Each SOAP message is an XML document that

³ The acronym SOAP was originally defined as Simple Object Access Protocol. Starting with the version 1.1 the authors decided that SOAP would be no longer an acronym. (Weerawarana, Curbera, Leymann, Storey, & Ferguson, 2005).

consists of an envelope, header, and a body. The envelope is the root element of the SOAP message. The optional header element contains information about the metadata of the message and the body element contains the actual payload of the message. However, the extensible mechanism of the SOAP specification provides support for more sophisticated message exchange patterns (MEP).

3.1.2 Web Services Definition Language (WSDL)

The WSDL⁴ v.1.1 specification (Christensen, Curbera, Meredith, & Weerawarana, 2001) is the service representation language used to provide a uniform mechanism for describing service interfaces and specific bindings that the service supports. As shown in Figure 3.3, WSDL specification can be divided into two parts: an interface and an implementation part.

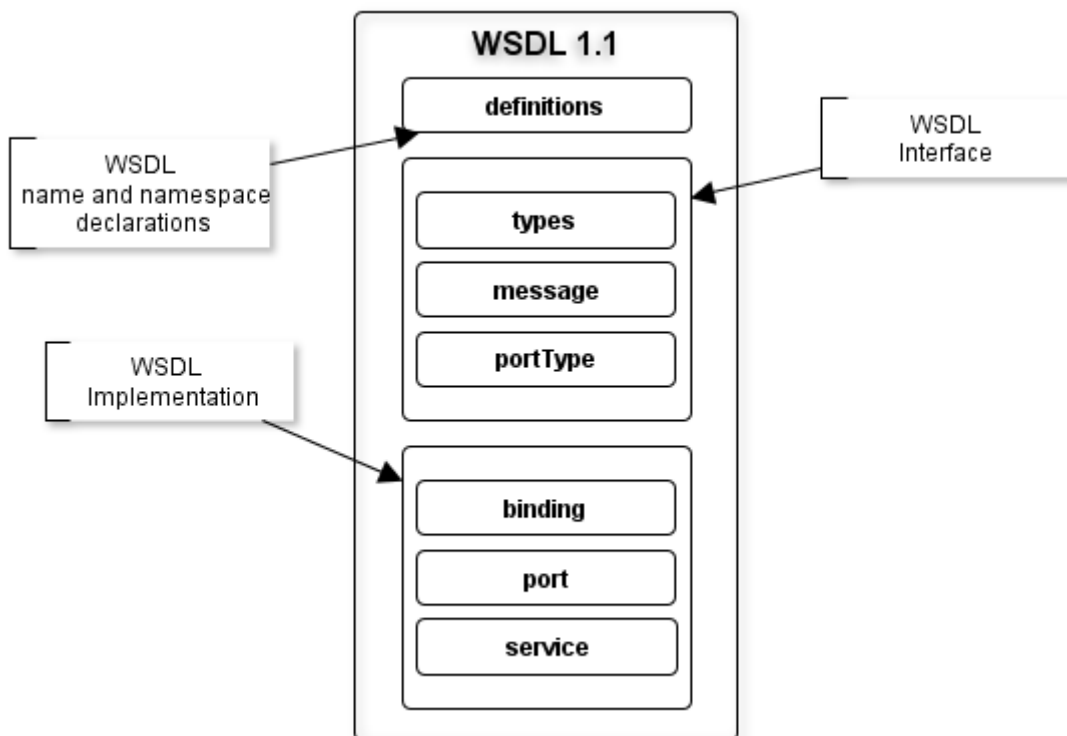


Figure 3.3: WSDL v.1.1 Specification

The interface part describes the general vendor-neutral structure of the service. It contains all the supported operations, its operation parameters, and the definitions or references of abstract data types. The data being exchanged between the service endpoints is specified as a part of messages. The collection of allowed operations is grouped into port types, which are similar to the interfaces in programming languages such as C# or Java.

⁴ The meaning of the acronym has changed in WSDL v.2.0 to Web Services Description Language. The version 2.0 is out of scope of this thesis due to incompatibility with WS-BPEL v.2.0.

The implementation part describes the bindings of abstract interface to its implementation details. The binding element specifies transport details for one or more interfaces. An endpoint associates a network address with a binding. A service groups endpoints together and associates it with the abstract interface.

3.2 Security

To protect all participants, the associated resources and their interactions in business process driven environments, several techniques must be considered. This section presents the well-known security requirements and their potential role in workflows. In addition, most important security mechanisms and the according standards used to enforce the security dimensions are introduced and briefly discussed.

3.2.1 Security Requirements

The three well-known security dimensions are (Bertino, Martino, Paci, & Squicciarini, 2009):

- **Confidentiality.** Confidentiality ensures that the transmitted data is protected from unauthorized disclosure, meaning that only authorized parties can view the contents.
- **Integrity.** Integrity ensures that the transmitted data remain unaltered during transmission, meaning that only authorized parties are allowed to modify data in authorized ways.
- **Availability.** Availability ensures that the data is delivered to intended authorized parties at appropriate times.

Further, the resources must be protected against unauthorized access. To address this issue, the requester of a resource must be identifiable. This requirement is also known as authentication. The specification of who can access which resources is called authorization. Access control, which will be addressed in the next section, captures the procedures and functions that control authentication and authorization.

3.2.2 Access Control

The main purpose of an access control model is to specify and enforce access control policies and to streamline the authorization management. To provide a better understanding of the concept behind role-based access we define the following notions, which are based on (Sandhu, Coyne, Feinstein, & Youman, 1996):

- **User.** A user generally refers to a human who interfaces with any computer system. A user may have multiple identities, such as a particular role or a custom attribute mapped to the user. These identities may be simultaneously active and are matched by an authentication mechanism.
- **Role.** A function or an attribute conferred on a user by an authority.
- **Role hierarchy.** A partial order relationship among roles.
- **Session.** A session is an instance of an access operation invoked by a user.
- **Subject.** A process acting on behalf of a user.
- **Object.** An object is any resource accessible on a computer system.
- **Operation.** An operation is a process invoked by a subject.
- **Permission.** A permission is an authorization of a subject to perform some operations on a subject.

As reviewed papers in related work demonstrate, there exist a lot of different approaches in access control models. What most access control models directly or indirectly share are the three well-known security principles (Sandhu, Coyne, Feinstein, & Youman, 1996):

- **Least privilege:** assigning only those permissions to a role required by a user to perform the assigned task.
- **Separation of duties:** Invocation of mutual exclusive roles to perform a certain task.
- **Data abstraction:** Instead of typical operating system read-write-execute permissions to provide more abstract permissions, such as credit and debit for an account object.

3.2.3 Security Mechanisms

Recent developments in Web service security allow designing and implementing security in business scenarios in a SOA compliant way, meaning to make the security mechanisms interoperable and extensible to address new requirements or new security technologies.

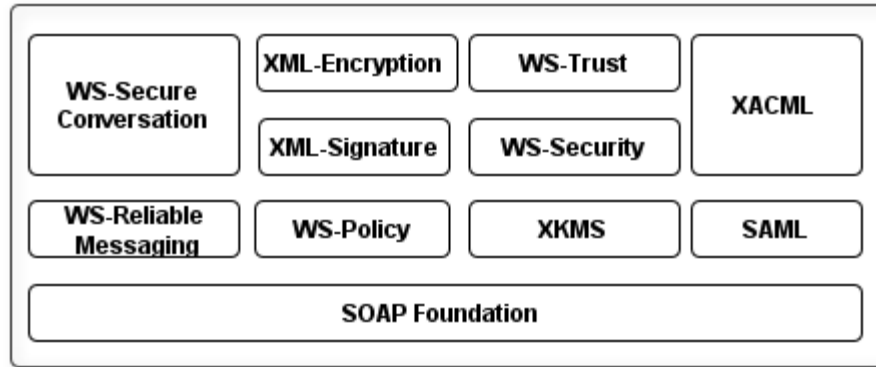


Figure 3.4: Web Services Security Specifications (Weerawarana, Curbera, Leymann, Storey, & Ferguson, 2005)

In order to provide a better understanding how to enforce the desired security requirements this section surveys the most promising existing or proposed Web service standards.

3.2.3.1 XML Signature

XML Signature (Bartel, Boyer, Fox, LaMaccia, & Simon, 2008) defines a standard for representing digital signature in a XML format. The main characteristics of the XML Signature standard are (Bertino, Martino, Paci, & Squicciarini, 2009):

- Signing data items consisting of complete XML documents, parts of it or any arbitrary binary data;
- Covering several resources with a single signature;
- Supporting enveloping, enveloped, or detached packing mechanisms. An enveloped signature means that the signature is inside the referenced resources, where an enveloping signature references data inside the signature. The detached signature, as the name suggests, references a resource that is separate from the signature. An example of a structure of a detached structure is shown in Figure 3.5.

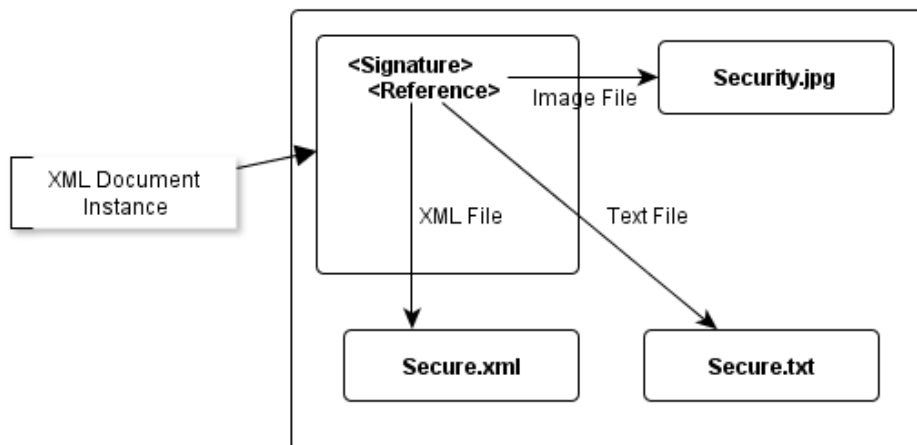


Figure 3.5: Detached Structure Referencing an External Resource (Rosenberg & Remy, 2004)

3.2.3.2 XML Encryption

XML Encryption (Imamura, Dillaway, & Simon, 2002) specifies how to represent and to reference information about the encryption mechanisms such the encryption key and encryption algorithm or information needed by the recipient to decrypt the sent data. It defines a standard model for confidentiality by hiding of data from anyone other than the private key holder. The XML Encryption standard supports (Bertino, Martino, Paci, & Squicciarini, 2009):

- Encryption of parts or complete XML documents or any arbitrary binary data;
- Separation of encryption information and encrypted data;
- Referencing mechanisms for addressing encryption information from encrypted data sections;
- End-to-end security, meaning allowing confidentiality at the application level for traversing multiple intermediaries.

3.2.3.3 The XML Key Management Specification (XKMS)

The XML Key Management Specification (Hallam-Baker & H. Mysore, 2005) defines a standard for distributing and registering public keys. XKMS does not depend on any particular underlying PKI (Public Key Infrastructure) implementation. A common approach is to use it in conjunction with the previously discussed XML Signature and XML Encryption standards.

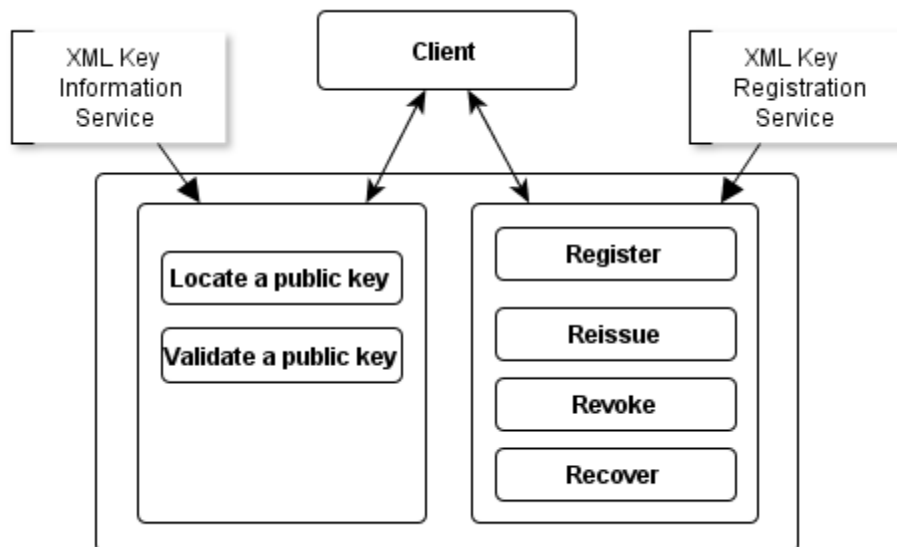


Figure 3.6: XKMS Services (Bertino, Martino, Paci, & Squicciarini, 2009)

As shown in Figure 3.6 the XRMS standard defines two services: the XML Key Information Services (X-KISS) and the XML Key Registration Service (X-KRSS). The X-

KISS service specifies functionality for locating and validating of public keys. To provide a simple example: a client may receive a document, where a signature is identified by the special *KeyInfo* XML element and is implemented by some security mechanism such as X.509 (Nadalin A. , Kaler, Monzillo, & Hallam-Baker, 2006).

Instead of resolving the key itself, the client may send it to the X-KISS service, which then resolves the required elements. However, the X-KISS service does not make any assertion concerning the validity of the binding between the data and the key. The X-KRSS service specification defines operations for registering, reissuing, revoking and recovering of the public keys. The register operation may bind additional information to the public key, such as name, identifier, or any further attributes defined by the implementation. Reissuing and revoking operations allow accordingly the registered keys to be reissued or to be revoked. The last operation allows recovering a key by providing information that allows authenticating the request to the X-KRSS service (Nordbotten, 2009).

3.2.3.4 WS-Security

WS-Security (Nadalin A. , Kaler, Monzillo, & Hallam-Baker, 2006) is a framework that defines end-to-end integrity and confidentiality for SOAP messages.

The framework makes use of the XML Encryption standard to provide confidentiality, and the XML Signature standard to enabling the message integrity. By defining the security-related information in the header of a SOAP message, the specification allows to sign and/or to encrypt SOAP message body elements, header blocks, or any combinations of them. The standard does not rely on any concrete security algorithms and allows using multiple signature formants, and multiple encryption technologies (Bertino, Martino, Paci, & Squicciarini, 2009).

Finally, WS-Security specifies a mechanism to include security tokens within SOAP messages. A security token represents a set of declarations (also known as assertions or claims) that are asserted by an authority and contain some security relevant information, such as a name, a key, or a privilege. WS-Security is neutral in relation to the security token formats and it can be extended to support any new security tokens. Currently five token types are supported in separate security profiles (Nordbotten, 2009):

- ***The UsernameToken Profile.*** The UsernameToken profile can be used to identify a requester by username. A password of the shared secret can be also provided. Further, the profile specifies a way to derive the shared key with a given username.
- ***The X.509 Certificate Token Profile.*** The X.509 Certificate Token Profile defines how to include X.509 certificates into a SOAP message. A X.509 certificate can be

used for authentication. The sender can be proved by signing the content using the corresponding private key.

- **The Kerberos Token Profile.** The Kerberos token profile defines how Kerberos tickets within security tokens can be applied to SOAP messages. Kerberos (Buckley, Hardjono, Yu, Hudson, R., & Tsitkova) is an authentication protocol that was developed by MIT to provide strong authentication by using secret-key cryptography.
- **The Rights Expression Language (REL) Token Profile.** The REL Token Profile defines how to include Rights Expressions expressed in XML Rights Management Language (XrML).
- **The SAML Token Profile.** The SAML token profile defines how to include SAML assertions in SOAP messages. SAML is discussed in more detail in the next section.

3.2.3.5 Security Assertions Markup Language (SAML)

The Security Assertion Markup Language (Ragouzis, Hughes, Philpott, Maier, Madsen, & Scavo, 2008) defines how to express assertions and how to exchange them between trusting parties.

An assertion is usually created by an asserting party. It conveys a piece of information in form of statements about the requesting party. As depicted in Figure 3.7, SAML supports three different kinds of assertions: authentication, authorization decision, and attribute statement decision types. All these assertions are issued by an asserting party that the relaying (receiving) party may trust.

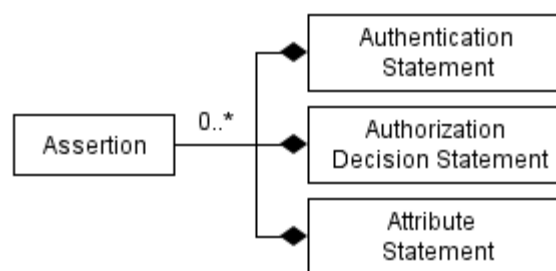


Figure 3.7: SAML Assertion Types

- An **authentication statement** conveys or references the authentication context. This context allows defining the type of authentication and the specific time at which the authentication was done.
- An **authorization decision statement** defines which specific operations on which resources the authorized party is entitled to or not. The authorization statement may reference further assertions and conditions, upon which the authorization was

made. Conditions allow placing restrictions such as restricting the potential relying parties or a validity period of the assertion.

- Finally, an ***attribute statement*** defines one or more specific attributes about the subject that can be utilized by a relying party.

In addition to assertion tokens, SAML supports various profiles for real-world scenarios, such as single sign-on (SSO) and attribute-based authorization (Bertino, Martino, Paci, & Squicciarini, 2009).

- ***Single Sign-On*** allows a subject, if authenticated once, to access additional resources, without any additional authentication. SAML enables the SSO through the communication of an assertion from one side to another one.
- ***Attribute-based Authorization*** allows deriving some characteristics of subject depending on which an authorization to some resources is granted or not.

(Ragouzis, Hughes, Philpott, Maier, Madsen, & Scavo, 2008) point out “*privacy generally refers to both a user’s ability to control how their identity data is shared and used, and to mechanisms that inhibit their actions at multiple service providers from being inappropriately correlated*”. As SAML is often deployed in scenarios where privacy must be explicitly addressed, the standard specification supports the following mechanisms:

- The establishment of ***pseudonyms*** (instead of an identity) between an identity provider and a service provider. Such pseudonyms are unique and cannot be shared between service providers.
- The establishment of ***transient identifiers***. These identifiers can be described as one-time tickets. The service provider is not able to recognize the same user on repeated request of a service.
- The establishment of ***claim-based facts*** of a user. These facts can be negotiated between the issuer and the identity provider (issuing party). The service provider may only recognize the claims made about the user, but not the identity itself.

3.2.3.6 eXtensible Access Control Markup Language (XACML)

XACML (Moses, et al., 2005) is a specification for defining access control policies. At its core, the XACML language defines the syntax for policies, the semantics for processing those policies, and the data flow model for the participants.

The most elementary element in a XACML policy is a rule. As shown in Figure 3.8, Figure 3.8: XACML Policy Language Model a rule consists of a target, effect, and condition. A target defines a set of resources, subjects, actions, and environments to which the rule

should apply. A rule always has an effect that defines the consequence of an evaluation to “true”. The effect allows two values: “permit” or “deny”. An optional condition may further refine the applicability of the rule defined by its target (Bertino, Martino, Paci, & Squicciarini, 2009).

The rules are combined in a policy, and multiple policies can be combined in a PolicySet. To define the order in which rules and policies are to be evaluated, XACML defines a set of rule and policy combining algorithms, namely: deny-overrides, ordered-deny-overrides, permit-overrides, ordered-permit-overrides, first-applicable, and only-one-applicable.

