



UNIVERSITÄT
KOBLENZ · LANDAU

Institut für Wirtschafts-
und Verwaltungsinformatik



FB 4

Informatik

SOA-Security

Rüdiger Grimm

Farid Mehr

Anastasia Meletiadou

Daniel Pähler

Ilka Uerz

Nr. 19/2007

**Arbeitsberichte aus dem
Fachbereich Informatik**

Die Arbeitsberichte aus dem Fachbereich Informatik dienen der Darstellung vorläufiger Ergebnisse, die in der Regel noch für spätere Veröffentlichungen überarbeitet werden. Die Autoren sind deshalb für kritische Hinweise dankbar. Alle Rechte vorbehalten, insbesondere die der Übersetzung, des Nachdruckes, des Vortrags, der Entnahme von Abbildungen und Tabellen – auch bei nur auszugsweiser Verwertung.

The “Arbeitsberichte aus dem Fachbereich Informatik“ comprise preliminary results which will usually be revised for subsequent publication. Critical comments are appreciated by the authors. All rights reserved. No part of this report may be reproduced by any means or translated.

Arbeitsberichte des Fachbereichs Informatik

ISSN (Print): 1864-0346

ISSN (Online): 1864-0850

Herausgeber / Edited by:

Der Dekan:

Prof. Dr. Paulus

Die Professoren des Fachbereichs:

Prof. Dr. Bátori, Jun.-Prof. Dr. Beckert, Prof. Dr. Burkhardt, Prof. Dr. Diller, Prof. Dr. Ebert, Prof. Dr. Furbach, Prof. Dr. Grimm, Prof. Dr. Hampe, Prof. Dr. Harbusch, Jun.-Prof. Dr. Hass, Prof. Dr. Krause, Prof. Dr. Lautenbach, Prof. Dr. Müller, Prof. Dr. Oppermann, Prof. Dr. Paulus, Prof. Dr. Priese, Prof. Dr. Rosendahl, Prof. Dr. Schubert, Prof. Dr. Staab, Prof. Dr. Steigner, Prof. Dr. Troitzsch, Prof. Dr. von Kortzfleisch, Prof. Dr. Walsh, Prof. Dr. Wimmer, Prof. Dr. Zöbel

Kontaktdaten der Verfasser

Rüdiger Grimm, Farid Mehr, Anastasia Meletiadou,

Daniel Pähler, Ilka Uerz

Institut für Wirtschafts- und Verwaltungsinformatik

Fachbereich Informatik

Universität Koblenz-Landau

Universitätsstraße 1

D-56070 Koblenz

E-Mail: grimm@uni-koblenz.de, farid.mehr@hs-furtwangen.de, nancy@uni-koblenz.de, tulkas@uni-koblenz.de, chnee@uni-koblenz.de

TABLES OF CONTENTS

1 INTRODUCTION	1
2 OUR SECURITY CONCEPT	1
2.1 Methodology of security analysis	1
2.2 Open versus closed cooperation systems	2
2.3 Security as application enabler	3
2.4 Enforcement vs. comprehensibility ("Nachvollziehbarkeit")	4
2.5 IT risk management as extension of IT security	5
2.6 A new "Web service view" on security requirements	6
2.6.1 Confidentiality	6
2.6.2 Privacy	7
2.6.3 Accountability	8
2.6.4 Controllability	8
2.6.5 Dependability	9
3 THE SECURITY CHALLENGE OF SOA	9
3.1 Openness	10
3.2 Dynamic configuration and integration of components and composite services ..	10
3.3 The integration of external services	10
4 WEB SERVICE SECURITY	11
4.1 Overview of WS security	11
4.1.1 XML Signature	12
4.1.2 XML Encryption	13
4.1.3 P3P	14
4.1.4 WS-Security: SOAP Message Security	14
4.1.5 WS-Policy	15
4.1.6 WS-Trust	15
4.1.7 WS-SecureConversation	16
4.1.8 WS-Federation	16
4.1.9 WS-Authorization	17
4.1.10 WS-Privacy	17
4.1.11 WS-Reliability and WS-ReliableMessaging	17