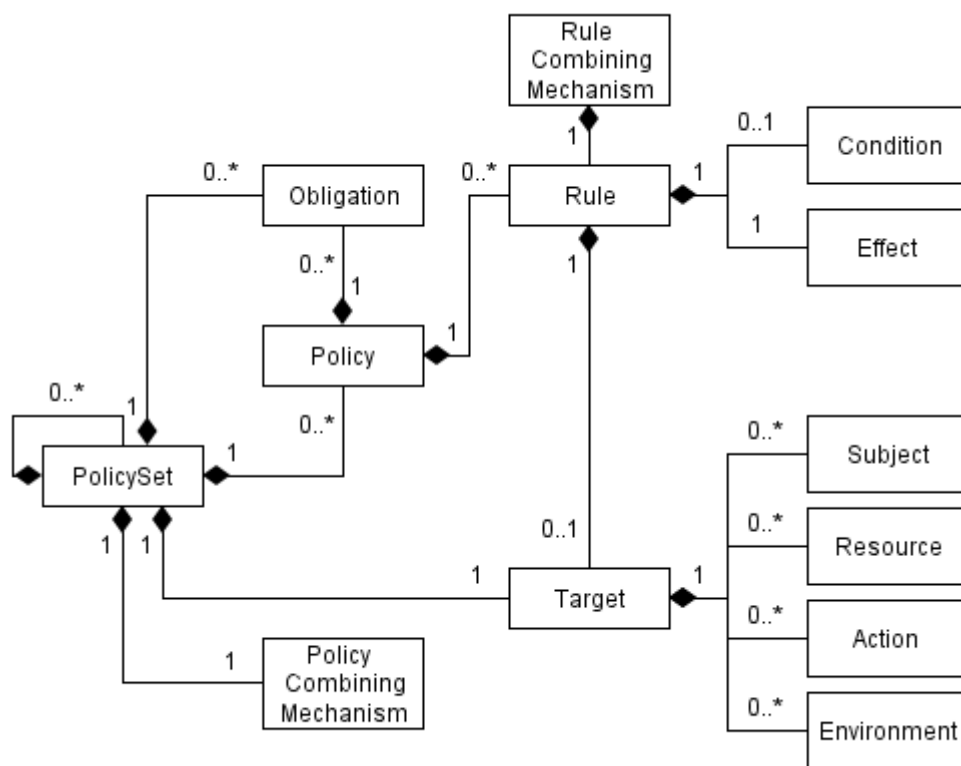


Figure 3.8: XACML Policy Language Model (Moses, et al., 2005)

3.2.3.7 WS-Trust

WS-Trust (Lawrence, et al., 2009) is a framework that augments WS-Security specification by defining functionalities for security token management and brokering trust relationships.

The general security model in WS-Trust is based on the exchange of security tokens that are issued by a Security Token Service (STS). The most common scenario is that a relying party may define policies that oblige the requestor to provide a predefined set of claims. If

the requestor does not provide the required claims issued by a trusted STS, the relying party ignores or rejects the request.

3.2.3.8 WS-SecureConversation and WS-ReliableMessaging

The WS-SecureConversations specification (Nadalin, Goodner, Gudgin, Barbir, & Granqvist, 2009) builds on top of WS-Security and defines extension mechanisms for establishing security context and sharing the session key material. The security context is typically used for exchanging SOAP messages for the lifetime of a communication session. The standard defines three different ways of establishing a security context. All of them reuse the mechanisms defined in WS-Trust. The first way is using the STS for the distribution of security context tokens. However, a security context token can be also created by a communicating party and propagated to other participants. The last approach represents the establishing of security context through a negotiation process among the communicating parties.

The WS-ReliableMessaging (Fremantle, et al., 2007) standard provides a modular and extensible mechanism for managing the reliable delivery of messages. The specification defines four types of delivery assurances:

- The *AtLeastOnce* assurance guarantees that each message is delivered at least once or an error must be raised.
- The *AtMostOnce* assurance guarantees that each message is delivered a most once.
- The *ExactlyOnce* assurance guarantees that each message is delivered exactly once. If a message cannot be delivered then an error must be raised by one of the endpoints.
- The *InOrder* assurance guarantees that a sequence of messages is delivered in the same order in which they were sent.

3.2.3.9 WS-Policy

WS-Policy framework (Vedamuthu, Orchard, Hirsch, Hondo, Yendluri, & Boubez, 2007) provides a general purpose model for expressing policies that may refer to any kind of Web services characteristics, domain-specific capabilities or requirements.

A policy in WS-Policy is defined as a collection of policy alternatives. These, in turn, describe a set of policy assertions. Policy assertions represent acceptable combinations and requirements, such as authentication or integrity assertions for SOAP messages that have to be met.

3.3 Business Process Management

The term Business Process Management encompasses the modeling and management of business processes. The WfMC (Workflow Management Coalition, 1999) defines a business process as “a set of one or more linked procedures or activities which collectively realize a business objective or policy goal, normally within the context of an organizational structure defining functional roles and relationships.”

Most enterprises have a repeatable set of processes that contribute to the value chain of an organization. These patterns of processes are so-called process models. A process model is a template from which a process or process instance with parameterized data is instantiated. Business processes that are carried in whole in part by computers are called workflow models and their instances, respectively, workflow or workflow instances as shown in Figure 3.9.

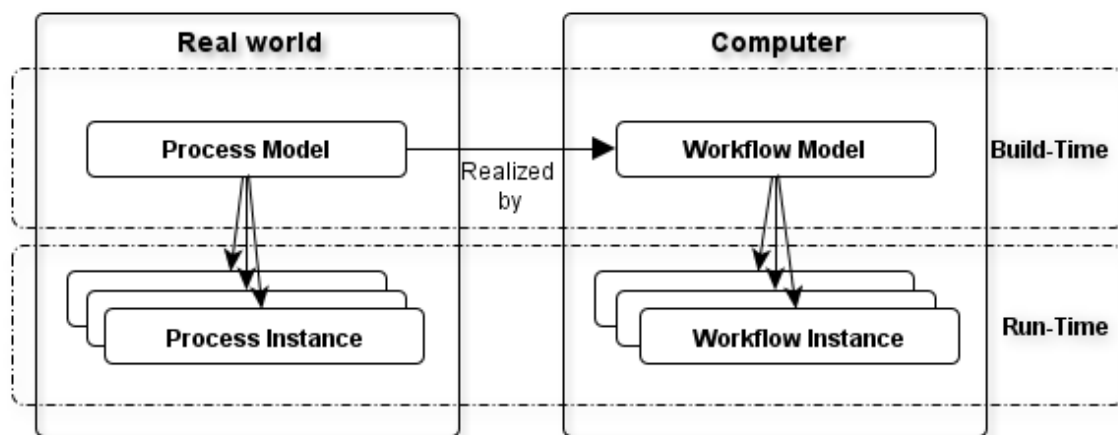


Figure 3.9: Distinction between process, workflow, and their models (Leymann & Roller, *Production workflow: Concepts and Techniques*, 2000)

For process models two lifecycles are generally distinguished (Workflow Management Coalition, 1999):

- **build-time**, when the layout and input/output requirements of a process are modeled
- **run-time**, when instances of the designed process are executed.

To provide a better understanding of the concept (Leymann & Roller, *Production workflow: Concepts and Techniques*, 2000) proposed a three dimensional view of workflows as shown in Figure 3.10.

The Process Logic Dimension of a workflow describes *what* in terms of what activities and in which order they need to be performed. Each process is composed of activities and/or sub-processes. An activity is an element that performs a specific function in a process

(Papazoglou, 2008). At a very abstract level there are two types of activities: automated activities that are enacted by a workflow engine and those which are not (manual activities). The activities are connected with the arrows which show the flow of control from one activity to the next.

The Organization Dimension defines *who* should perform each activity. Typically, an activity is assigned to a certain role or department in an organization. At runtime a special query is performed in order to identify the set of people authorized to perform an activity. If the activity does not have any human interaction, the workflow management may perform the activity on behalf of the requesting user.

The last dimension defines, *which* IT resources (programs, Web Services, etc.) are required to perform each activity within workflow instance.

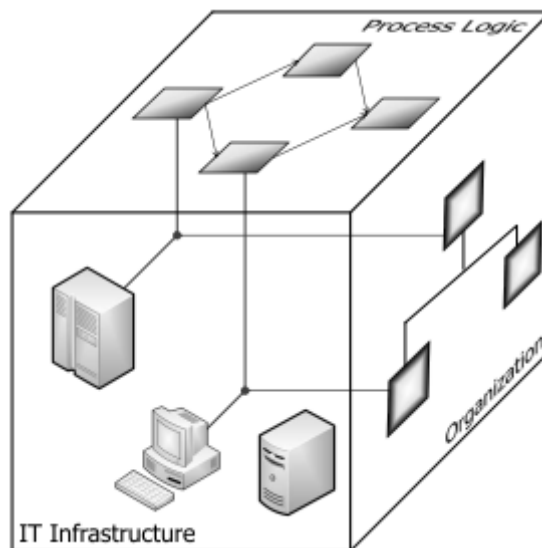


Figure 3.10: Three dimensions of a workflow (Leymann & Roller, *Production workflow: Concepts and Techniques*, 2000)

To define business processes a large number of business process modeling languages (Mili, Jaoude, Lefebvre, & Petrenko, 2003), (Lu & Sadiq, 2007) have been developed. The number of well-recognized and standardized languages is smaller, but on the other hand, it has to be considered, that most prominent languages such as BPMN, gained a lot of criticism (Muehlen & Recker, 2008), (Aagesen & Krogstie, 2010). Further, as some empirically validated case studies state: focus of the research in business process management is not always well aligned with the needs of industry (Indulska, Recker, Rosemann, & Green, 2009), (Recker, 2010). Thus, our approach is to review most promising technologies and standards, as detailed in the following Sections 3.3.1-3.3.3. Based on this information we introduce a general workflow model in Chapter 4 that provides a technology-independent view and can be applied to many recent technologies.

3.3.1 WS Business Process Execution Language (WS-BPEL)

The major execution language for business processes is WS-BPEL (Alves, et al., 2007). The language was standardized by the OASIS consortium and is widely used in the industry to provide interoperability between applications. WS-BPEL is an XML language that utilizes many open standards and technologies such as WSDL, XML Schema and XSLT. One of the most shortcomings is the fact that WS-BPEL provides no standard graphical notation for its language.

3.3.1.1 Variables

Variables are used to store data within a particular scope in a WS-BPEL process. The variables may be bound to: an input activity, such as <receive> or <pick>, to an outbound activity of a synchronous <invoke>, or to an <assign> activity.

3.3.1.2 Activities

The WS-BPEL language classifies its activities into two main categories: the basic (atomic) activities and the structured (compound) activities that may contain other basic or structured activities.

3.3.1.3 Scopes

In WS-BPEL a scope is a collection of activities that can have its own local variables and activities. This concept is similar to scopes in other programming languages.

3.3.1.4 WS-BPEL Extensions Mechanism

The WS-BPEL standard supports an extension mechanism for the extension of the existing or the definition of completely new activities. Typically, these extensions are declared with the help of separate XML Namespaces that may be optional or mandatory for the execution engine.

One of the most prominent extensions is BPEL4People, which has been published by BEA, IBM, Oracle, and SAP (Agrawal, et al., 2007). BPEL4People allows specifying users who have to perform the WS-BPEL activities directly in the process by defining user identifiers or groups of people. However, the specification does not provide any support for how the assignment is done or how to enforce constraints like separation of duties.

3.3.2 Business Process Model and Notation (BPMN)

The Business Process Modeling Notation (BPMN) v.1.2 (White, et al., 2009), which was developed and maintained under the coordination of OMG (Object Management Group), is

one of widely known notation (visual modeling) languages. As to time of this writing the final version of BPMN 2.0 was released. The new version promises to provide comprehensive operational semantics for executing processes. As the discussion about BPMN 2.0 potential use, its shortcomings, and its future role is going on (Völzer, 2010) (Leymann, BPEL vs. BPMN 2.0: Should You Care?, 2010), this thesis concentrates on BPMN v.1.2 as a typical representative of a visual modeling language.

The business process models in BPMN are expressed in business process diagrams (BPD). (Weske, 2007) divides the core modeling elements in BPD into four categories as shown in Figure 3.11:

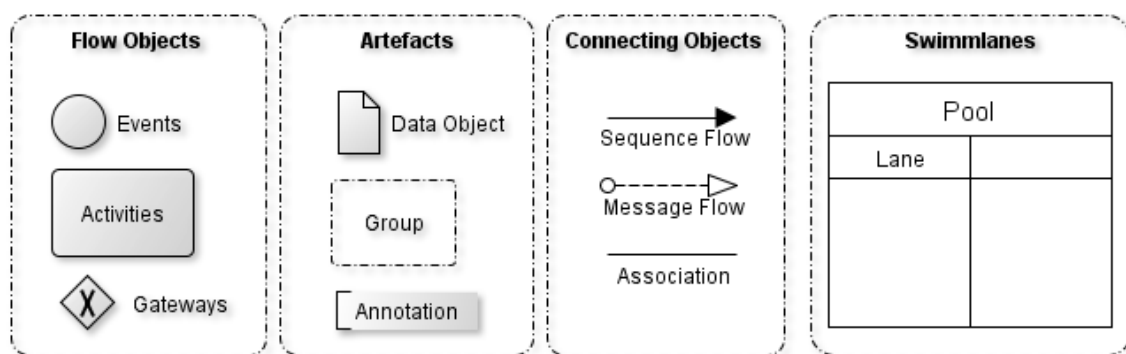


Figure 3.11: Core BPMN elements and its categories (Weske, 2007)

- **Flow objects.** Flow objects consist of activities, events, and gateways. Activities represent the work to be done. Events are used to represent states relevant for a business process. Gateways represent the split or join behavior in the control flow between activities.
- **Swimlanes.** Swimlanes represent organizational aspects and consist of pools and lanes. Pool is a notion for organizations and lanes represent organizational entities such as roles in an organization.
- **Artefacts.** Artefacts consist of data objects, groups, and annotations. These elements represent additional information associated with flow elements. As the BPMN specification states (White, et al., 2009) artefacts “*are not directly relevant for sequence flow or message flow of the process*”, and thus, serve only documentation and information purposes. Data object element is used to document the data object used by an activity. Text annotation provides additional information in a textual form. Group object does not have any formal meaning and is used to group any elements of a process.
- **Connecting Objects.** Connecting objects consist of sequence flow, message flow and association objects. These are used to connect the elements of the flow objects, swimlanes and artefacts. Sequence flow (control flow) specifies the execution order

of flow objects, message flow defines the flow of messages between swimlanes, and association is used to link artefacts to flow objects.

3.3.3 Workflow Foundation (WF)

The Windows Workflow Foundation (WF) is a technology for building workflow-centric applications introduced by Microsoft. The API provides a lightweight workflow engine that supports long-running processes, as well as workflows involving human interaction. As WF provides a considerable support of workflow patterns (Zapletal, van der Aalst, Russell, Liegl, & Werthner, 2009) and a highly customizable incorporation in any type of .NET applications and technologies, we used this technology for our prototypical implementation.

As shown in Figure 3.12: Windows Workflow Architecture (Microsoft, 2011), the Windows Workflow Foundation can be divided into three parts: runtime services, runtime engine, and activity library. Further, WF supports an extension mechanism that allows extending the existing activity libraries and services.

The standard activity library provides a long list of elements that can be used to compose a workflow. According to (Bukovics, 2010) the activities can be organized into the following categories:

- ***Procedural Flow***. The Procedural Flow category consists of flow control activities that determine activities that implement branching and looping.
- ***Flowchart***. The Flowchart provides free-form and flexible links between activities to control the flow of execution.
- ***Collection***. The Collection category consists of activities that allow working with collections of data.
- ***Messaging***. The Messaging category provides activities that allow communicating with other applications and Web services.
- ***Transactions***. The Transactions category provides functionality to ensure the consistency of transactions. This category includes compensation activities that ensure the consistency by undoing activities that have already been completed.
- ***Error Handling***. The Error Handling category provides support for the handling of exceptions. The activities are similar to the try/catch code blocks in programming languages.

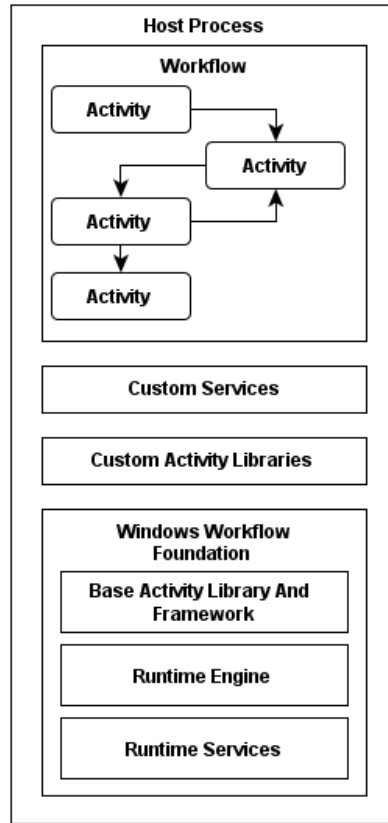


Figure 3.12: Windows Workflow Architecture (Microsoft, 2011)

4 Workflow Model Concept and Specification

In the previous chapters, some essential fundamentals in workflows and security were introduced. Based on the presented business modeling constructs a general workflow model and according access model are discussed. The elaborated models and their terminology are used as a basis for the security framework provided in the next chapter.

4.1 Conceptual Terminology

To capture complexity, different abstraction concepts exist. The same can be applied to a workflow model, which may be expressed in different ways. The consequence is that several workflow models and notions in literature exist. It is unquestionable that not even one of these models can pretend to be the only true, not to mention the fact that no complete representation will ever exist. As a consequence, this analysis does not pretend to present a complete and unique view on workflow models.

The main goal is to provide a sufficient, extensible and technology-independent model appropriate for further discussion. The presented model was motivated by the process viewing patterns presented in (Schumm, Leymann, & Streule, 2010).

For a better conception, this analysis begins with the atomic and semantically reasonable parts and gradually brings it to a concept of a workflow model as a whole.

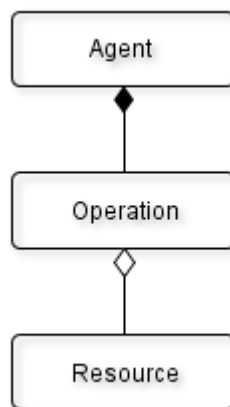


Figure 4.1: Dependency hierarchy of an agent, operation, and resource

Each piece of work on some materials must be performed by someone. An **agent**⁵ is someone who performs some **operations** on some **resources**. Thus, a dependency tree as

⁵ The meaning of an “agent” as proposed in this thesis is semantically equivalent to that of a “role” in (Leymann & Roller, Production workflow: Concepts and Techniques, 2000). But the notion of a “role” is misleading and collides with a general concept of an agent. An agent is what can be identified and assigned

shown in Figure 4.1 can be derived. A combination of an agent, an operation, and a piece of resources is called an **activity**.

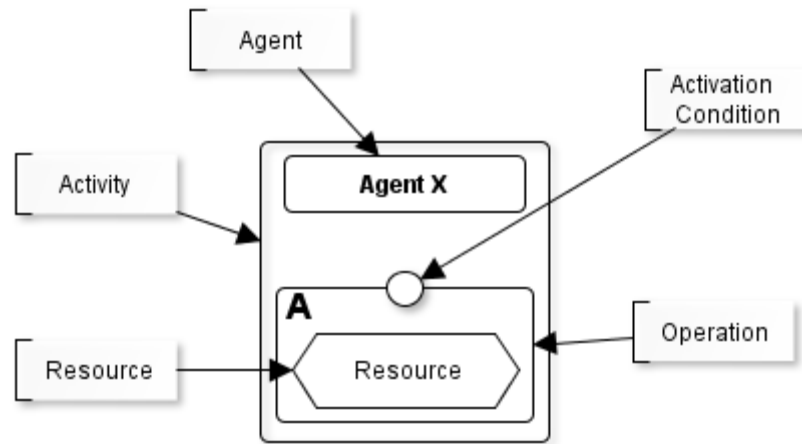


Figure 4.2: An activity is a combination of an agent, operation, and resource

To define an order and some attribute-based conditions of an activity, a set of predefined rules is required. This set of rules is called an **activity condition**⁶. The set of rules in an activity condition is divided into two parts: **activation conditions** and **execution conditions**.

The distinction between them is subtle but important: An activation condition is bound to a piece of information which must evaluate to true to invoke the associated activity, while the execution condition is a condition that holds a control of the activated activity or several iterations of it, as long as the execution condition is not fulfilled. To give an example, the activation condition of an activity B may be based on the fact of the previous execution of an activity A. The execution condition of an activity may be the fact that the utilization of a specific resource has to be at least 80 %. Only after these values have been achieved, the flow of control is passed to the next activities in the graph.

A **workflow** is a set of activities that are connected together in the form of a directed graph. A directed graph is a set of nodes and a set of ordered pairs of vertices called directed edges. In the case of a workflow: the nodes are the activities and the directed edges are the **arrows**. Those arrows and the activity conditions represent the **control flow** in a workflow.

Further, as shown in Figure 4.3, a distinction has to be made between a model and its instances. A **workflow model** is a template for some concrete instances of it. There may be multiple instances of a particular workflow model. An executing instance of a workflow

to perform a particular operation. The identity of it can be almost everything from IP address to a hierarchical property, or any combinations of it.

⁶ In a broader sense, an activity condition can be understood as a set of constraints on an activity.

model is called a **workflow instance**. An agent who is executing an operation in a concrete instance is called an **executing agent**⁷.

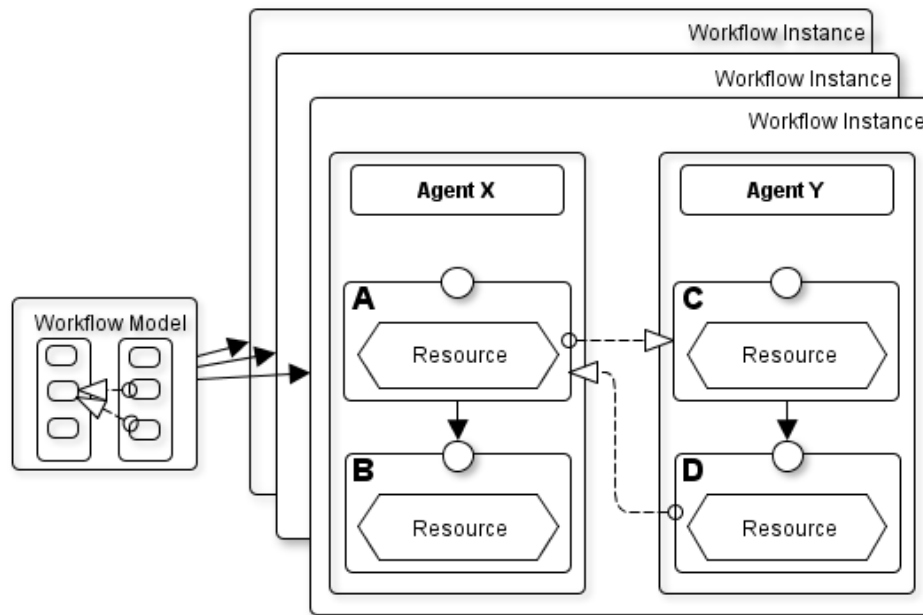


Figure 4.3: A Workflow Model and its Workflow Instances

In a sum, four main components that can be identified in a workflow model:

- **Agents.** An agent who performs operations on some resources.
- **Operations.** An operation that specifies a set of transitions on some resources.
- **Resources.** A resource is something that is target of some operations in a workflow.
- **Activity Conditions.** An activity condition that defines a set of rules for the activation and execution of an activity.

4.2 Conceptual Workflow Model

Each business process has a specific business goal. An **agent** is a generic name for someone or something that has to carry out the assigned operations. Further, it can be assumed that:

- There are several agents performing operations in a workflow.
- Each agent has its own set of assigned operations.
- Each operation is performed in context of some assigned agent.
- The agents may belong to different organizational structures.

⁷ An „executing agent“ is semantically equivalent to the notion of an „agent“ in (Leymann & Roller, Production workflow: Concepts and Techniques, 2000)

- The operations that cross organizational boundaries may require different communication and security mechanisms as the local ones.

4.2.1 Agent Lanes

The operations can be organized in **agent lanes**⁸. An agent lane implies an assigned agent and a set of operations and in a predefined order. The assigned agent has to perform the operations according the control flow within the agent lane.

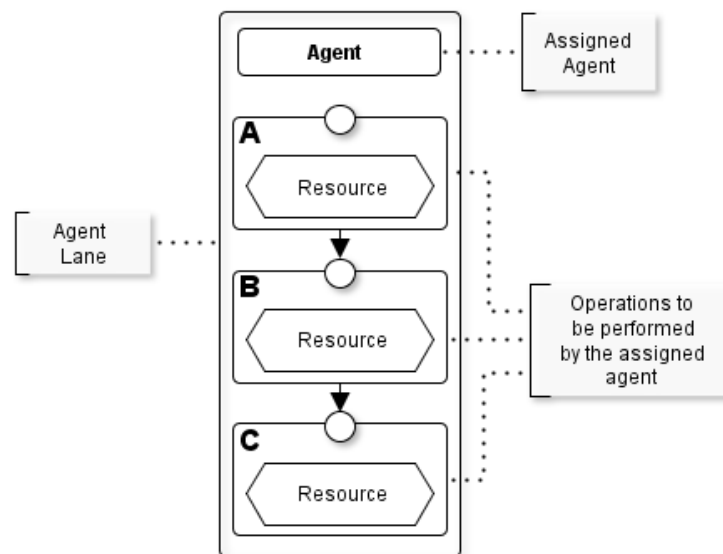


Figure 4.4: Agent Lane

To give an example, in Figure 4.4 the identity of the assigned agent has to perform operations A, B, and C in a sequence order. An **identity** is a piece of information that reliably identifies the executing agent.

4.2.2 Agent Assignment Types

As shown in Figure 4.5: Agent Assignment Types, the specification of an agent to identify that has to perform the defined operations may be defined in a **direct** or **indirect** way. The indirect agent assignment provides the ability to specify a group of suitable agents, which share similar characteristics (e.g.: Manager role, or a German IP address). The actual executing agent is only known at run-time. On the other hand, the identity of the executing agent made by the direct assignment is already known at build time (e.g.: UserID).

⁸ This concept is similar to the concept of a lane in BPMN.

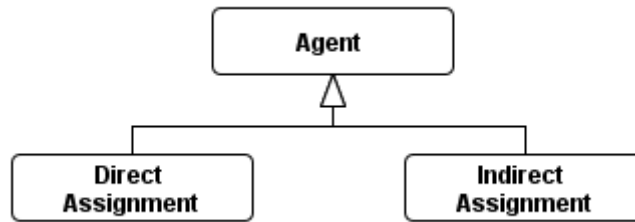


Figure 4.5: Agent Assignment Types

4.2.3 Operations

Each business process language, as may be noticed in the previous chapter, provides a comprehensive set of various operations. The BPMN language, to give an example, classifies the operations on the nature of the action to be performed. The operations are divided into: send, receive, user, manual, business rule, service, and script operations.

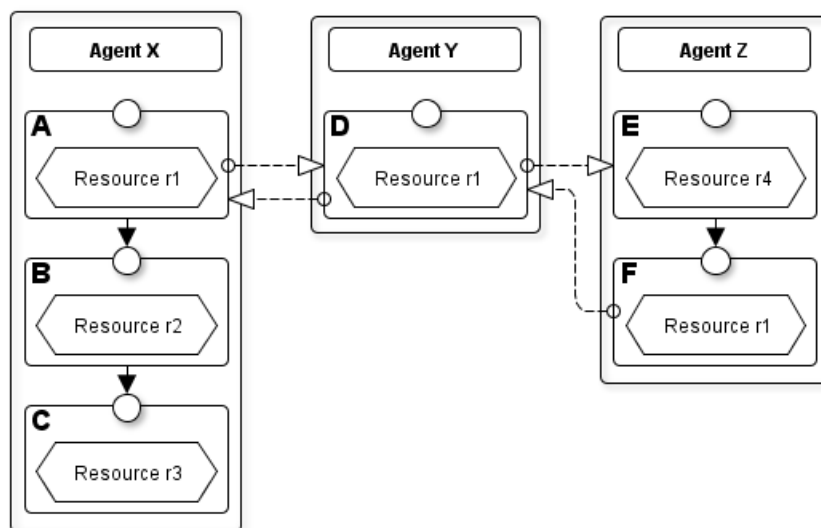


Figure 4.6: Multiple Agent Lines

In order to keep the concept lean and succinct, only few fundamental assumptions are made:

- The flow of activities is **implicit**, meaning that after activity conditions have been fulfilled, the assigned operation was completed, and not faulted; the control flow is passed to the next connected activity.
- As the difference between the **local and remote operations** is of great importance in security, the assumption is made, that the connection between security lanes is generally unsure. To illustrate that aspect, the flow can be visualized with a dotted line, as shown in Figure 4.6. This style of visualization has similarities to the message flow in BPMN.

4.2.4 Resources

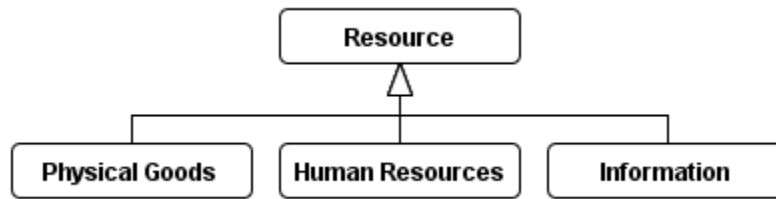


Figure 4.7: Resource Types

Resources can be classified into three main categories as shown Figure 4.7:

- **Physical goods** have a physical representation in a real world, such as a palette of perishables.
- **Human resources** are usually those resources that are used by executing agents to perform the assigned operations. To give an example, the applicants in a hiring process are the human resources, whereas hiring employees performing the operation “approve or reject applicants” are the executing agents.
- The last category of resources is **information**. In broader sense, everything that can be represented in a digital form may be handled as an information resource.

4.3 Conceptual Access Control Model

Due to their distributed nature workflow scenarios involve many security-critical operations. Based on security concepts in (Meier, et al., 2008) we present a general access control model in Section 4.3.1-4.3.3 that enables the enforcement of authentication and authorization in workflow related systems.

4.3.1 Authentication

Authentication ensures that the requesting party is really what it pretends to be. As shown in Figure 4.8, the authentication can be distinguished as: manual, user-name-based, certificate- and security token-based authentication.

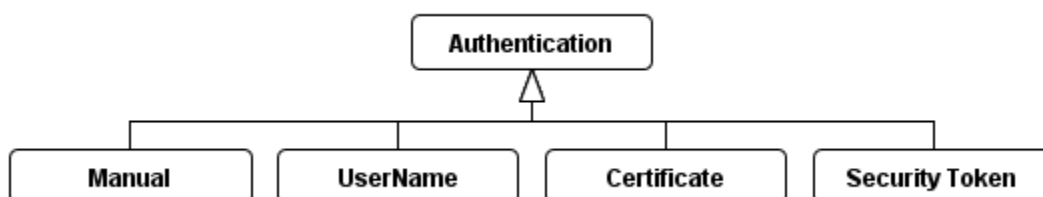


Figure 4.8: Authentication Types

- **Manual.** This type of authentication involves human or any physical resources for a successful request. To give an example, presenting a passport in an airport at the ticket counter involves airline employees for a successful check-in.
- **UserName.** UserName-based authentication requires the requesting party to provide a username and password. The requested services typically validate the credentials against a custom store. Typical examples are the Active Directory in Windows networks or the membership providers in database systems.
- **Certificate.** The requesting party has to provide a client certificate that the requested service looks up. For a successful authentication the requested service has to trust the issuer of the certificate.
- **Security Token.** The Security Token authentication is based on the WS-Trust specification. The basic idea is to externalize the authentication process to an external service, called the Security Token Service (STS), as shown in Figure 4.9. All needed information for a successful authentication is negotiated by the caller with the STS. The whole procedure consists of a total of three steps. The requesting party wants to access a resource that requires a security token. After successful negotiation (e.g. by presenting the required credentials) the STS issues a token that the requester passes through to the initial workflow service. The workflow service validates the token and authenticates the caller.

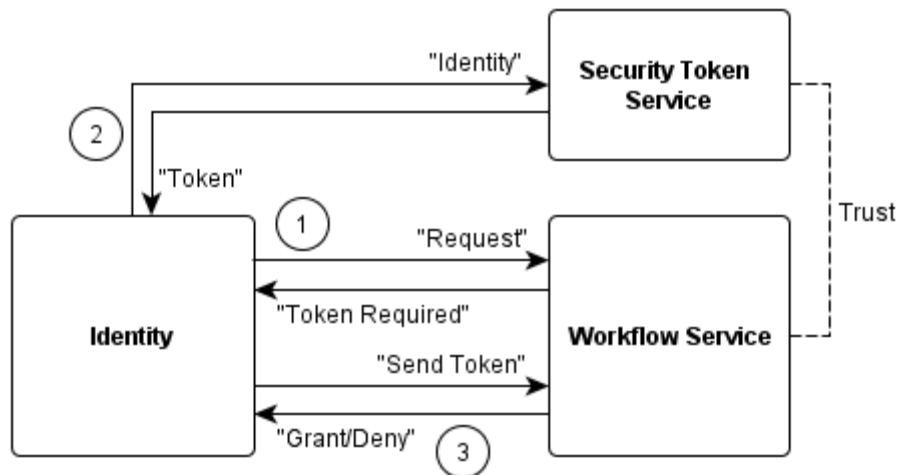


Figure 4.9: Authentication/Authorization with Security Token Service (Bayer, 2008)

4.3.2 Authorization

As shown in Figure 4.10 authorization can be divided into three types (Meier, et al., 2008): role-based, claim-based and impersonation-based⁹:

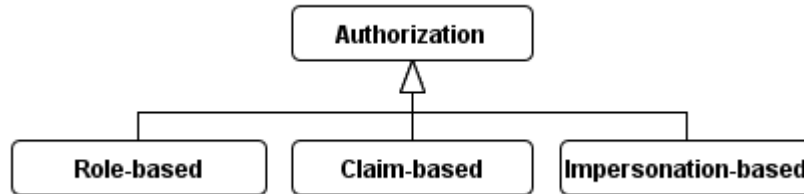


Figure 4.10: Authorization Types

- **Role-based.** The authorization decision is made on basis of evaluating the role membership of a caller.
- **Claim-based.** Claim-based authorization widens the role-based concept to any type of caller related information. A claim is a statement about a caller: for example, a name, an IP address, a key, or any further capability.
- **Impersonation-based.** Impersonation-based authorization is a concept, where the requested service impersonates the requestor's identity prior to call the associated resources.

4.3.3 Impersonation

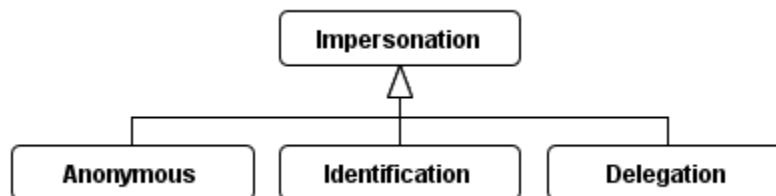


Figure 4.11: Impersonation Types

The concept of impersonation is based on the “Delegated Key Transfer” mechanism in WS-Trust specification (Lawrence, et al., 2009). The main goal is to allow one party to transfer the right to use the own identity to another party. A common usage is for example in business letters, which are signed on behalf of another person. Based on the elaborated classification in (Meier, et al., 2008), three different types of impersonation can be distinguished:

⁹ In conjunction with Windows access control lists also known as “resource-base authorization” (Meier, et al., 2008)

- ***Anonymous.*** The requested party authenticates the requesting party as anonymous and cannot obtain any information for impersonation.
- ***Identification.*** The requested party gets requestor's identity but must not use it for impersonation.
- ***Delegation.*** The requested party gets requestor's identity and is allowed to use it for impersonation.

5 Security Model Concept and Specification

Based on the previous conceptual workflow model, this chapter introduces a security framework that entails four layers: business modeling layer, security goals layer, security requirements, and security implementations layer.

The analysis begins with the construction of the main security goals and their mapping to the business modeling layer. The security goals are enforced by the security requirements; these in turn are realized with the technology-independent and platform-independent security mechanisms introduced in Section 3.2.3. For each layer a description of possible combinations to the underlying layer is elaborated. At the end of each section the devised concepts are summarized in transition matrices.

5.1 General Concept

To visualize and to incorporate security into business processes we propose to view security in several levels of abstraction as shown in Figure 5.1. The order of arrangement in layers is done with the focus on the domain experts. Typically, the domain expert is assisted with a toolset of modeling elements that provide support for building a visual representation of the desired business process. After completion, the finished model is used as a template for the execution or as an instrument of communication with the stakeholders, or both.

Based on these considerations, the first layer of the security framework is the “**Business Process Modeling**” layer. The presented layer is intended to provide a unified and readable representation of a business process applicable to any recent technology. The concept and its vocabulary base on the workflow model were presented in Chapter 4.

In essence, it can be assumed, that domain experts, who typically model the business processes, have often little knowledge of the security modeling. Thus, the resulting requirements can be defined as follows:

- Modeling of security should happen in a transparent fashion, so that the incorporation of security can be easily understood by domain experts.
- A business process modeling environment should actively assist the domain expert in designing secure business processes.

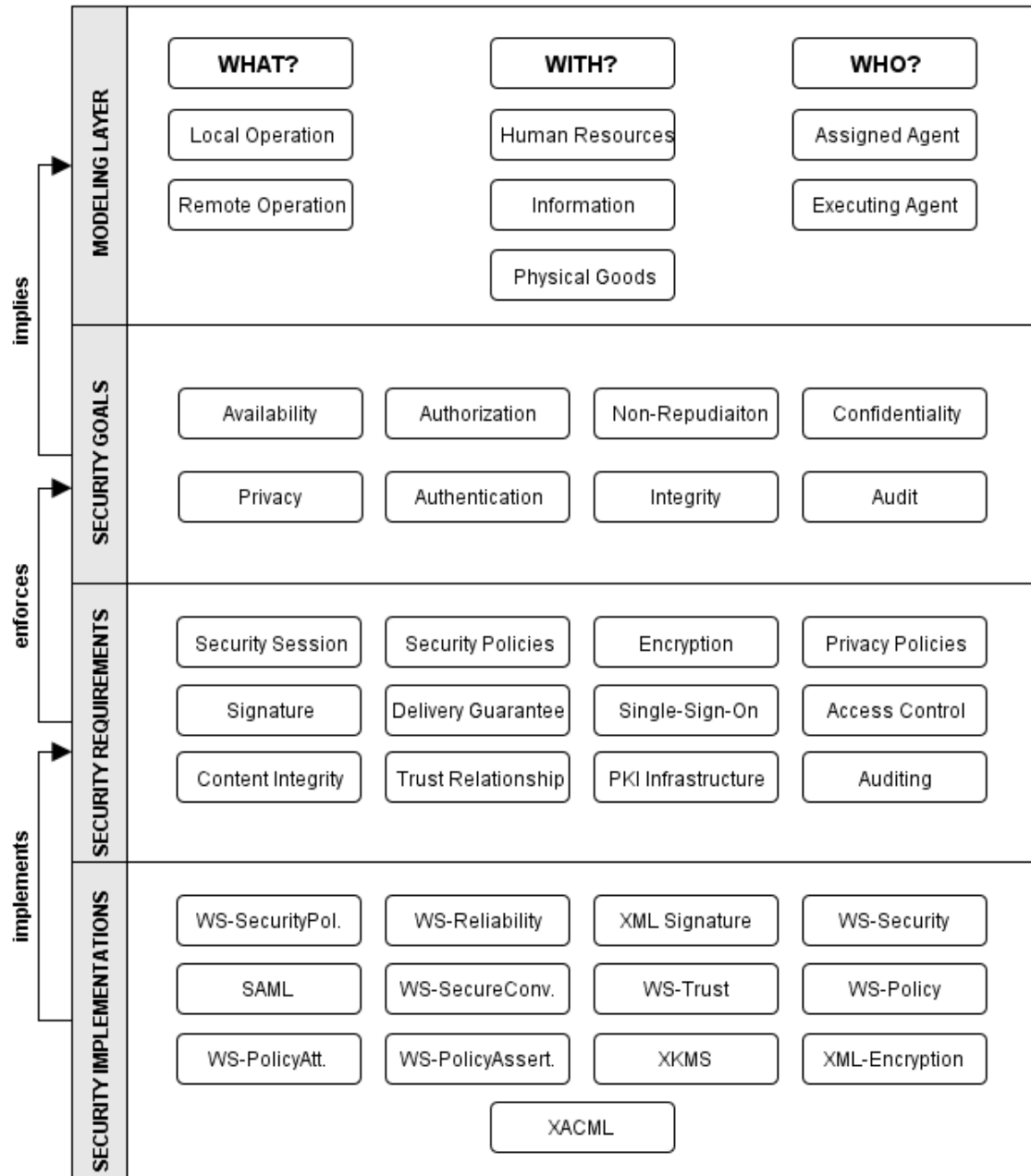


Figure 5.1: Conceptual Security Framework

The next layer is the “**Security Goals**” layer. The security goals that can be automatically or semi-automatically derived from the context of the modeled business processes are named as the implicit security goals. These goals can be seamlessly integrated into the modeling environment and usually require no additional training. Unfortunately, not all security goals fall into this category. Those security goals that require profound knowledge or active involvement of the domain expert belong to a category of explicit security goals. These types of requirements have usually a representational shape with configurable attributes similar to other business modeling elements in the modeling environment. A sophisticated design of the modeling environment must consider both approaches.