4.2 Impact of WS-Security and related technologies on SOA	17
4.2.1 Confidentiality	17
4.2.2 Privacy	18
4.2.3 Accountability.....	18
4.2.4 Controllability.....	19
4.2.5 Dependability.....	19
4.3 Remaining challenges	20
4.3.1 WS-Security alone not sufficient.....	20
4.3.2 Compatibility of specifications.....	20
4.3.3 Integration of external legacy systems.....	20
4.3.4 Privacy	21
5 BEST PRACTICE EXAMPLES	22
5.1 The ASG middleware platform (JBossWS).....	22
5.1.1 Confidentiality	23
5.1.2 Integrity and Authenticity	23
5.1.3 Summary	24
5.2 Overview of existing SOA system providers and their customers	25
5.3 A survey on SOA security practice.....	28
6 CONCLUSION.....	30
REFERENCES.....	31
PROJECT CONSORTIUM INFORMATION.....	34

LIST OF FIGURES

Figure 1: Basic methodology of IT security analysis	2
Figure 2: Enforcing and proving security mechanisms	5
Figure 3: Overview of the WS-Security framework.....	11
Figure 4: A Reference Element [14].....	12
Figure 5: A SignedInfo Element [14].	13
Figure 6: The Web Services Trust Model [22].	16

LIST OF TABLES

Table 1: Comparison of transport layer and message-level security [14].	15
Table 2: Implementation status of WS-Security related specification	25

ABSTRACT

This paper is a part of the ASG project (Adaptive Services Grid) and addresses some IT security issues of service oriented architectures. It defines a service-oriented security concept, it explores the SOA security challenge, it describes the existing WS-Security standard, and it undertakes a first step into a survey on best practice examples. In particular, the ASG middleware platform technology (JBossWS) is analyzed with respect to its ability to handle security functions

1 INTRODUCTION

The goal of Adaptive Services Grid (ASG) project is to develop an architectural blueprint and a proof-of-concept prototype of an open development platform for adaptive and reliable matchmaking discovery, composition, and enactment. ASG provides the integration of its sub-projects in the context of an open platform, including tool development by small and medium sized enterprises. Based on semantic specifications of requested services by service customers, ASG discovers appropriate services, creates composed services and enacts them.

The acceptance of a service-oriented architecture for ASG depends not only on a richer functionality and a better manageability, but also on the solutions of the security problems which arise from the specific architecture of openly interacting services.

This work sub-package in the context of the ASG project addresses the security problem of SOA within ASG. In section 2 a service-oriented structure of IT security requirements is developed as a basis for the subsequent security analysis. This structure is used in section 3 to identify IT security problems which are specific to an open service architecture like ASG. Section 4 describes the existing WS-Security suite, how it can solve the specific SOA threats and which security gaps remain open. In section 5 existing SOA security solutions are surveyed, especially the ASG middleware platform technology (JBossWS). Finally other SOA providers and users are asked about their experiences with IT security in SOA.

In summary, this deliverable is not a concrete implementation guideline, but can rather be seen as a structured framework for further research and development in order to improve the ASG technology. In particular, relating to the “ASG Service Provisioning Features”, security problems are named which arise with the integration of external services as well as on-demand service composition.

2 OUR SECURITY CONCEPT

2.1 Methodology of security analysis

There is no commonly agreed terminology of security, safety, and dependability of information systems, neither among experts, nor among users, see, for example [1],[2],[3],[4]. However, they all agree on the basic understanding that information security is concerned with the “protection” of “assets” against “threats”, which can be

specified by security “requirements”. The earliest IT security criteria catalogue (“Orange Book”, [1]) addressed only the security requirement “confidentiality”. But shortly after, IT security was extended to the basic security requirements confidentiality, integrity, and availability (“CIA”). By further development of computers and networks, in particular by the introduction of open networks via the Internet and the World Wide Web, we learnt that security requirements make up an open list which is to be adapted to the specific application context. Information systems which operate energy plants or traffic systems have other security requirements than information systems used by banks. This is reflected by the current international standard on IT security criteria, the so-called Common Criteria [4]. In addition to the classical CIA-triad of requirements some applications may rely on other security requirements like privacy, non-repudiation, or accountability, to mention only a few.

In the context of this project, we follow the methodology of security analysis, which starts with the assets of the real world. Assets are under the threat of manipulation by conflicting parties. According to the application context, the authorized users agree on a set of security requirements, which are to be supported by implemented security mechanisms. Security mechanism may be technical (such as encryption), organisational (such as separation of duty, or redundant components), or even legal (such as responsibility of CEOs by law).

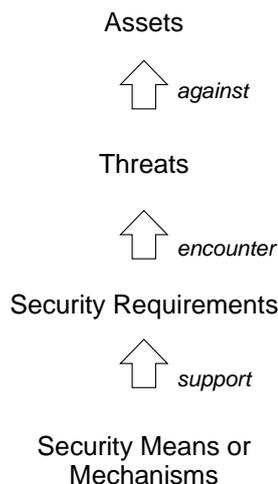


Figure 1: Basic methodology of IT security analysis

2.2 Open versus closed cooperation systems

The significant difference between open and closed cooperation systems is that closed systems are (or at least can be) under central control, while open systems are not. In open systems new users and applications are free to enter, and existing users and applications are free to leave on their choice. In open systems there is not only no central control, but also there is no central knowledge about the transactions. In general, open systems are insecure. Network architects would call this a feature of the Internet,

not a bug. End-to-end security mechanisms across insecure communication media provide an appropriate means to overcome this problem.

In open systems, cooperation partners must be regarded as autonomous persons who cannot be forced by technical means to behave according to the agreed cooperation rules. Partners who wish to do transactions in open systems require mechanisms to prove their trustworthiness. Binding policy statements, contracts, and continuous accountability are examples for such mechanisms.

2.3 Security as application enabler

The classical view of IT security focuses on the defence against attacks and errors. In this view, security meets non-functional requirements, because it protects functions which exist already within the system. However, an unbiased analysis of security problems must not be restricted to the conflict between good guys (authorized users) and bad guys (attackers), but it should take into account any conflict of interests between different roles of usage. For example, in a service-oriented architecture binding contracts such as purchase and payment can only be concluded if the system provides functions to protect buyers and sellers against one another. Any contractor must be enabled to trust his partners. Trust can and should be supported by security functions such as encryption of sensitive communication and non-repudiation (signatures) of binding contracts. This way, security functions which help to balance conflicts of interests enable new applications.

As to the role of security as a balance between conflicting interests, the classical view that threats are directed from unauthorized “bad guys” against “good users” does indeed cover an important subset of security requirements. Access control, encryption, and virus detection are typical mechanisms to keep “bad guys” away. But there is another important subset of security requirements to protect conflicting cooperation between (authorized!) partners. Electronic commerce is an application context which requires mechanisms, e.g., for fair exchange of money for goods, for non-repudiation of contracts, or for transparent usage of private data. These security requirements do not address bad guys, but they allow good guys to perform binding cooperation across open networks which belong to nobody and which are not under central control. Therefore, this is a top challenge for the security in service-oriented architectures (SOA).

As to the role of security as application enabler, it is important to understand that security requirements are not only “non-functional properties” to protect application functions, which are there with or without security mechanisms, but which may fail their aims if attacked by bad guys. While the “non-functional” role of IT security remains an important subset of security requirements, there is another important subset of security requirements which address a “functional role” of IT security, namely the application enabling security functions. Digital signatures, fair exchange protocols, and privacy enhancing technology support this application enabling role of IT security. This is particularly important in open networks of autonomous partners and thus poses another top challenge to the security in SOA.