The third “**Security Requirements**” layer, as the name already indicates, describes the security requirements, which are used to enforce the security goals. The security goals can be realized in a multitude of ways. Thus, several security mechanisms have to be addressed and discussed for each element of the security requirements layer.

The last “**Security Implementations**” layer provides possible implementation mechanisms of the identified constructs in the requirements layer. The security mechanisms rely on the technology-independent specifications introduced in fundamentals. However, as in our motivating example can be extended or replaced with platform or implementation dependent mechanism.

5.2 Business Process Modeling Layer

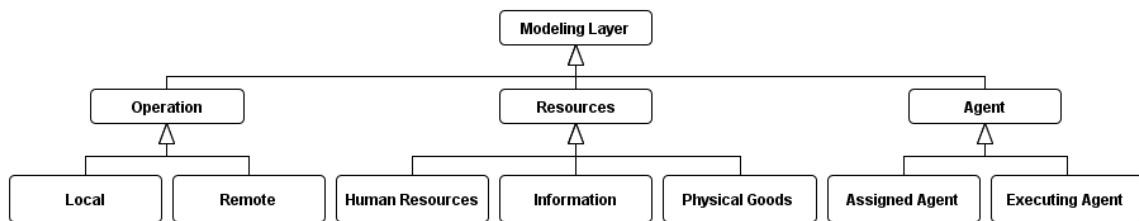



Figure 5.2: Modeling Layer Constructs


The concept of the incorporation of security requirements should be technology-neutral and applicable to other business process languages. To achieve the desired effect, the number of constructs should be kept to a minimum, but at the same time these constructs should be expressive enough, to represent the most essential aspects of a workflow environment. To provide a meaningful definition of the security requirements the aforementioned categories of the previous chapter must further be refined as shown in Figure 5.2.


The following tables summarize the presented business modeling constructs and the related security goals.

Name:	Agent (WHO Dimension)	
Definition:	An agent is someone who is assigned to perform some operations on some resources.	
Types:	Assigned Agent, Executing Agent.	

¹⁰ The icons in Sections 5.2-5.4 were adapted/reused from the OSA Icon Library 11.02, available at <http://www.opensecurityarchitecture.org> under Creative Commons Licence (CC BY-SA 3.0).

Description:	<ul style="list-style-type: none"> • Assigned Agent. An assigned agent is someone who is assigned at build time to perform an operation at runtime. • Executing Agent. An execution agent is someone who is performing an assigned operation at runtime.
Related Security Goals:	<ul style="list-style-type: none"> • Assigned Agent. Authentication, Privacy. • Executing Agent. Authentication, Privacy, Availability.

Name:	Operation (WHAT Dimension)	
Definition:	An operation is a unit of work performed by an agent.	
Types:	Local Operation, Remote Operation.	
Description:	<ul style="list-style-type: none"> • Local Operation. A local operation is usually under control of the workflow management system and exists within same boundaries as the deployed instance of a workflow model. • Remote Operation. A remote operation has usually a remote location and is invoked using various messaging protocols. 	
Related Security Goals:	<ul style="list-style-type: none"> • Local Operation. Authorization, Audit, Non-Repudiation. • Remote Operation. Authorization, Audit, Non-Repudiation, Privacy. 	

Name:	Resource (WITH Dimension)	
Definition:	A resource is something that is target of some operation(s) in a business process.	
Types:	Physical Goods, Human Resources, Information.	
Description:	<ul style="list-style-type: none"> • Physical Goods. Physical goods represent any kind of material resources that are utilized in a business process model. • Human Resources. Human resources are people involved in a business process as a target of the defined operations. • Information. Information is everything that can be represented in a digital form. Information resources encompass a wide range of resources: from a text file to a complete business process. 	

Related Security Goals:	<ul style="list-style-type: none"> • Physical Goods. Authorization, Availability. • Human Resources. Availability. • Information. Authorization, Confidentiality, Integrity, Availability.
-------------------------	--


<i>Security Goals</i>	<i>Modeling Constructs</i>	<i>Assigned Agent</i>	<i>Executing Agent</i>	<i>Local Operation</i>	<i>Remote Operation</i>	<i>Physical Goods</i>	<i>Human Resource</i>	<i>Information</i>
<i>Authentication</i>		x	x					
<i>Authorization</i>						x		x
<i>Confidentiality</i>				x	x			x
<i>Integrity</i>								x
<i>Availability</i>			x			x	x	x
<i>Audit</i>				x	x			
<i>Non-Repudiation</i>				x	x			
<i>Privacy</i>		x	x		x			


Table 5.1: Modeling Constructs & Security Goals Matrix


The related security goals can be summarized as shown Table 5.1. The agent must be authenticated for any further operation in a workflow system environment. Further, the executing agent must be available at execution time. The operation may require confidentiality and in case of a remote operation, privacy related requirements. The operations performed by an agent must be able to support non-repudiation, meaning preventing agents from denying having performed the according operation. The non-repudiation is an important prerequisite for the audit goal that ensures the collection of all security related information. The resources should be available when they needed. Finally, the information resources should provide an access control in form of authorization and support the confidentiality and integrity of data.


5.3 Security Goals


The previously discussed security goals involve the enforcement of several security mechanisms. The following tables present a context for each security goal under which the problem or the requirement occurs. The concept describes what preconditions must be met, how the goals can be realized with the related security mechanisms, and what constraints are to be considered.

Name:	Authentication	
Context:	Recognize and verify the identity of an agent.	
Constraints:	The requirement can be only applied to assigned and executing agents.	
Concept:	<p>The requirement is closely linked to access control policies. The authenticating party must be able to trust the authenticating subject. Further, there must be a possibility to negotiate further requirements, such as confidentiality and integrity requirements. The negotiated requirements may result in a need for a security session and a public key infrastructure (PKI). The authenticating party must be able to validate the signature of the authenticating subject.</p>	
Related Requirements	<p>Access Control Policies, Identity Authentication, PKI Infrastructure, Security Policies, Single-Sign-On, Trust Relationship.</p>	


Name:	Authorization	
Context:	Grant or deny a request made by an agent to controlled resources.	
Constraints:	The requirement can be only applied to operations and resources. The prerequisite is a successful authentication of the security subject.	
Concept:	<p>The requirement implies that access control policies to the related resources and operations are defined. Further, access control may require further security policies, which in turn, may result in a need for security context (eg to be exchanged sequence of messages) and a public key infrastructure (PKI). The authenticated identity must have a valid signature that can be trusted.</p>	
Related Requirements:	<p>Access Control Policies, PKI Infrastructure, Security Policies, Signature, Single-Sign-On, Trust Relationship.</p>	


Name:	Confidentiality	
Context:	Ensure the protection of information from improper and unauthorized disclosure.	
Constraints:	The requirement can be only applied to operations and resources. Confidentiality may require appropriate cryptographic key material or/and access control policies to support successful authorization of the security subject making a request to an object.	
Concept:	The requirement implies the use of encryption mechanisms. The enforcement of confidentiality can be expressed in negotiated or predefined security policies.	
Related Requirements:	Encryption, Security Session.	


Name:	Integrity	
Context:	Ensure the protection of information from unauthorized modification.	
Constraints:	The requirements can be only applied to resources.	
Concept:	The requirement implies the use of signature and mechanisms of reliable delivery of information.	
Related Requirements:	Content Integrity, Delivery Guarantee, Signature.	

Name:	Availability	
Context:	Ensure readiness of correct operations and resources to agents when they are entitled to.	
Constraints:	The requirement can be only applied to resources and executing agents.	
Concept:	The requirement implies the use of security policies and mechanisms of reliable delivery of information.	

Related Requirements:	Delivery Guarantee, Security Policies.
-----------------------	--

Name:	Audit	
Context:	Ensure the collection and organization of information to discover security violations.	
Constraints:	The requirement can be only applied to referring operations. The prerequisite is a successful authentication, maintaining integrity and non-repudiation of the performed operations.	
Concept:	The requirement implies successful authentication, which in turn requires a valid signature of the performing agent.	
Related Requirements:	Auditing, Identity Authentication, Signature.	

Name:	Non-repudiation	
Context:	Prevent an agent from denying having performed a particular operation.	
Constraints:	The requirement can be only applied to the referring operations. The prerequisite is a successful authentication and maintaining integrity.	
Concept:	The requirement implies a valid signature and a trusted relationship with communicating parties.	
Related Requirements:	Signature, Trust Relationship.	

Name:	Privacy	
Context:	Enable privacy-related policies.	
Constraints:	Privacy can be only applied to the remote operations.	

Concept:	Besides the encryption mechanisms privacy requires a relatively complex privacy infrastructure. For example, the privacy requirement may prescribe to the authentication service, which data of an agent should not be shared to parties that require authentication and authorization.
Related Requirements:	Privacy, Encryption.

<i>Security Requirements</i>	<i>Security Goals</i>	<i>Authentication</i>	<i>Authorization</i>	<i>Confidentiality</i>	<i>Integrity</i>	<i>Availability</i>	<i>Audit</i>	<i>Non-Repudiation</i>	<i>Privacy</i>
<i>Access Control Policies</i>		X	X	X					
<i>Auditing</i>							X		
<i>Content Integrity</i>					X				
<i>Delivery Guarantee</i>					X	X			
<i>Encryption</i>				X					X
<i>PKI Infrastructure</i>		X	X						
<i>Privacy Policies</i>									X
<i>Security Policies</i>		X	X						
<i>Security Session</i>				X	X				
<i>Signature</i>			X		X		X	X	
<i>Single-Sign-On</i>		X							
<i>Trust Relationship</i>		X	X					X	


Table 5.2: Security Goals and Related Security Requirements Matrix.


5.4 Security Requirements

Recent advances of Web service technology alleviate the implementation of security mechanisms in open and highly dynamic environments. Despite the heterogeneity of the


underlying security implementations those standards efficiently increase the interoperability and the manageability in business process driven environments. Thus, our framework basically focuses on recent and/or recognized open standards, without addressing, for example, transport security algorithms and implementations such as SSL (Secure Sockets Layer). As our prototypical implementation shows, such a technology-independent approach is not detrimental and can be easily extended or replaced with any platform or implementation dependent mechanisms.


The following tables present a context for each security requirement under which the problem may occur. The concept describes what preconditions must be met, how the requirements can be realized with the related security mechanisms, and what constraints are to be considered.


Name:	Access Control Policies	
Context:	Define access control policies for resources that need to be controlled.	
Concept:	An access control policy restricts which agents may perform which operations on which resources. Access control models, as presented in related work, provide a set of requirements for the access control policies. The derived rules, intended consequences, and the target, on which the rules are applied, may be very different in nature. The implementing mechanisms must be able to cope with various standards and allow flexible extension mechanisms for new access control profiles.	
Security Mechanisms:	XACML.	


Name:	Auditing	
Context:	Facilitate the capturing of security relevant operations providing irrefutable evidence of all security relevant events.	


Concept:	The audit requirement must allow analyzing logs, reports, and other information that allows indicating security violations. The support of auditing depends on the underlying implementation of the infrastructure, and is not directly addressed by any WS*- standard.	
Security Mechanisms:	Infrastructure.	


Name:	Content Integrity	
Context:	Provide data integrity, which allows verifying that data has not been manipulated in transit, and is correct and complete.	
Concept:	To ensure integrity, data must be protected from being modified, e.g. by using a signature. The semantic integrity, meaning that the data is correct and complete, is usually enforced by a schema, e.g. XML schema.	
Security Mechanisms:	XML Signature, XML schema.	


Name:	Delivery Guarantee	
Context:	Prevent data from being lost, duplicated, or reordered.	
Problem:	Ensuring delivery guarantee involves several quality of service assurances between the communicating endpoints. The implementing standard has to support: at most once, at least once, exactly once, and in order semantics.	
Security Mechanisms:	WS-Reliability.	


Name:	Encryption	
Context:	Provide a confidentiality of data.	
Concept:	The encrypting mechanism must provide an end-to-end mechanism that assures confidentiality of data traversing multiple intermediaries. The implementing mechanism should allow separating and referencing the encrypted data from the encryption information, and vice versa.	
Security Mechanisms:	XML-Encryption.	


Name:	Public Key Infrastructure	
Context:	Provide management for signature and digital certificates.	
Concept:	As public keys are the basic blocks for various types of data security is it particularly important to ensure a management of creation, distribution, and revocation in a consistent and uniform fashion.	
Security Mechanisms:	XML Key Management Standard.	


Name:	Privacy	
Context:	Capture and enforce privacy policies.	
Concept:	Identity may encompass some attributes and characteristics that should be shared only with the access point. It should be possible to define policies that prevent or restrict further usage, e.g. in single-sign-on scenarios.	
Security Mechanisms:	SAML, XACML.	

Name:	Security Policies	
Context:	Provide a model for expressing various types of security policies.	
Concept:	The security policies should be able to cover a broad field of security requirements, such as integrity and confidentiality of data. The requirements should be expressible in assertions that specify a behavior that is mandatory or optional requirement, or a capability of a policy target.	
Security Mechanisms:	WS-Policy, WS-SecurityPolicy, WS-PolicyAttachment, WS-PolicyAssertion.	

Name:	Security Session	
Context:	Provide a secure communication across multiple boundaries.	
Concept:	To establish and to manage a secure session, several preconditions must be met. At first, the session protocol must allow end-to-end scenarios with involvement of multiple communicating parties. Further, the security session must be able to provide a security context, which is shared among related endpoints for the lifetime of the session. The communicating parties belonging to the same session should be also able to reference the security context (instead of attaching it to each piece of data), in order to achieve a more efficient exchange of data. WS-SecureConversation, which builds on WS-Security and WS-Trust, satisfies all these requirements and allows different strategies for establishing a security context.	
Security Mechanisms:	WS-SecureConversation, WS-Security, WS-Trust.	

Name:	Signature	
Context:	Provide an end-to-end mechanism for signing data.	
Concept:	The signing mechanism must not only provide an end-to-end mechanism, but also support multiple packing strategies. The packing strategies must allow attaching and referencing signature to XML and Non-XML data files, or in case of XML, any part of it.	
Security Mechanisms:	XML Signature.	

Name:	Single-Sign-On	
Context:	Provide a single access point that grants or denies access to further resources and operations without being prompted for authentication each time.	
Concept:	The standard should supply mechanisms for providing a single access point including cross-domain environments, allowing agents set up requests to other domains without having to repeat the authentication each time.	
Description:	SAML, Infrastructure ¹¹	

Name:	Trust Relationship	
Context:	Provide an infrastructure for trusting relationships.	
Concept:	The standard should provide mechanisms for establishing and providing trust relationships across security realms.	
Description:	WS-Trust.	

¹¹ Our prototype implements an alternative approach of delayed authentication for operations within agent lane.

<i>Sec. Mechanisms</i>	<i>Sec. Requirements</i>										
	<i>Access Control Policies</i>	<i>Content Integrity</i>	<i>Delivery Guarantee</i>	<i>Encryption</i>	<i>PKI Infrastructure</i>	<i>Privacy</i>	<i>Security Policies</i>	<i>Security Session</i>	<i>Signature</i>	<i>Single-Sign-On</i>	<i>Trust Relationship</i>
<i>SAML</i>	x					x				x	x
<i>WS-Policy</i>							x				
<i>WS-Policy Attachment</i>							x				
<i>WS-Policy Assertion</i>							x				
<i>WS-Reliability</i>			x								
<i>WS-Secure Conversation</i>								x			
<i>WS-Security</i>	x										
<i>WS-Security Policy</i>							x				
<i>WS-Trust</i>											x
<i>XACML</i>	x					x					
<i>XML Encryption</i>				x							
<i>XML Key Management Standard</i>					x				x		
<i>XML Signature</i>		x							x		x

Table 5.3: Security Requirements and Related Security Mechanisms Matrix

5.5 Motivating Example

To outline how the previously discussed framework can be applied to support domain experts, we take the motivating example introduced in the first chapter. Based on the retailer operations we are going to investigate each layer of the security framework up to the actual implementation.

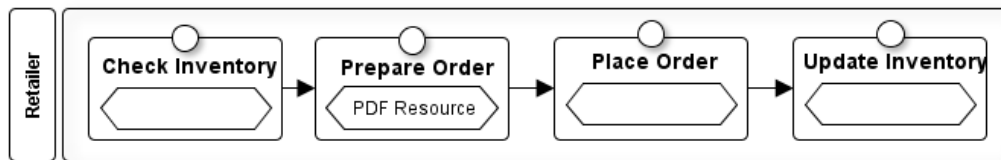


Figure 5.3: Retailer Lane with Operation Sequence

As described in Figure 5.3, a retailer lane consists of four sequential operations: Check Inventory, Prepare Order, Place Order, and Update Inventory. Check Inventory, and Update Inventory operations can be seen as inter-organizational operations, whereas Prepare Order and Place Order operations involve the flow of information crossing organizational boundaries.

5.5.1 Agent Security Goals

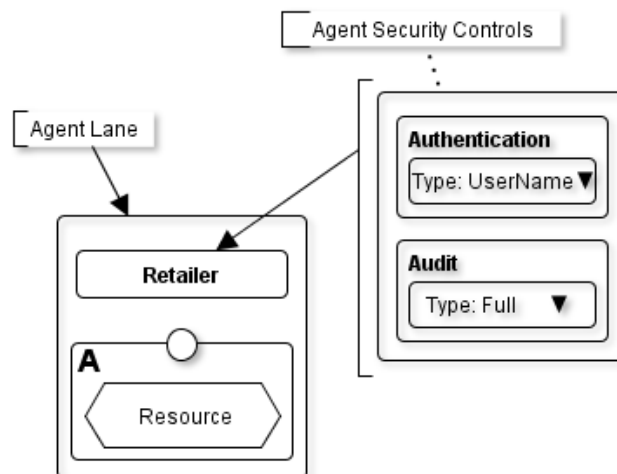


Figure 5.4: Annotating Retailer Lane with Security Controls

The modeling of the presented scenario begins with the placing of an agent lane discussed in Section 4.2.1. As soon the agent lane element is added to the workplace, the system can start assisting the domain expert with implicit security goals as proposed in Section 5.1.

As all security relevant operations require a prior authentication, the system may advise in form of an info dialog or a warning to place an authentication security control. The security

controls represent the configurable UI elements for defining and configuring the security goals.

Depending on the preferred type of authentication, the domain expert may choose the preferred type of based on the workflow access control model elaborated in Section 4.3. In our case the authentication can be fulfilled based on the username and password information of a user account in the referring organization.

An important constraint of the retailer in the scenario is that all illegal access order placements have to be prohibited and traced. Depending on the configuration of the considered workflow managing system this step may be initiated implicitly or on explicit demand. In our case, the domain expert sets the audit security control explicitly at the top of the retailer lane. As the audit can be only applied to agent operations, this placement has a meaning equivalent to annotating each operation of the retailer with an audit control.