The starting point of IT security analysis is given by the real assets in information systems. Attackers look for ways to gain access to assets in a way the authorized users do not want. In general, the access of unauthorized attackers to application assets is at the cost of the authorized users. A successful attack would provoke an unfair (or even illegal) flow of values from authorized users to the attacker.

In the extended view of IT security as a protection of open cooperation between conflicting partners, the aim is a fair exchange of values of the application assets. For example, a contract binds both parties to fulfil their promises. Especially in e-commerce, a seller is bound to deliver the good to the user, after he has received the payment. Vice versa, the buyer is bound to deliver the money after he has received the purchased good. A bilaterally signed contract may support the non-repudiation of such a mutual promise. If one of the parties misbehaves, the contract can be shown to trusted third-parties which are able to enforce legal behaviour (courts, referees). In most cases contracts are not enforced by third parties, but they are kept by well-behaving parties. However, without the possibility of contracts, open commercial cooperation wouldn't work. In this sense, digital signatures protect binding e-commerce and enables e-commerce across open networks.

Therefore, a general security analysis of service-oriented architectures does not only address unauthorized attackers, but also the conflicting interests within the authorized user community. In accompany of the real assets in information systems, a general security analysis of open networks must specify the roles of the partners and their interest. The possible conflicts are to be identified. Service providers and service users are a typical pair of cooperation partners with different interests. But also different subsets of service users may have different interests, such as, for example, power sellers, private sellers, business purchasers and private buyers within eBay.

2.4 Enforcement vs. comprehensibility ("Nachvollziehbarkeit")

We need to differentiate two types of security mechanisms in open systems. Type one are the security enforcing mechanisms. They enforce with technical means that nothing can go wrong. Strong encryption is an example for confidentiality enforcement. Security mechanisms of type number two do not enforce any requirements, but they prove the compliance with or the break of requirements. For example, digital signatures provide a proof of integrity. Due to the security analysis of an application context in open systems, the security designer constructs a mixed architecture of technical and organisational mechanisms which enforce or prove the compliance with the application rules by the cooperation partners.

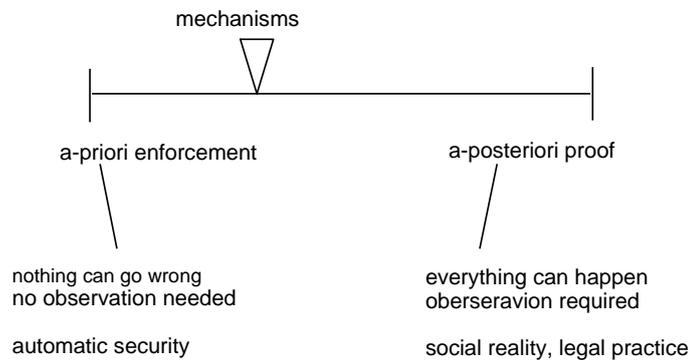


Figure 2: Enforcing and proving security mechanisms

2.5 IT risk management as extension of IT security

IT security addresses the protection of IT systems against unauthorized, unfair or illegal usage. It is specified by security requirements which feed the needs of their users. Confidentiality, integrity, authenticity, availability, non-repudiation, privacy, accountability are some typical examples of IT security requirements.

The dependability of users on their information systems lies beyond the borders of IT security, but requires attention, too. Avizienis et al. [3] define dependability by the requirements reliability, safety, and maintainability, which add to the classical “CIA”-triad. In some communities, especially in Germany, it has become common to emphasize the specific role of safety in contrast to security. While the security community defines security as the protection against purposeful attacks and safety as protection against errors, the safety community defines safety as the protection of the environment against bad effects of the information systems, and security as the protection of the internal procedures of the information systems.