5.5.2 Check Inventory Security Goals

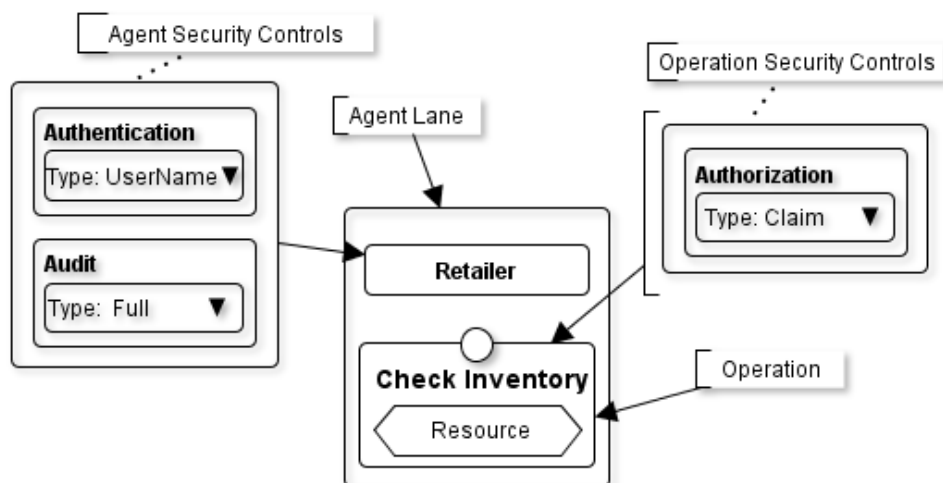


Figure 5.5: Retailer and Check Inventory Security Controls

After placing the retailer lane the Check Inventory operation, as shown in Figure 5.5, can be added. This operation requires an appropriate authorization. The workflow management system enforces this step by presentencing an error or an info dialog. Depending on the configuration and the implemented access control model, the system may advice the type of authorization to the referring operation. In our case it is the claim-based authorization discussed in Section 4.3.2.

The technical details of the underlying concepts for the claim-based security can be found in Section 6.4.4.

5.5.3 Prepare Order Security Goals

The subsequent Prepare Order operation is a representative example of custom authentication and authorization with separate security controls for the associated resources.

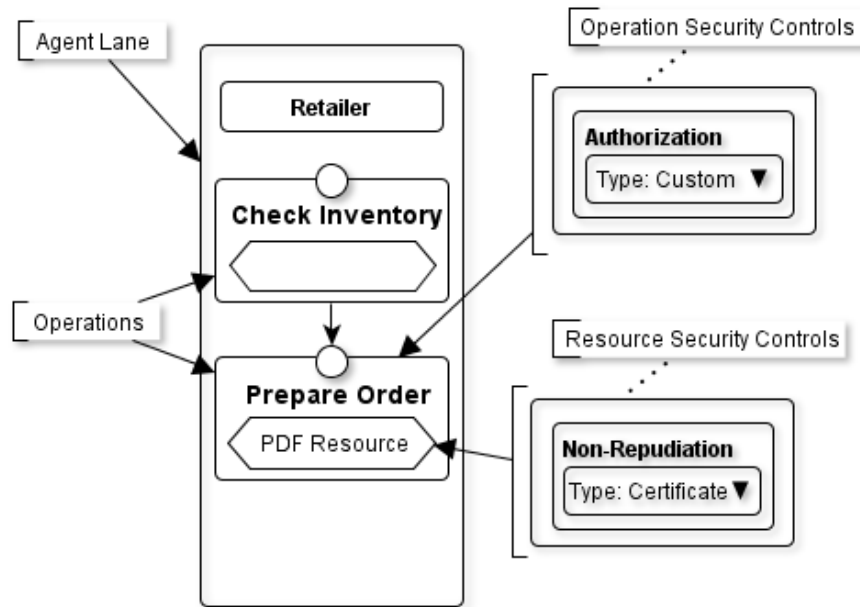


Figure 5.6: Prepare Order and PDF Resource Security Controls

The Prepare Order is intended to prepare a digitally signed purchasing order. In the presented scenario the prepared order is sent as a PDF document through an e-mail gateway. This gateway requires custom credentials that must be provided separately and are not of any use to the rest of the scenario. The workflow management system may be able to derive the context of the preconfigured Prepare Order operation and enforce the input of custom type credentials.

Besides the mail credentials, the PDF resource must be signed by the executing agent. To facilitate compliance with Commercial Code this step involves a non-repudiation security control. Depending on the type of authentication, the signature can be derived from the agent authentication or added manually to the non-repudiation control. In the presented scenario the agent authentication is username-based, so that the certificate has to be provided separately.

5.5.4 Place Order Security Goals

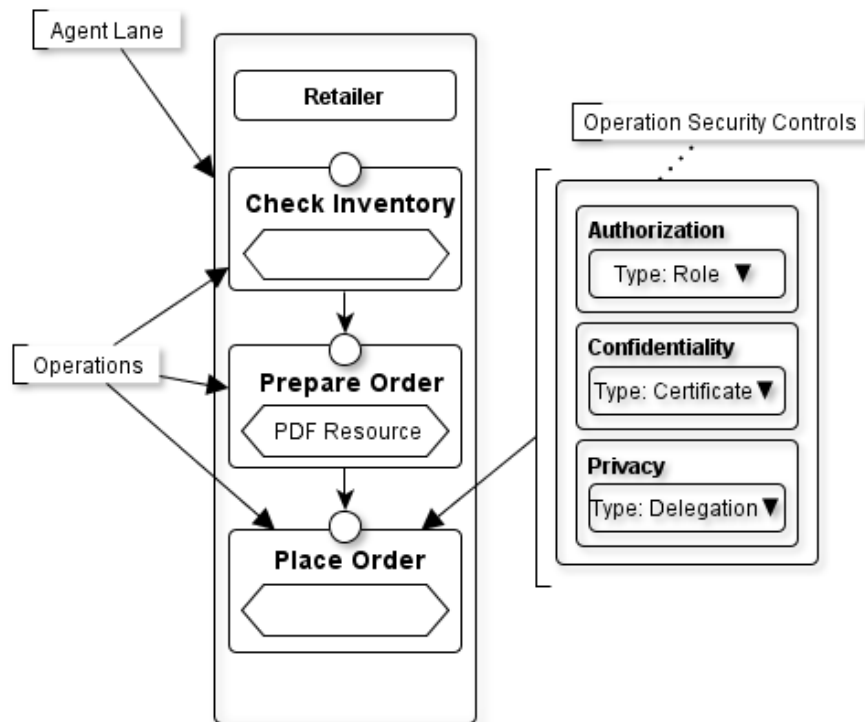


Figure 5.7: Place Order Security Controls

The next operation is a typical representation of a remote operation that involves, as shown in Figure 5.7, several security controls.

To begin with, a remote operation requires end-to-end security which in turn is enforced by the confidentiality security control. Depending on the underlying implementation the confidentiality type can be implicitly derived or as in our case be manually ensured by providing a certificate trusted by the referring party.

The authorization is based on the role the authenticated agent provides. To achieve the desired effect the system may imply a role-based authorization security control.

Finally, as the motivating scenario indicates, the manufacturer, who receives the order, has to carry a check of the retailer rating. This is done by sending a request to the rating agency. To prevent and detect fraud the rating agency service discloses data only to parties this information refers to. As shown in Figure 5.9, this can be done only on the behalf of the retailer, by placing a privacy security control. The manufacturer and the rating agency have to set the type of authorization for the referring operations to impersonation, as described in Section 4.3.2.

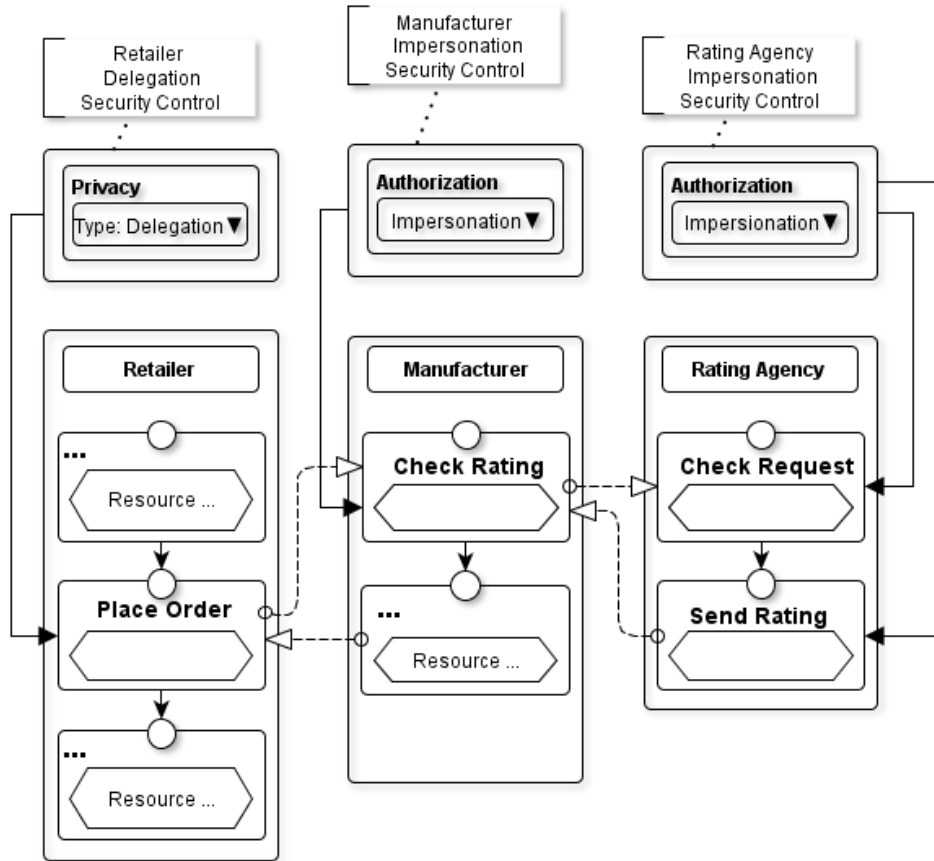


Figure 5.8: Delegation and Related Security Controls

5.5.5 Update Inventory Security Goals

As shown in Figure 5.9 the Update Inventory operation is analogue to the Check Inventory Operation. This operation also requires a claim-based authorization, which is authorization that can be enforced by the workflow management system enforces by presenting an error or an info dialog to the domain expert.

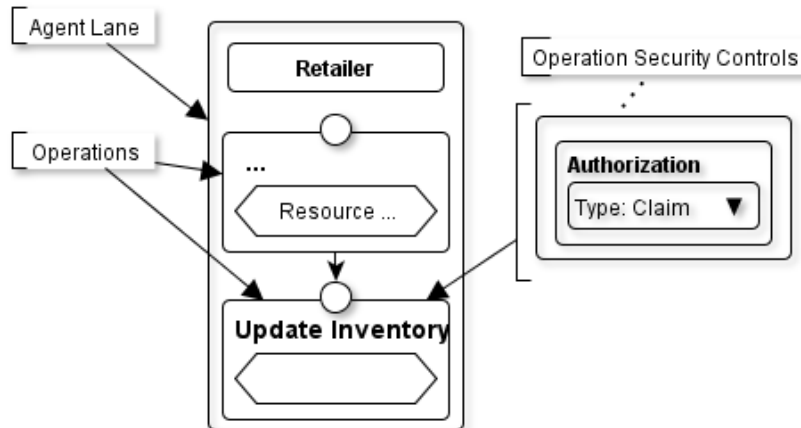


Figure 5.9: Update Inventory Security Control

5.5.6 Mapping Security Goals to the Security Requirements

The additional abstraction layer between the Security Goals and the Security Implementation layer brings the advantage of flexible and highly-customizable implementation mechanisms of the identified security goals. The Security Requirements layer can be mapped not only to the WCF Binding elements and custom implementations, as in our prototype, but also to many recent technologies (e.g. Apache Rampart Axis2 Security Module) (Apache Software Foundation, 2011) that supply the identified security features.

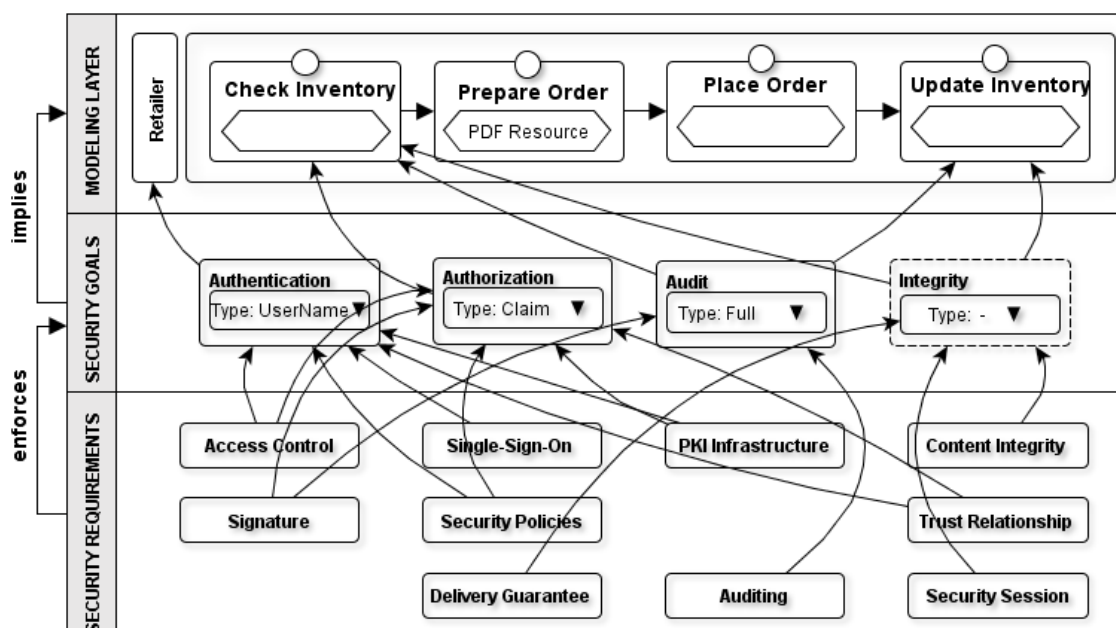


Figure 5.10: Security Requirements for Retailer Agent, Check Inventory, and Update Inventory.

To identify the related security requirements we apply the transitions defined in Table 5.2 of our security framework. Instead of listing the related security requirements for each security goal, we group them into categories that share the same functionality, as shown in Figure 5.10, Figure 5.11, and Figure 5.12.

The security requirements for the Check Inventory and Update Inventory encompass the claim-based scenario and involve the intervention of a Security Token Service, as described in Section 4.3.1. The Prepare Order and PDF Resource scenario refer to the mixed security implementations. The modeling elements affected by the mixed security scenario share the characteristic that they cannot be accessed or manipulated in a technology-neutral manner. The last Place Order scenario refers to the role/identity-based security implementation and builds upon the username credentials and according roles.

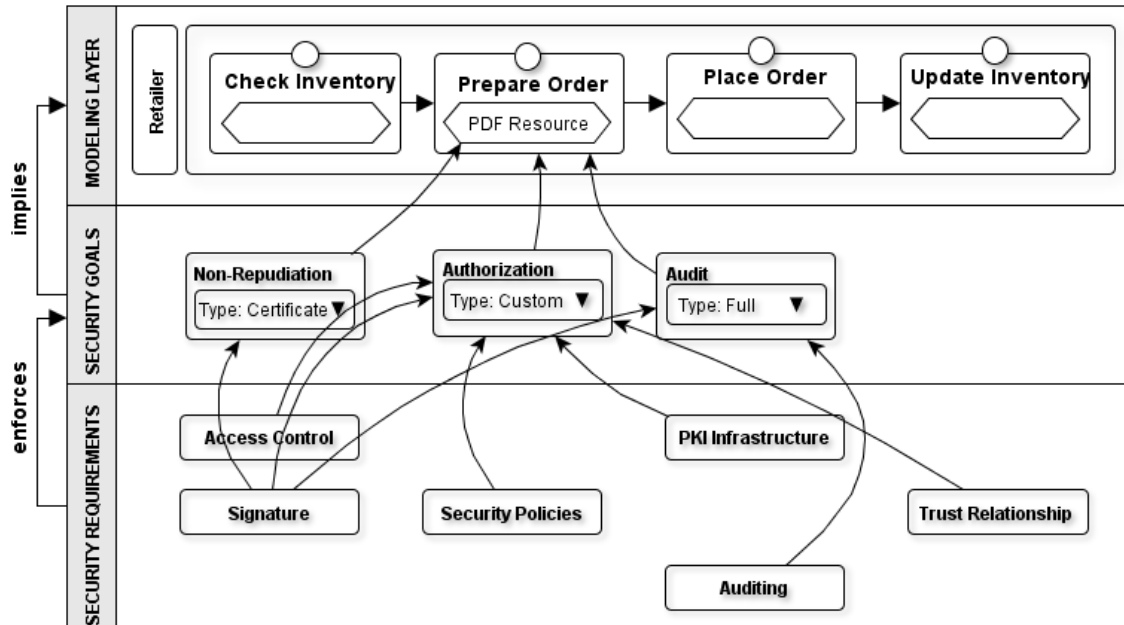


Figure 5.11: Security Requirements for Prepare Order and PDF Resource

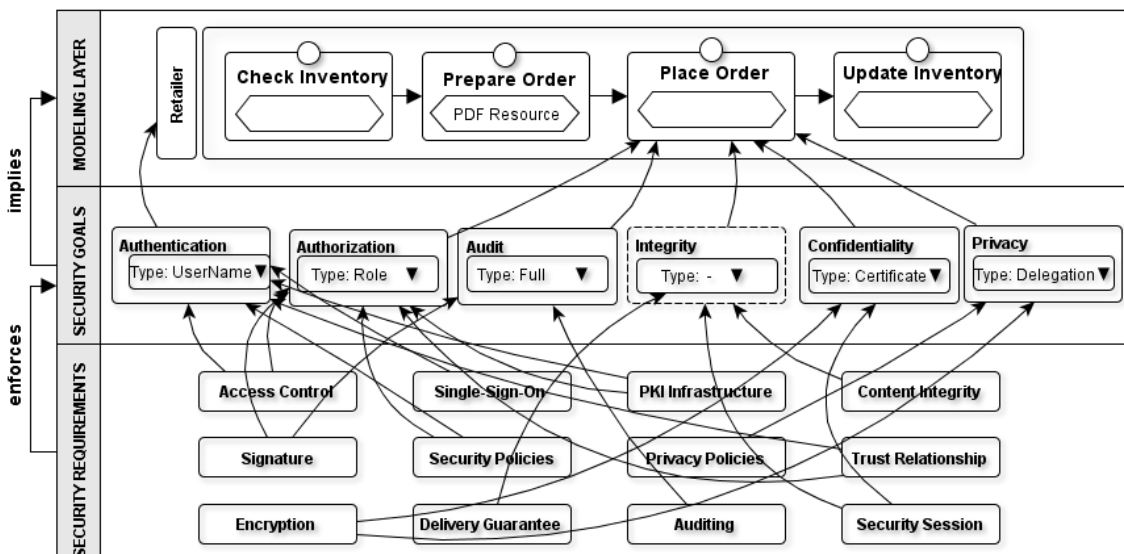


Figure 5.12: Security Requirements for Place Order

6 Design and Implementation

This chapter introduces a prototypical implementation that is meant to be a proof of concept of the previously discussed and elaborated concepts, cf. the workflow model in Chapter 4, the general access control model in Chapter 4.3, and finally the security model presented in Chapter 5.

The chapter begins with a brief overview of the prototype. After the general overview the architecture of the underlying model is presented. This should provide the basis to the reader for understanding the implementation details presented in the next part. The implementation section covers the technologies that we have used and what kind of according security mechanisms were considered to develop the prototype.

6.1 Secure Workflow Designer Prototype

The prototype, named Secure Workflow Designer, is the result of the elaboration with the goal to provide an intuitive graphical modeling environment with integrated security based on the proposed concepts of the previous chapters.

As shown in Figure 6.1, the program user interface consists of following components:

1. **Activities Pane.** The activities pane consists of modeling elements that can be added by a drag-and-drop operation to the workspace. The modeling elements are organized into following categories: agents, operations, resources, and security controls.
2. **Designer Workspace Pane.** The designer workspace pane is an area where to modeling elements can be placed.
3. **Source Code Pane.** The source code pane shows the source code of the designed workflow model.
4. **Properties Pane.** The property pane shows all available properties for the selected modeling element.
5. **Agents Category.** The agents' category consists of an agent lane element presented in Section 4.2.1.
6. **Operations Category.** The operations category consists of operations based on the model presented in Section 4.2.3.
7. **Resources Category.** The resources category consists of resources based on the model presented in Section 4.2.4.
8. **Security Controls Category.** The security controls category consists of modeling elements that represent the security goals presented in Section 5.3.