Dierstein [5] has integrated these views into a new structure of terminology, differentiating between dependability (“Verlässlichkeit”) and controllability (“Beherrschbarkeit”). The first describes what Dierstein calls the technical view: it is based on the aforementioned CIA requirements, demanding that the system itself work according to its specification (and therefore, be dependable). The latter, on the other hand, provides the complementary users’ view on IT security, in which CIA plays a subordinate role. It demands that the user be protected of negative effects which even a dependable system can cause: open networks, and particularly open service architectures increase the complexity of offers and effects. This is security relevant in the sense that users can lose assets in that they use the services in an inappropriate way or that they are overrun by unforeseen effects. To encounter these problems, controllability subsumes the notions of accountability and legal liability: it must be possible to assign the responsibility for the system’s effects to someone and to prove

this responsibility without doubt. While the fulfillment of these requirements can increase the user's trust in the system, it also poses another challenge to SOA security. An even broader extension to security is the approach of IT risk management (as part of a general risk management) of enterprises. It adds to the requirements of dependability and controllability the contingency planning, which includes incident response, disaster recovery and business continuity planning (e.g., [6]). Contingency planning is out of scope of this project.

2.6 A new "Web service view" on security requirements

Different application contexts require different security protection, according to the assets, persons, their interest, and the typical threat scenario given by the technological and organisational basis of the application. Services within an open network, realized by Web technology, are such an application context with specific security requirements. Therefore, instead of structuring our security analysis by the basic triangle of confidentiality, integrity and availability only, we group some basic IT security requirements in five service-relevant higher-level categories. These five categories are confidentiality, privacy, accountability, dependability, and controllability. They can be broken down to lower-level security requirements, the usage of which can be evaluated directly in a security analysis of services.

2.6.1 Confidentiality

Confidentiality is the property of communication that unauthorized partners are excluded from listening ("eaves-dropping"). Confidentiality of data refers to transfer and storage of the data. Stronger than confidentiality is unlinkability, which hides the link between any kind of data and nodes in the network. Unlinkable communication hides the fact that there is any communication. Therefore, it does not only hide the content of communication, but also who is talking to whom at what time and in what extent.

In open environments like the Internet it is not possible in general to exclude unauthorized listeners from observation of communication. However, they should not be able to understand the content of the data transferred through the network. In this case, encryption is an appropriate means to ensure confidentiality. Encryption can be enforced between end-users and is effective even in extremely open and insecure networks like the Internet.

However, unlinkability requires stronger means than encryption alone. In this case, anonymizing systems must be used such as David Chaum's MIX networks, which are realized by remailers [9] and by the Anonymity.Online Service of the Universities of Regensburg and Dresden (2000-2006) [10].

For all kinds of applications confidentiality is most important. It is one of the columns of trust between persons who need to communicate sensible issues. This includes the contexts of e-government, e-commerce, and private applications. In many applications such as private usage, confidentiality is even required by law (privacy). E-business wouldn't work without confidentiality, not only for protection against competitors, but also as a means to govern the public relations of a company.

2.6.2 Privacy

The Directive 95/46/EC on privacy of the EU [8] has been implemented in all European countries. It also affected the USA through the Safe Harbor contract [7]. It is based on a set of principles which can partly be supported by technical means in order to enhance privacy of Internet applications. These are the principles of privacy:

- Visibility ("transparency") of how services use personal data of their users
- Policy control by data protection law enforcement bodies
- Strong binding between personal data and purpose of their usage
- Informed consent by users' choice
- Data protection by system design and by user control

"System design and user control" are based on these principles:

- Thriftiness or even avoidance of personal data by system design
- Anonymous and "persona" access to a service
- Support of user control functions over personal user data

Personal data would be "avoided", for example, by anonymous access to a service. A service would still be "thrifty" with personal data, if it offers "persona" access, i.e. user pseudonyms like "Mickey Mouse" which, on misuse, may be associated with a real user by a trusted third party. Another example for thriftiness is an immediate billing after usage by direct electronic payment in contrast to long term storage of billing data.