9. **Basic Category.** The basic category consists of some basic modeling elements that allow defining some of typical flow constructs as described in Section 3.3.2.
10. **Workflow Model.** The model represents the composition of the modeling elements. The composed model is based on the conceptual workflow model presented in Section 4.2.
11. **Variables/Arguments Bar.** This bar allows defining in/out arguments and local variables of the designed model.

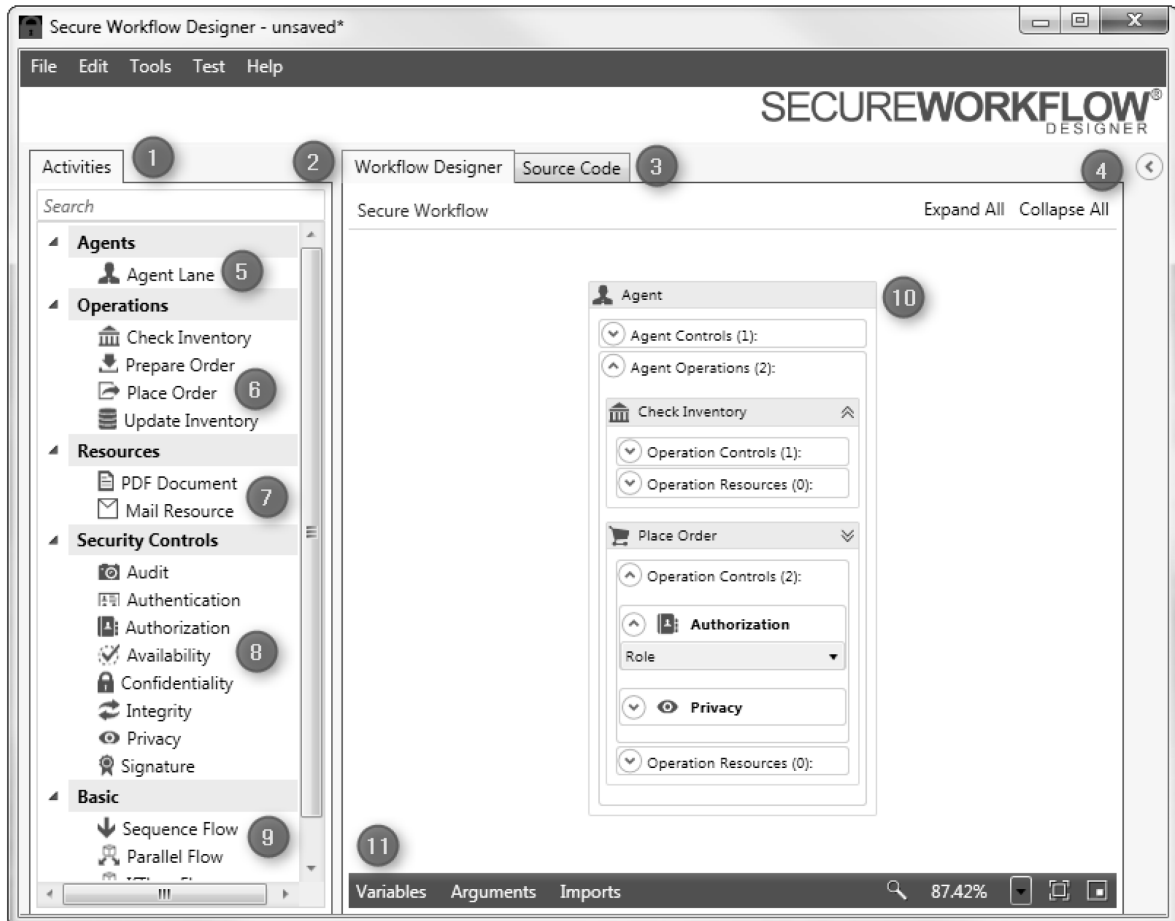


Figure 6.1: Validating Prototype

6.2 Development Environment

The Secure Workflow Designer prototype was built using the following technologies:

- .NET Framework v.4.0 (especially Windows Workflow Foundation and Windows Communication Foundation frameworks)
- SQL Server 2008 Express Edition
- Microsoft Visual Studio 2010

Additionally, following libraries and frameworks has been partly used:

- Windows Identity Foundation SDK v.4.0 (for the claim-based security)
- iTextSharp v.5.0.6 (for the PDF documents related security)
- User Interface Design Framework¹² (graphic icons)

As authors in (Zapletal, van der Aalst, Russell, Liegl, & Werthner, 2009) state, Windows Workflow Foundation (WF) provides a greater expressiveness, in terms of workflow patterns, than BPEL or jBPM. Further, the advantage of being an integral part of the .NET Framework allows WF a smooth and highly customizable incorporation in any type of .NET applications and technologies. For this reason, WF v.4.0 as part of the Microsoft .NET Framework v.4.0.3031 was used as the starting point for the implementation. As described in Section 3.3.3, WF provides a flexible and powerful framework for developing workflow enabled applications. Based on this framework the prototype was developed in Microsoft Visual Studio 2010 used as Integrated Development Environment (IDE), and in C# as the programming language.

The Windows Communication Foundation (WCF) v.4.0 was used for creating, hosting, and consuming of services. WCF is the technology for developing service-oriented applications, and is along WF, Windows Presentation Foundation (WPF), and ADO.NET an integral part of the .NET Framework v.4.0.x. The view of prototype was developed using the WPF framework and its Extensible Application Markup Language (XAML) language for specifying the visual aspects of the prototype. The implementation of PDF related security was done with the iTextSharp v.5.0.6 library. The iTextSharp library provides a comprehensive support for creating and manipulating PDF resources. The library is available under the GNU Affero General Public License v.3 and provides a C# port, which makes it reasonable candidate for our implementation. The database layer of the retailer scenario, presented in Section 5.5, was realized with SQL Server 2008 Express Edition and Entity Framework (EF). Entity Framework as part of ADO.NET v.4.0 is an API for using an object model to communicate with relational databases. Finally, the Windows Identity Foundation (WIF) was used for communicating with the STS of the retailer scenario. WIF is a Software Development Kit (SDK) shipped by Microsoft for claim-based identity in applications.

6.3 General Architecture

As shown in Figure 6.2 the workflow model instance is the central hub for several components of implemented architecture.

¹² This framework is free for every use, even for commercial project, and available under this page: <http://www.webalys.com/design-interface-application-framework.php>.

The Secure Workflow Client component represents the visual and the controller part of the Model-View-Controller (MVC) architectural pattern (Krasner & Pope, 1988). The UI of the application is defined via XML markup in XAML. This markup is used to connect to the control and the message UI handlers of the view. Each part of the UI, as well as the modeled elements (e.g. operations, resources, security controls) in the workspace, has its own representation in XAML. The client controller aggregates the information and implements a publish/subscribe mechanism for events in the workflow model.

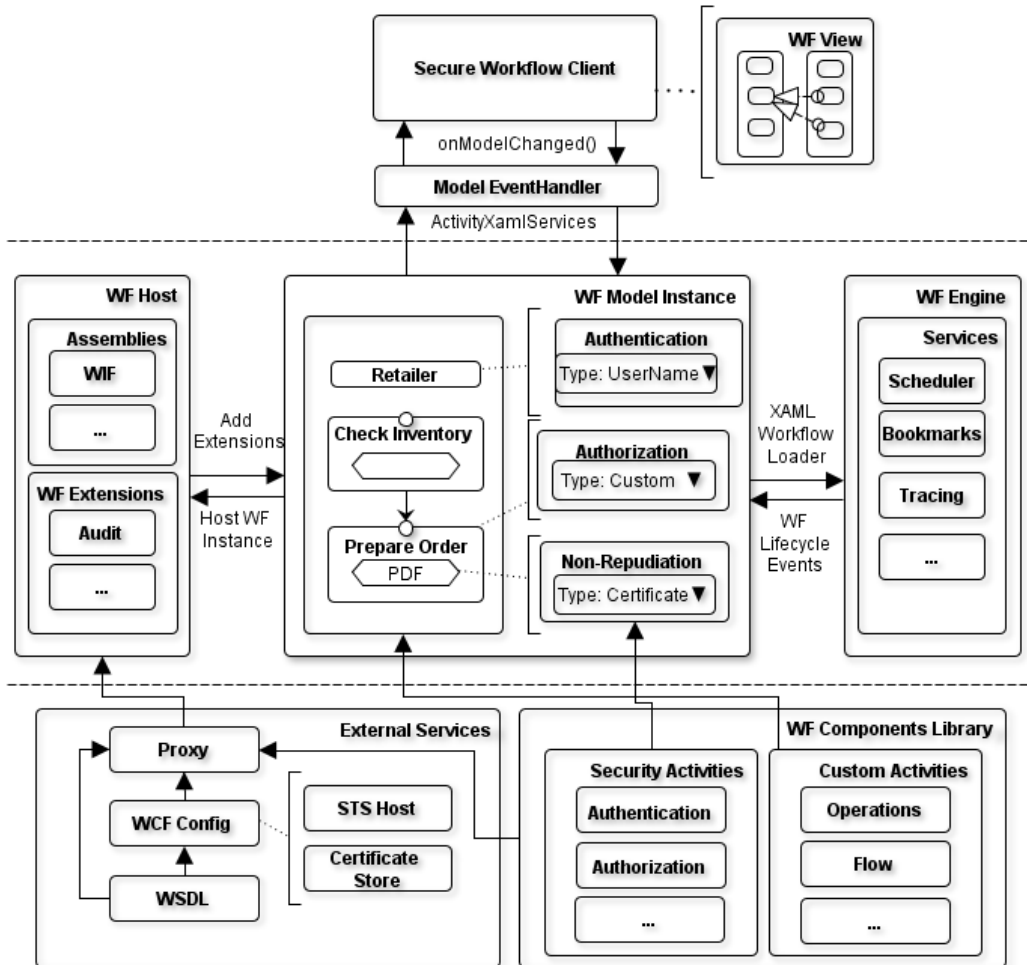


Figure 6.2: General Prototype Architecture¹³

The Workflow Host component hosts the workflow engine and communicates with the client. Further, it implements some infrastructure services, such as the audit functionality, which is based on a special extension mechanism in the workflow engine component. Finally, the Workflow Host is responsible for dynamic loading of missing assemblies such as classes of the WIF or iTextSharp libraries, which are not part of the Client Profile in the .NET Framework.

¹³ For reasons of clarity, the term workflow is abbreviated as WF.

The Workflow Engine is responsible for scheduling and executing the activities of the generated workflow model instance. To deploy a workflow instance, the whole workflow model tree must be previously serialized into XAML code with a special class called *ActivityXamlServices*. This class creates an instance of the model tree in XAML and makes it available for persistence or loading into the workflow engine.

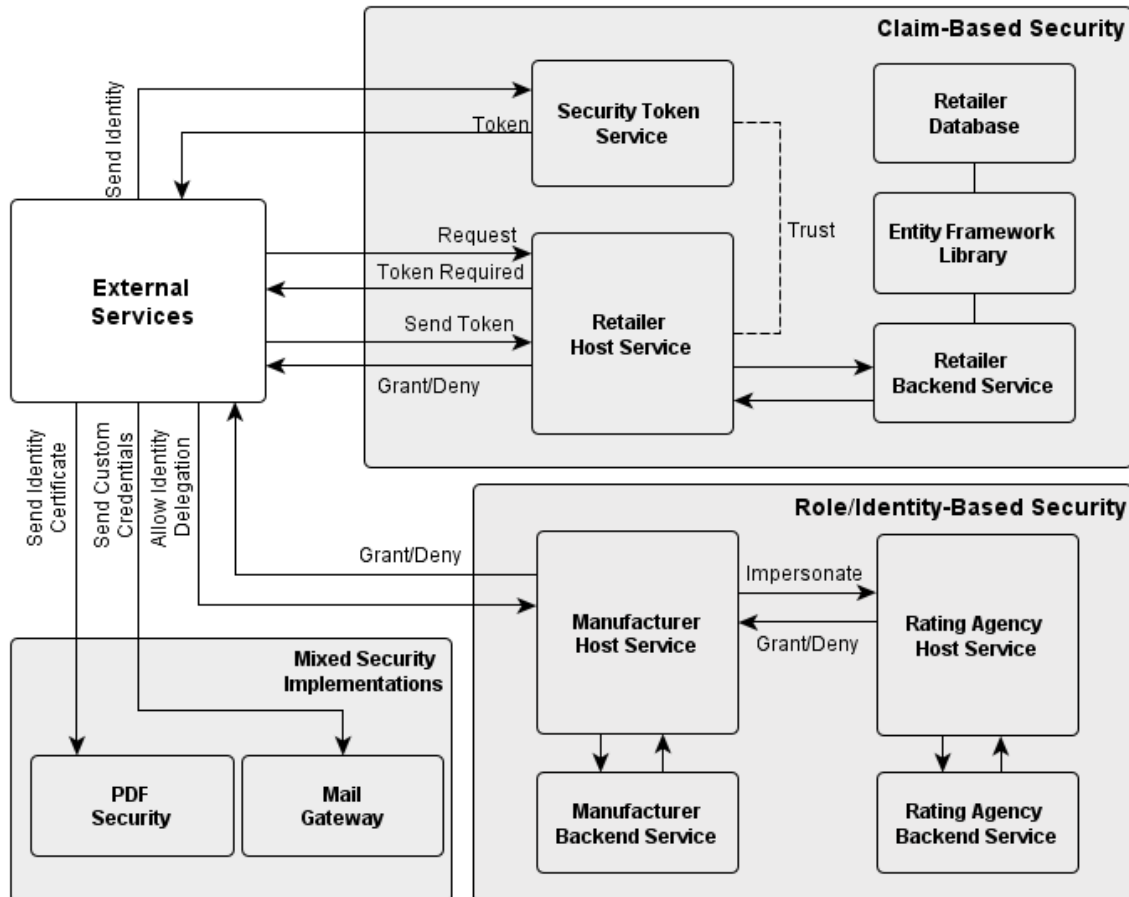


Figure 6.3: External Services

The External Services component is intended for communicating with external web services and mixed security implementations. As shown in Figure 6.3 the communication can be divided into three main categories:

- **Claim-Based Security.** The claim-based security is based on the security token based authentication presented in Section 4.3.1 and claim-based authorization presented in Section 4.3.2.
- **Role/Identity-Based Security.** The role/identity-based security is based on the UserName and security token authentication presented in Section 4.3.1 and role-based authorization presented in Section 4.3.2. As the motivating example involves the indirect communication to the rating agency service, this category impersonation related implementation presented in Section 4.3.3.

- ***Mixed Security Implementation.*** This category consists of legacy implementations or components that cannot be accessed in technology-neutral manner. In our example it is the mail gateway that requires custom credentials and the PDF library for signing and encrypting the PDF resources.

The Workflow Components Library consists of elements that encapsulate the behavior of the security goals presented in Section 5.3. Further, the component defines a custom activities library that provides the logic for all related modeling elements in the workflow designer client, such as agents, operations and resources defined in Section 4.2.

To extend the Secure Workflow Designer architecture (e.g. by implementing new types of security controls or operations), the given logic must be encapsulated in a class that derives from the *Activity* class, as described in Section 6.4.2. To make it available in the UI, the new component must be registered in *getActivitiesToInclude()* and *LoadToolboxItemWrapper()* methods in the *ToolboxUtil* class.

6.4 Prototype Implementation

The Secure Workflow Designer prototype was implemented in several stages. The entire implementation may be summarized as follows:

- Re-hosting of the designer and the workflow engine;
- Implementation of workflow-related logic and workflow modeling elements;
- Implementation of the scenario-based infrastructure (e.g. host and backend services);
- Implementation of the security related infrastructure (e.g. the security token service) and logic (e.g. security enforcing logic of the security controls).

6.4.1 Re-hosting Functionality

To enable the use of the workflow engine and deployment of the implemented modeling elements as part of the workflow tree, a significant effort was made by exploring the *System.Activities.Presentation.Metadata* and *System.Activity.Presentation.Model* namespaces in the .NET Framework. These packages contain many important classes for associating the visual parts with the underlying model implementations. These classes are extensively used for the native modeling in Visual Studio. However, these packages can be also reused in a stand-alone application. This allows implementing a customized designer canvas that can be associated with the underlying workflow runtime.

After implementing the UI of the main application window, the first step was to create an instance to the *WorkflowDesigner* class, and then to reference several UI parts to the designer visual components. In the next step we created an instance of the workflow engine and registered the in-memory representations of the visual shapes. Then, a custom method that loops through the available modeling elements was responsible for adding references to the Activities Pane.

The execution of the visualized model is basically done by flushing and loading the visual model to an instance of the *StringReader* class, then passing the reference to the *ActivityXamlServices*, and finally creating a single instance of the workflow runtime by calling the execution method of the *WorkflowApplication* class.

6.4.2 Workflow-related Functionality

A WF model is organized in custom units of work that can represent a discrete step (e.g. operation or control flow), a resource (e.g. a security goal¹⁴ defined in Section 5.3 or a workflow resource defined in Section 4.2.4), or any further custom implementation as long as its parent class is, or derives from the *Activity* class in the *System.Activities* namespace of the .NET Framework. These custom units of work are called **activities**¹⁵ (Microsoft, 2011) and can be nested in various ways. The resulting hierarchical tree of activities is what we call a workflow model.

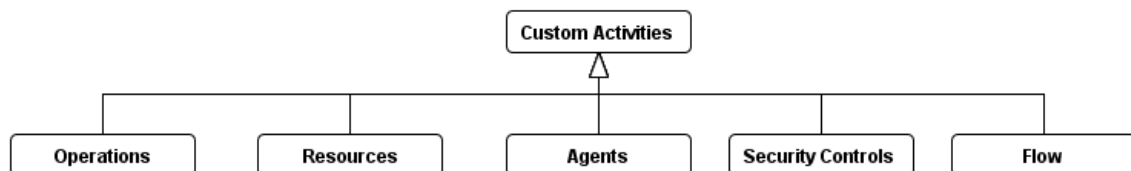


Figure 6.4: Categories of the Custom Categories

As shown in Figure 6.4, our prototype implements several individual activities that can be organized in five categories: Operations, Resources, Agents, Security Controls, and Flow category.

As our main focus is on security related implementation, we shortly present the functionality and implementation of an activity on the example of the *AgentLane*, introduced in Section 4.2.1.

¹⁴ The internal representation of security goals is stored in data variables.

¹⁵ This term is in contrast with what we defined in Section 4.2, where an activity is specified as a combination of an agent, an operation, and a piece of resources. As our prototype builds upon the Workflow Foundation, we use the framework specific terminology in this section.

As indicated in Figure 6.5, the *AgentLane* implementation follows the MVC pattern and consists of three parts:

- **View.** The view consists of two drag-and-drop WPF view controls. These view controls allow the user to add agent security controls and related operations, as shown in Figure 5.3.
- **Controller.** The controller validates the input and passes it to the model.
- **Model.** The model consists of several blocks, each responsible for a particular task in the workflow. The In/Out Arguments are declarations of the *Arguments* class in the *System.Activity* namespace and are responsible for passing data to and out of the activity. The assignment to these arguments can be done in code or in the Variables/Arguments Bar of UI. The next Child Activities block represents the internal storage container for the agent security controls and related operations. Each of these child activities maintains its own state and is scheduled in the Execute Block. The WF runtime makes a clear distinction between built-time and run-time as described in Section 3.3. Prior the actual execution of the activity, the WF creates an instance of the *ActivityMetadata* class in the CacheMetadata block. This metadata instance is used to validate the relationships and dependencies of the activity. In our case, we may detect the type of the available security controls or even add some new implicit security controls as described in Section 5.1.

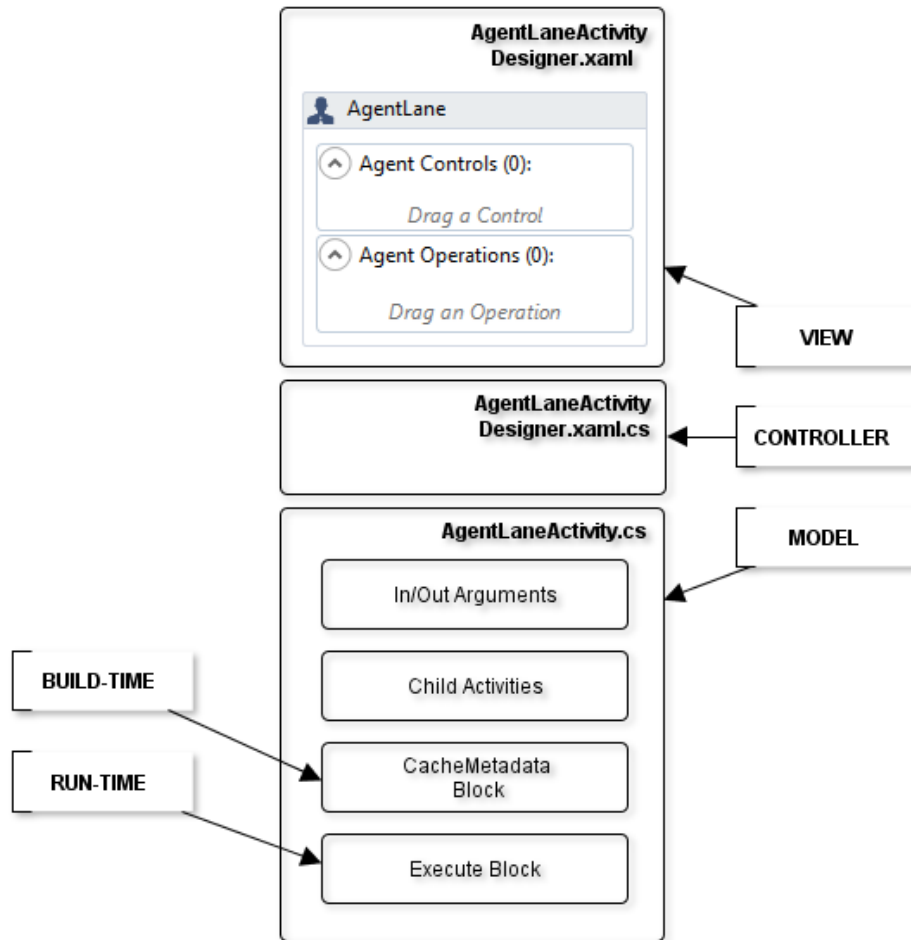


Figure 6.5: Components of the AgentLane Activity

The elaborated design facilitates the implementation of the context-based security. Our realizing algorithm basically consists of two parts: the visual part and the logical part. The visual part is done, by overriding the *CacheMetadata* method. The metadata instance of the *NativeActivityMetadata* class is used to add visual errors and warnings, depending on data in in/out arguments or internal data (e.g. security controls). Then, by using the manual configuring methods of the metadata instance, we schedule the custom activities that are not automatically detected by the reflection mechanism.

The second part implements the run-time related logic by overriding the *Execute()* method of the *Activity* class. Besides the scheduling of the activities for the execution, we may detect the available security controls by creating an instance of the data context of the current workflow environment. The instance contains the manual model property descriptors that we can navigate for the previously added data, e.g. authentication context or a custom signature.

6.4.3 Auditing Security Requirement

WF provides a built-in mechanism for tracking workflow runtime data. The described tracking architecture is based on (Microsoft, 2011) and (Bukovics, 2010).

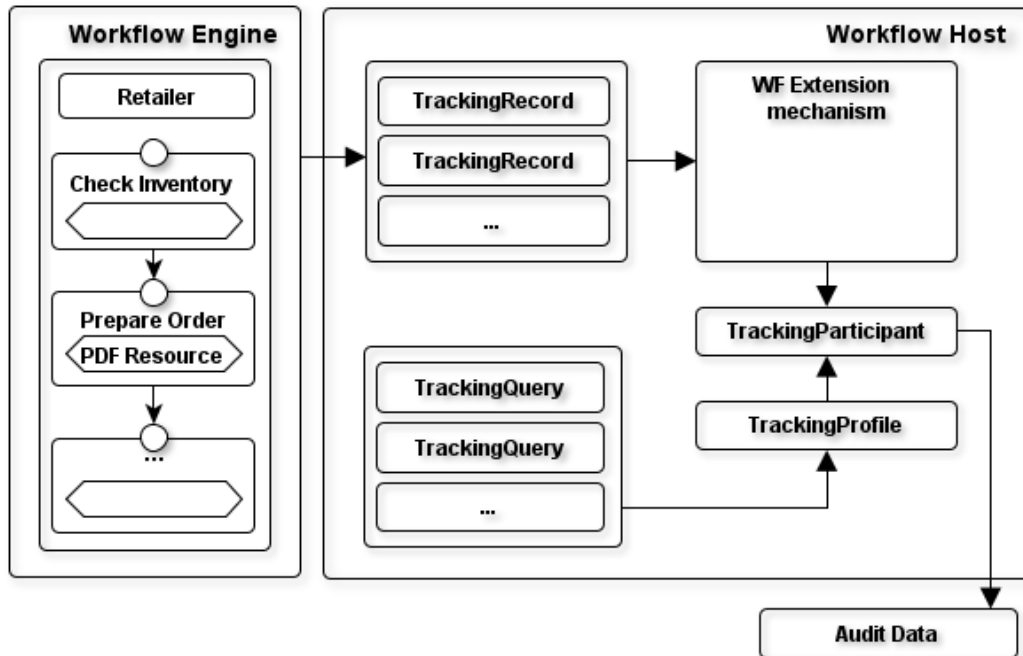


Figure 6.6: Workflow Tracking in WF (Bukovics, 2010)

The tracking mechanism is realized using the publish/subscribe pattern. The published data is organized in tracking records, as shown in Figure 6.6: Workflow Tracking in WF (Bukovics, 2010). Each tracking record is a raw piece of data that can be consumed by a tracking participant. To filter the tracking profiles each tracking participant may implement a tracking profile. A tracking profile consists of tracking query objects that derive from the *TrackingQuery* class and define the target queries against the different types of *TrackingRecord* classes. Our prototype implements the auditing security requirement by adding the previously discussed mechanism via the extension property of the workflow engine.

6.4.4 Infrastructure-related Functionality

Based on the example of claim-based security, as shown previously in Figure 6.3, we describe the implementation of the infrastructure-related functionality. The WCF, as implementing technology, provides a comprehensive API for building secure, reliable, and interoperable services. Each service provides a set of endpoints that represent the sets of resources to which client messages can be sent.

As shown in Figure 6.7, the specification of the communication and the enforcement of the security policies is done via channels and related binding configurations. Each operation (e.g. Check Inventory) has its own configuration set that is filled by the information (e.g. authentication security token) retrieved from the activities, classified in Figure 6.4.

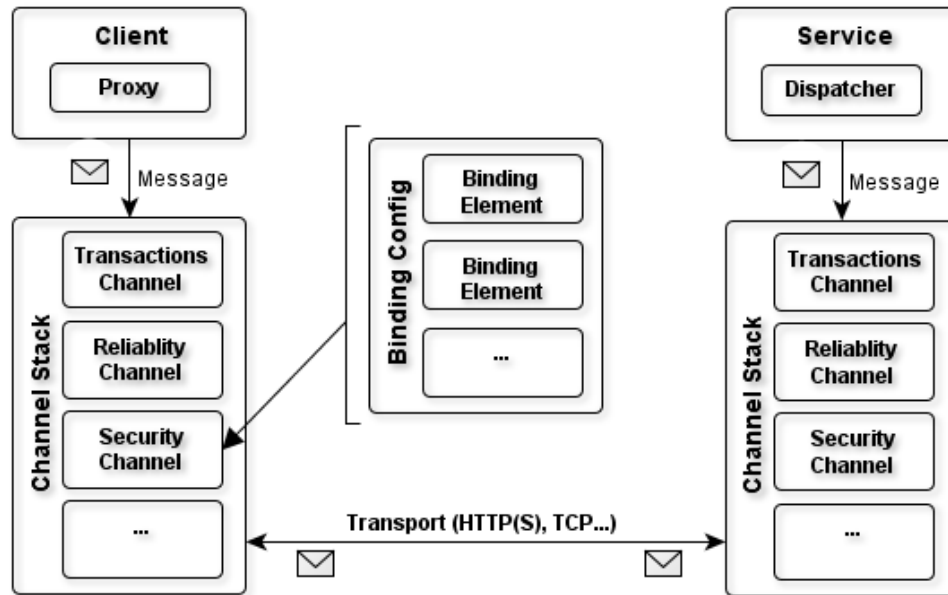


Figure 6.7: Channels and Bindings in WCF (Cibraro, Claeys, Cozzolino, & Grabner, 2010)

The entire approach implementation for the claim-based scenario consists of the following steps:

- Mapping of the relevant parts of the Retailer Database to the backend data model;
- Implementation and configuration (e.g. authorization decisions) of the Retailer backend and frontend services;
- Implementation and configuration of the Retailer Security Token Service;
- Configuration of the Retailer frontend service and the related claim-based operations in the Secure Workflow Designer.

The Retailer data model is realized by using the object relational mapping mechanism shipped with Entity Framework as part of the ADO.NET API. The Figure 6.8 summarizes the data available in the Retailer scenario.

Column Name	Data Type	Database Data Type
ProductID	int	Int32
Name	nvarchar	String
ProductNumber	nvarchar	String
StandardCost	money	Decimal
ListPrice	money	Decimal
SafetyStockLevel	int	Int32
CurrentStockLevel	int	Int32

Figure 6.8: Mappings to the Retailer Database

The generated data model classes are mapped to the physical database by using the connection string added to the *App.Config* file of the Retailer data model project. The Retailer backend service manipulates the generated data model by placing queries in Language Integrated Query (LINQ) against the newly created *InteventoryEntities* class.

The *System.ServiceModel* and *System.Runtime.Serialization* namespaces provide functionality for defining services and their transmission. To identify data and service contract classes that should be exposed to the related WSDL documents, we annotate them with the following WCF-defined attributes: the *[DataContract]* and *[DataMember]* attributes define the data types available to the Retailer service, the *[ServiceContract]* attribute indicates the Retailer service interface and the *[OperationContract]* attributes the related operations.

The implementation of the Security Token Service mainly consists of the *SecurityWorkflowTokenService* and *SecurityWorkflowTokenHandler* classes. The *SecurityWorkflowTokenService* overrides the *GetScope()* and *GetOutputClaims()* methods from the abstract *SecurityTokenService* class defined in WIF. The *GetScope()* takes the requestor's identity as parameter and is responsible for validating and manipulating the incoming token issuance request. Besides the validation, we define the Retailer certificate (stored in the local certificate store) that should be used for encrypting the issued tokens. The *GetOutputClaims()* mainly contains the logic for claims that should be included into the issued token. Our prototype supports besides the predefined rules (e.g. that only managers and employees from the purchasing department are allowed to place purchase requests) a dynamic claims allocation retrieved from the UI. The *SecurityWorkflowTokenHandler* class interferes in the actual token processing by overriding the *ValidateToken()* method. Depending on the provided *UserName* credentials this method authenticates the issued security token and returns a set of authorizing policies in form of a *ClaimsIdentityCollection*.

The configuration of the Retailer frontend service is done, by defining the binding and behavior configuration in the *App.config* file. As our implementation is mainly based on open security standards, we widely reuse the binding elements of the *WS2007HttpBinding* and *WS2007FederationBinding* classes. These classes provide support for several security mechanisms discussed in Section 3.2.3. The most important sections concern the message security and the type of the accepted claims. The Retailer frontend service defines a claim-based authentication contract that in turn represents a combination of the “*http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name*” and “*http://SecureWorkflow/2011/02/Role*” claim types. To be authenticated by the Retailer frontend each requesting token must include the defined claim types. Further, the requesting token must be issued by a trusted STS (Retailer STS). The whole scenario for the claim-based security is depicted in Figure 6.9:

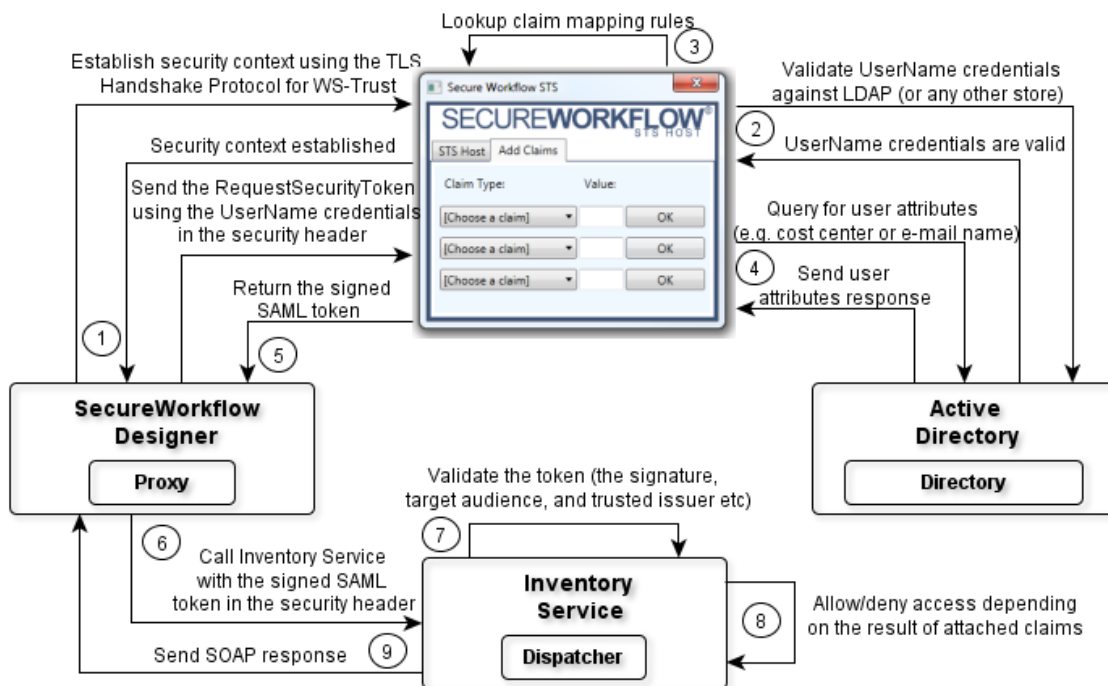


Figure 6.9: The Claim-Based Security Scenario

1. To establish the authenticity and a shared security context between SecurityWorkflowDesigner and Inventory STS, the client sends a *RequestSecurityToken (RST)* message using the *TLS Handshake Protocol for WS-Trust* (Alexander, et al., 2007). This protocol defines a request response pattern for security token acquisition (for a secured communication channel) and defines exchanges for negotiation and challenges. After establishing the security context the proxy of the SecurityWorkflowDesigner sends a *RequestSecurityToken* using the *UserName* credentials (retrieved in our case from the *AgentLane*).

2. The STS Service receives the credentials and validates them against an active directory or any other custom store.
3. After passing the validation process, the STS looks up the associated claims for the provided identity. The WIF significantly alleviates this step by providing a code representation of various well-known claim type schemas (Microsoft, 2011).
4. If the claim mapping rules require additional information (e.g. the actual email address of the authenticated identity as value for the "*http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress*" claim) the STS may query for missing user attributes the Active Directory or any other custom store.
5. After completing the claim mapping process, the STS sends a *RequestSecurityTokenResponse* (RSTR) message. This message contains an attached SAML token that includes a set of issued claims by the STS.
6. Once the SecureWorkflowDesigner receives the SAML security token, it calls the Inventory Service by including the obtained token to WS-Security header of the requesting message.
7. On each request the Inventory Service decrypts the SAML information (e.g. by using the public key of the issuing STS, if the SAML token was encrypted by the private key) and verifies that it can trust the included set of claims (e.g. by extracting the signature, expiration date, target audience etc.).
8. Depending on the extracted claims the Inventory Service can make authorization decisions (e.g. if the provided role claim is not in the authorized group, the service can deny the request).
9. Depending on the authorization decision the Inventory sends the requested information or a SOAP fault.

7 Summary and Future Work

The main goal of this thesis was to provide a technology-independent framework for security visualization and enforcement in business process driven environments. The work was motivated by the observation that domain experts often need a profound knowledge of security relevant aspects to be able to model real-life scenarios.

In the first part we presented a motivating use case typically found in supply chains. Based on this example we deduced the technical implications, which in turn significantly shaped the problem statement covered in this thesis.

As access control is one of the most fundamental mechanisms in any security solution, we began the second part with a detailed analysis of appropriate access control models and related constraints. Furthermore, we identified and reviewed previous works on modeling security in workflows. In the subsequent chapter we have continued our work by dealing with main concepts and technologies in workflow and security. Our analysis included the in-depth identification and discussion of competing workflow models, security goals, related security requirements, and implementing security mechanisms.

Based on these results we began to develop our framework by identifying the business modeling constructs for a general workflow model. The elaborated workflow model was used as a basis for the security framework introduced in the subsequent chapter. The transition to the security framework was fulfilled by investigating in a general access control model that would facilitate the enforcement of authentication, authorization, impersonation in workflow related systems. To reflect the variety of business scenarios and related security mechanisms the presented security framework included several levels of abstraction. Each level was provided with detailed transitions matrices, allowing encompassing several workflow-related security mechanisms from the convenient visual modeling up to the actual enforcement and implementation.

After elaborating the several layers of security, we discussed how the motivating example, introduced in the first part, can be applied to the unified security framework. Finally, the thesis was rounded by a synopsis of the architecture, design, and implementation details of the validating prototype.

Since we mainly focused on security relevant aspects, future work can be continued in the following adjacent areas:

Establishing of Trust. As (Bishop, 2009) points out, the proof of authenticity in distributed systems is often realized with a help of a trusted authority. In general, the trusted authority

is responsible for recognizing the requesting party as the claimed one. Thus, the establishing of a trustful communication between two or more parties highly depends on the trustworthiness of the issuing party. The recent news about compromising one of the root certificate authorities (Bright, 2011) circumstantiates the fundamental vulnerability of such systems. Based on some common trust perspectives reviewed in (Saunders, Wu, Li, & Weisfeld, 2004) it can be assumed that despite recent advances in technological infrastructure business environments will still have to rely on initial human interaction. We may assume that the establishing of trust would be easier in business process driven environments if it could be modeled as a part of the business process.

Business rules and constraints. According to the Business Rules Group (The Business Rules Group, 2011) a business rule “*is a statement that defines or constraints some aspect of the business. It is intended to assert business structure, or to control or influence the behavior of the business.*” As many rule-based approaches in recent literature exist, we could imagine a unified view that combines the realized business rules engines (e.g. first-order logic) with the security relevant aspects elaborated in context of this thesis.

Compliance. Depending on the point of view it’s coming from, the security requirements may be seen as part of regulatory compliance laws and regulations. The Directive 95/46/EC Directive (The European Parliament, 1995), also known as the EU Data Protection Directive and the Sarbanes–Oxley Act (SOX) (Sarbanes-Oxley Act, 2002), especially the Section 404, that covers the topic of internal controls, can be seen as the most prominent examples in that area. As companies strive or as most cases are even obliged to fulfill an increasing number of standards and regulations, a unified view, especially on visualization of compliance control objectives and its enforcement, would be a significant step to decrease the burden of modeling business processes in compliance with applicable laws and regulations.

Bibliography

- Aagesen, G., & Krogstie, J. (2010). Analysis and design of business processes using BPMN. *Handbook on Business Process Management 1* , pp. 213--235.
- Agrawal, A., Amend, M., Das, M., Ford, M., Keller, C., Kloppmann, M., et al. (2007, 6). *WS-BPEL Extension for People (BPEL4People), Version 1.0*. Retrieved from <https://www.sdn.sap.com/irj/sdn/go/portal/prtroot/docs/library/uuid/30c6f5b5-ef02-2a10-c8b5-cc1147f4d58c>
- Alexander, J., Della-Libera, G., Gajjala, V., Gavrylyuk, K., Kaler, C., McIntosh, M., et al. (2007, 09 07). *Application Note: Using WS-Trust for TLS Handshake*. Retrieved from <http://schemas.xmlsoap.org/ws/2005/02/trust/tls/WSTrustForTLS.pdf>
- Alves, A., Arkin, A., Askary, S., Barreto, C., Bloch, B., Curbera, F., et al. (2007, 4 11). *OASIS. Web Services Business Process Execution Language Version 2.0*. Retrieved from <http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf>
- Apache Software Foundation. (2011). *Apache Rampart - Axis2 Security Module*. Retrieved from <http://axis.apache.org/axis2/java/rampart/index.html>
- Artelsmair, C., Essmayr, W., Lang, P. W., & Weippl, E. (2002). CoSMo: An Approach Towards Conceptual Security Modeling. (Springer-Verlag, Ed.) *LECTURE NOTES IN COMPUTER SCIENCE* , pp. 557-566.
- Baier, D., Bertocci, V., Brown, K., Pace, E., & Woloski, M. (2010). *A Guide to Claims-Based Identity and Access Control*. Microsoft Press.
- Bartel, M., Boyer, J., Fox, B., LaMaccia, B., & Simon, E. (2008, 06 10). *W3C XML Signature Syntax and Processing* . Retrieved from <http://www.w3.org/TR/xmlsig-core/>
- Bayer, T. (2008, 07). *Example of authorization with WS-Trust and SAML*. Retrieved from <http://www.predic8.com/ws-trust-example.htm>
- Bertino, E., Ferrari, E., & Atluri, V. (1999, 2). The Specification and Enforcement of Authorization Constraints in Workflow Management Systems. *ACM Transactions on Information and System Security* , 2, pp. 65-104.
- Bertino, E., Martino, L., Paci, F., & Squicciarini, A. (2009). *Security for Web Services and Service-Oriented Architectures*. Springer.

- Bertocci, V. (2011). *Programming Windows Identity Foundation*. Microsoft Press.
- Bishop, M. (2004). *Introduction to Computer Security*. Prentice Hall PTR.
- Bishop, J. (2009). *Security in Computing Systems. Challenges, Approaches and Solutions*. Springer.
- Brewer, D. (2000). *Risk Assessment Models and Evolving Approaches*. Retrieved from <http://www.gammasl.co.uk/topics/IAAC.htm>
- Bright, P. (2011). *Ars Technica. How the Comodo certificate fraud calls CA trust into question*. Retrieved from <http://arstechnica.com/security/news/2011/03/how-the-comodo-certificate-fraud-calls-ca-trust-into-question.ars>
- Buckley, S., Hardjono, T., Yu, T., Hudson, G., R., S., & Tsitkova, Z. (n.d.). *Kerberos: The Network Authentication Protocol*. Retrieved from <http://web.mit.edu/kerberos/>
- Bukovics, B. (2010). *Pro WF. Windows Workflow In .NET 4*. Apress.
- Bussard, L., Neven, G., & Preiss, F.-S. (2010). Downstream Usage Control. *Policies for Distributed Systems and Networks (POLICY), 2010 IEEE International Symposium on*.
- Christensen, E., Curbera, F., Meredith, G., & Weerawarana, S. (2001, 3 15). *W3C. Web Services Description Language (WSDL) 1.1*. Retrieved from <http://www.w3.org/TR/wsdl>
- Cibraro, P., Claeys, K., Cozzolino, F., & Grabner, J. (2010). *Professional WCF 4: Windows Communication Foundation with .NET 4*. John Wiley & Sons.
- Ciocoiu, C., & Dobrea, R. (2010). The Role of Standardization in Improvement the Effectiveness of Integrated Risk Management. In *Advances in Risk Management*. Sciyo Publishing.
- Cope, E. W., Kuster, J. M., Etzweiler, D., Deleris, L. A., & Ray, B. (2010, 6). Incorporating risk into business process models. *IBM Journal of Research and Development* , pp. 4:1 - 4:13.
- Debervoise, T. (2005). *Business Process Management with a Business Rules Approach. Implementing the Service Oriented Architecture*. Arbor Books.

- Emig, C., Brandt, F., Abeck, S., Biermann, J., & Klarl, H. (2007, 8). An Access Control Metamodel for Web Service-Oriented Architecture. *International Conference on Software Engineering Advances (ICSEA 2007)* , pp. 57-57.
- Feng, X., Jun, X., Hao, H., & Li, X. (2004). Context-Aware Role-Based Access Control Model for Web Services. (S. Verlag, Ed.) *LECTURE NOTES IN COMPUTER SCIENCE* , 3252, pp. 430-436.
- Ferraiolo, D., Kuhn, R., & Chandramouli, R. (2007). *Role-Based Access Control*. Artech House.
- Forster, A., Engels, G., Schattkowsky, T., & Straeten, R. V. (2007). Verification of business process quality constraints based on visual process patterns. *Theoretical Aspects of Software Engineering* , pp. 197-208.
- Fremantle, P., Patil, S., Davis, D., Karmarkar, A., Pilz, G., Winkler, S., et al. (2007, 6 14). *OASIS. Web Services Reliable Message (WS-Reliable Messaging) Version 1.1*. Retrieved from <http://docs.oasis-open.org/ws-rx/wsrn/200702/wsrn-1.1-spec-os-01.pdf>
- Friedman, T. L. (2007). *The World Is Flat: A Brief History of the Twenty-First Century*. Douglas & McIntyre.
- Gudgin, M., Hadley, M., Mendelson, N., Moreau, J.-J., Nielsen, H., Karmarkar, A., et al. (2007, 4 27). *W3C. SOAP Version 1.2 Part 1: Messaging Framework*. Retrieved from <http://www.w3.org/TR/soap12-part1/>
- Hafner, M., & Breu, R. (2009). *Security Engineering for Service-Oriented Architectures*. Springer-Verlag.
- Hallam-Baker, P., & H. Mysore, S. (2005, 6 28). *W3C XML Key Management Specification (XKMS 2.0)*. Retrieved from <http://www.w3.org/TR/xkms2/>
- Harmon, P. (2003). *Business process change: a manager's guide to improving, redesigning, and automating processes*. Morgan Kaufmann.
- Harvey, M. (2005). *Essential business process modeling*. O'Reilly Media.
- Hernan, S., Lambert, S., Ostwald, T., & Shostack, A. (2006, 11). Uncover Security Design Flaws Using The STRIDE Approach. *MSDN Magazine* .
- Herrmann, P., & Herrmann, G. (2006). Security requirement analysis of business processes. *ELECTRONIC COMMERCE RESEARCH* , 6 (3-4), pp. 305-335.

- Hewett, T., Baecker, R., Card, S., Carey, T., Gasen, J., Mantei, M., et al. (1992). Curricula for Human-Computer Interaction. *Report of the ACM SIGHI Curriculum Development Group*.
- Imamura, T., Dillaway, B., & Simon, E. (2002, 12 10). *W3C XML Encryption Syntax and Processing*. Retrieved from <http://www.w3.org/TR/xmlenc-core/>
- Indulska, M., Recker, J., Rosemann, M., & Green, P. (2009). Business process modeling: Current issues and future challenges. *Advanced Information Systems Engineering* , pp. 501--514.
- Iwasa, K., Durand, J., Rutt, T., Peel, M., Kunisetty, S., et al. (2004, 11 15). *OASIS. Web Services Reliable Messaging TC. WS-Reliability 1.1*. Retrieved from http://docs.oasis-open.org/wsrn/ws-reliability/v1.1/wsrn-ws_reliability-1.1-spec-os.pdf
- Jurjens, J. (2002). UMLsec: Extending UML for Secure Systems Development. *LECTURE NOTES IN COMPUTER SCIENCE* , pp. 412-425.
- Krasner, G., & Pope, S. (1988). A cookbook for using the model-view controller user interface paradigm in Smalltalk-80. *Journal of Object-oriented programming* , S. 26-49.
- Lawrence, K., Kaler, C., Nadalin, A., Goodner, M., Gudgin, M., Barbir, A., et al. (2009, 2 2). *OASIS. WS-Trust 1.4 Specification*. Retrieved from <http://docs.oasis-open.org/ws-sx/ws-trust/v1.4/ws-trust.html>
- Leymann, F. (2010). BPEL vs. BPMN 2.0: Should You Care? *Second International Workshop, BPMN 2010* (pp. 8-13). Potsdam: Springer.
- Leymann, F., & Roller, D. (2000). *Production workflow: Concepts and Techniques*. Prentice Hall PTR.
- Li, J., Li, X., Xie, S., Chen, C., Liu, G., & Pan, Y. (2008, 12). Multi-hierarchy and fine-grained task-role-based access control in collaborative environments. *Industrial Engineering and Engineering Management* , p. 1591.
- Li, W., & Fan, Y. (2009). A Time Management Method in Workflow Management System. *Grid and Pervasive Computing Conference, 2009. GPC '09.*, (p. 3). Geneva.
- Lu, R., & Sadiq, S. (2007). A survey of comparative business process modeling approaches. *Business Information Systems* , pp. 82-94.

- Luc, C., & al., e. (2010). *OASIS. WS-BPEL Extension for People (BPEL4People) Specification Version 1.1*. Retrieved from <http://docs.oasis-open.org/bpel4people/B4P-CD-07-CD-09-Diffs.pdf>
- Meier, J., Farre, C., Taylor, J., Bansode, P., Gregersen, S., Sundararajan, M., et al. (2008). *Improving Web Services Security. Patterns and Practices*. Microsoft Press.
- Meier, J., Mackman, A., Dunner, M., Vasireddy, S., Escamilla, R., & Murukan, A. (2003). *Improving Web Application Security: Threats and Countermeasures*. Retrieved from <http://msdn.microsoft.com/en-us/library/ff648644.aspx>
- Melzer, I. (2004). *Service-orientierte Architekturen mit Web Services: Konzepte- Standards- Praxis*. Spektrum Verlage.
- Microsoft. (2011). *MSDN. Windows Identity Foundation. ClaimTypes Members*. Retrieved from http://msdn.microsoft.com/en-us/library/microsoft.identitymodel.claims.claimtypes_members.aspx
- Microsoft. (2011). *MSDN. Windows Workflow Foundation Glossary for .NET Framework 4*. Retrieved from <http://msdn.microsoft.com/en-us/library/dd489447.aspx>
- Microsoft. (2011). *MSDN. Windows Workflow Overview*. Retrieved from <http://msdn.microsoft.com/en-us/library/dd489465.aspx>
- Microsoft. (2011). *MSDN. Workflow Tracking and Tracing*. Retrieved from <http://msdn.microsoft.com/en-us/library/ee513992.aspx>
- Mili, H., Jaoude, G., Lefebvre, T. G., & Petrenko, A. (2003, 11). Business process modeling languages: Sorting through the alphabet soup. *TR LATECE* .
- Moses, T., Anderson, A., Nadalin, A., Parducci, B., Engovatov, D., Flinn, D., et al. (2005, 2 1). *OASIS. eXtensible Access Control Markup Language (XACML) Version 2.0*. Retrieved from http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf
- Muehlen, M. z. (2004). *Workflow-based Process Controlling. Foundation, Design, and Application of workflow-driven Process Information Systems*. Berlin: Logos.
- Muehlen, M., & Recker, J. (2008). How much language is enough? Theoretical and practical use of the business process modeling notation. (Springer, Ed.) *Advanced Information Systems Engineering* , pp. 465-479.

- Nadalin, A., Goodner, M., Gudgin, M., Barbir, A., & Granqvist, H. (2009, 2 2). *OASIS WS-SecureConversation 1.4*. Retrieved from <http://docs.oasis-open.org/ws-sx/ws-secureconversation/v1.4/os/ws-secureconversation-1.4-spec-os.doc>
- Nadalin, A., Kaler, C., Monzillo, R., & Hallam-Baker, P. (2006, 02 01). *OASIS. Web Services Security UsernameToken Profile 1.1*. Retrieved from <http://www.oasis-open.org/committees/download.php/16782/wss-v1.1-spec-os-UsernameTokenProfile.pdf>
- Nadalin, A., Kaler, C., Monzillo, R., & Hallam-Baker, P. (2006, 2 1). *OASIS. Web Services Security: SOAP Message Security 1.1*. Retrieved from <http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf>
- Nadalin, A., Kaler, C., Monzillo, R., & Hallam-Baker, P. (2006, 2 1). *OASIS. X.509 Certificate Token Profile 1.1*. Retrieved from <http://www.oasis-open.org/committees/download.php/16785/wss-v1.1-spec-os-x509TokenProfile.pdf>
- Newcomer, E., & Lomow, G. (2004). *Understanding SOA with Web Services*. Addison Wesley Professional.
- Nordbotten, N. (2009, 08 21). XML and Web Services Security Standards. *IEEE Communications Surveys & Tutorials* , pp. 4-21.
- Nysetvold, A., & Krogstie, J. (2006). Assessing Business Processing Modeling Languages Using a Generic Quality Framework. *Advanced Topics in Database Research. Volume 5 of Advanced Topics in Database Research Series* , pp. 79-93.
- Oh, S., & Park, S. (1999). Task-Role Based Access Control (T-RBAC): An Improved Access Control Model for Enterprise Environment. (Springer-Verlag, Ed.) *LECTURE NOTES IN COMPUTER SCIENCE* , pp. 264-273.
- Paci, F., Bertino, E., & Crampton, J. (2008). An Access-Control Framework for WS-BPEL. *International Journal of Web Services Research* , 5.
- Papazoglou, M. P. (2008). *Web services: Principles and Technology*. Pearson Education.
- Pfleeger, C. P. (2006). *Security in Computing*. Prentice Hall.

- Ragouzis, N., Hughes, J., Philpott, R., Maier, E., Madsen, P., & Scavo, T. (2008, 03 25). *OASIS. Security Assertion Markup Language (SAML) V2.0 Technical Overview*. Retrieved from <http://www.oasis-open.org/committees/download.php/27819/sstc-saml-tech-overview-2.0-cd-02.pdf>
- Recker, J. (2010). Opportunities and constraints: the current struggle with BPMN. (E. G. Limited, Ed.) *Business Process Management Journal* , 16, pp. 181-201.
- Resnick, S., R., C., & Bowen, C. (2008). *Essential Windows Communication Foundation. For .NET Framework 3.5*. Pearson Education.
- Riesner, M., & Pernul, G. (2010). Supporting Compliance through Enhancing Internal Control Systems by Conceptual Business Security Modeling. *ACIS 2010 Proceedings* .
- Rodriguez, A., Fernandez-Medina, E., & Piattini, M. (2007). A BPMN Extension for the Modeling of Security Requirements in Business Processes. *IEICE TRANSACTIONS ON INFORMATION AND SYSTEMS E SERIES D* , 90, pp. 745-752.
- Röhrig, S. (2003). *Using Process Models to Analyse IT Security Requirements*.
- Rosenberg, J., & Remy, D. (2004). *Securing Web services with WS-Security: demystifying WS-Security, WS-Policy, SAML, XML Signature, and XML Encryption*. SAMS.
- Russell, N., Ter Hofstede, A., Edmond, D., & van der Aalst, W. (2004). Workflow Data Patterns. *QUT Technical report, FIT-TR-2004-01*. Queensland University of Technology, Brisbane.
- Russell, N., Ter Hofstede, A., Edmond, D., & van der Aalst, W. (2004). Workflow Resource Patterns. *BETA Working Paper Series, WP 127*. Eindhoven University of Technology, Eindhoven.
- Russell, N., Ter Hofstede, A., van der Aalst, W., & Mulyar, N. (2006). Workflow Control-Flow Patterns: A Revised View.
- Sandhu, R. S., Coyne, E. J., Feinstein, H. L., & Youman, C. E. (1996). Role-Based Access Control Models. *COMPUTER* , 29 (2), pp. 38-48.
- Sarbanes-Oxley Act. (2002, 07 30). *U.S. Government. Public Law 107 - 204*. Retrieved from <http://www.gpo.gov/fdsys/pkg/PLAW-107publ204/content-detail.html>

- Saunders, C., Wu, Y., Li, Y., & Weisfeld, S. (2004). Interorganizational trust in B2B relationships. *Proceedings of the 6th international conference on Electronic commerce* (pp. 272--279). ACM.
- Schaad, A. (2003). *A Framework for Organisaitonal Control, PhD Thesis, in Department of Computer Science*. University of York.
- Schaad, A., & Moffett, J. (2002). A framework for organisational control principles. *Computer Security Applications Conference, 2002. Proceedings. 18th Annual*, (pp. 229 - 238).
- Schumm, D., Leymann, F., & Streule, A. (2010). Process Viewing Patterns. *Proceedings of the 14th IEEE International EDOC Conference* (pp. 89--98). Vitória, Brazil: IEEE Computer Society.
- Shostack, A. (2007). *MSDN. The Security Development Lifecycle. STRIDE chart*. Retrieved from <http://blogs.msdn.com/b/sdl/archive/2007/09/11/stride-chart.aspx>
- Sleeper, B., & Robins, B. (2002). *The Laws of Evolution: A Pragmatic Analysis of the Emerging Web Services Market. The Stencil Group*. Retrieved from [http://www.sysedv.tu-berlin.de/intranet/kc-kb.nsf/8f6cd592abb29d6bc1256979005defbc/40EB248462528FF2C1256DB400498DEC/\\$File/Laws_of_Evolution.pdf?OpenElement](http://www.sysedv.tu-berlin.de/intranet/kc-kb.nsf/8f6cd592abb29d6bc1256979005defbc/40EB248462528FF2C1256DB400498DEC/$File/Laws_of_Evolution.pdf?OpenElement)
- The Business Rules Group. (2011). Defining Business Rules... What is a Business Rule?
- The European Parliament. (1995, 11 23). *Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data*.
- van der Aalst, W. M., & Pesic, M. (2006). DecSerFlow: Towards a Truly Declarative Service Flow Language. (Springer, Ed.) *LECTURE NOTES IN COMPUTER SCIENCE* , pp. 1-23.
- van Der Aalst, W., Ter Hofstede, A., Kiepuszewski, B., & Barros, A. (2003). Workflow patterns. *Distributed and parallel databases* , pp. 5-51.
- Vedamuthu, A., Orchard, D., Hirsch, F., Hondo, M., Yendluri, P., & Boubez, T. (2007, 9 4). *W3C. Web Services Policy 1.5 - Framework*. Retrieved from <http://www.w3.org/TR/ws-policy/>

- Vijayalakshmi, A., & Warner, J. (2008). Security for Workflow Systems. In S. Jajodia, *Handbook of database security: applications and trends*. Springer.
- Völzer, H. (2010). An Overview of BPMN 2.0 and Its Potential Use. *Second International Workshop, BPMN 2010* (pp. 14-15). Potsdam: Springer.
- Weerawarana, S., Curbera, F., Leymann, F., Storey, T., & Ferguson, D. (2005). *Web Services Platform Architecture: SOAP, WSDL, WS-Policy, WS-Addressing, WS-BPEL, WS-Reliable Messaging, and More*. Prentice Hall PTR.
- Weske, M. (2007). *Business Process Management: Concepts, Languages, Architectures*. Springer.
- White, A., Anthony, M., Arkin, A., Astier, S., Bartel, R., Bakmeyer, E., et al. (2009, 1). *OMG. Business Process Model and Notation (BPMN) 1.2*. Retrieved from <http://www.omg.org/spec/BPMN/1.2/>
- Wöhrle, F. (2008). *Verschlüsselung und Signaturen in der kollaborativen Entwicklung von WS-BPEL Prozessen*. University of Stuttgart, Institute of Architecture of Application Systems.
- Wolter, C., Menzel, M., Schaad, A., Miseldine, P., & Meinel, C. (2009, 4). Model-driven business process security requirement specification. *Journal of Systems Architecture* , 55, pp. 211-223.
- Workflow Management Coalition. (1999). *Workflow Management Coalition Terminology & Glossary*. Retrieved from http://www.wfmc.org/standards/docs/TC-1011_term_glossary_v3.pdf
- Yuan, E., & Tong, J. (2005, 7). Attributed based access control (ABAC) for Web services. *Web Services, 2005. ICWS 2005. Proceedings*.
- Zapletal, M., van der Aalst, W., Russell, N., Liegl, P., & Werthner, H. (2009). An Analysis of Windows Workflow's Control-Flow Expressiveness. *Web Services, 2009. ECOWS'09. Seventh IEEE European Conference on* , pp. 200-209.
- zur Muehlen, M., & M., R. (2005). Integrating Risks in Business Process Models. *16th Australasian Conference on Information Systems*. Sydney.

All links were last followed on 2011-05-10

Declaration

All the work contained within this thesis, except where otherwise acknowledged, was solely the effort of the author. At no stage was any collaboration entered into with any other party.

(Thomas Karsten)