A Web service in an open environment would support user control over personal user data in that it offers this group of "user control functions":

1. Service identification labels (a flag with contact data)
2. Online information on privacy practices of a service
3. Electronic consent (and its withdrawal) by users on exploitation of their personal data
4. Electronic inspection, deletion and correction of their data by users

A strong instrument for the transparency of privacy policies in open systems is the P3P project of the World-Wide Web Consortium [11]. It allows services to offer their privacy policies to users, and it allows users to keep their privacy preferences in their client tools such as browsers. When clients and services meet, the P3P preference tool would warn the user if the privacy policy of the service does not match the user's preferences. Apart from this existent use of P3P, it remains to be determined in how far P3P could be used in semantic Web services: it might not be needed within closed systems or in B2B communication between systems, but in scenarios where personal data is used by external services, privacy policies are a vital means to make sure that the data is only treated in the way the user intended. In the future, semantic descriptions could be a way to assert the transitivity of policies and automatically solve conflicts between Web services with incompatible policies.

2.6.3 Accountability

Accountability is the property of a service that every action of the service provider is clearly (and legally) mapped to a responsible person. Accountability is often linked with liability. Accountability may also be applied to user actions. This is certainly the case in B2B applications. In B2C applications, however, it is good business practice to treat customers fairly if they produce acceptable errors.

Accountability depends on the transparency and comprehensibility of what is going on in a service. Accountability is technically supported by proofs of action, which use non-repudiation functions such as digital signatures. They prove the integrity of data, and the non-repudiation of data origin. Finally, logging transactions is a good basis for proving actions.

A service cannot be accountable if the persons responsible cannot rely on the service functions. On a service level, not only single actions must be reliable, but whole transactions must be secured.

In summary, accountability relies on the following requirements:

- Responsibility
- Liability
- Transparency
- Authenticity
- Non-repudiation of origin
- Reliability of functions
- Integrity of data
- Securing Transactions

2.6.4 Controllability

According to Dierstein's taxonomy on IT security and safety [5], controllability refers to the protection of users from the system. This is of course related to dependability, i.e. to the technical functioning of the system. However, beyond the dependability of the system, users must be able to use the system in a way that feeds their needs. For example, a system must be clearly designed and it must be "usable" on the basis of good ergonomics and on the basis of a good economy of resources.

Especially with open services, simple and safe single services may have negative effects when they are used together. For example, a music download shop may work nicely with a payment system of the shop provider, but not with the payment system of the customer. So, the download shop should be able to offer more than only one payment system. The diversity of service components may, however, cause negative effects, for example unforeseen charges.

Another problem is the complexity of compound services. Will the combination of a translation service and a help function provide a help function in the new language or

just cause confusion? Or what are the effects of the composition of two translation services?

In summary, controllability relies on the following requirements:

- Management of complexity (e.g. coupling of several services to a compound service)
- Management of diversity (e.g. many payment systems, alternative media)
- Usability
- Ergonomics
- Economy of resources

2.6.5 Dependability

Dependability is the most technical view on IT security. A system is called dependable if its users can safely depend on the system. That is, the system is going to perform exactly the way the designers expect. This is called “functional reliability”: the system acts in the way as it is specified. The classical IT security experts started with dependability by the well-known “CIA” structure, which stands for confidentiality, integrity and availability. Integrity is more than proof-of-authenticity, because it includes the conformance to the expected functionality. But it is also less than proof-of-authenticity, because a proof of authenticity includes a proof-of-origin, which we deal with in the category of accountability. Integrity, strictly speaking, is only the property of data of not having been tampered with. However, it is a good practice to combine dependability and accountability to a reliable and secure IT system.

In summary, dependability relies on the following requirements:

- Functional Reliability
- Confidentiality
- Integrity
- Availability

3 THE SECURITY CHALLENGE OF SOA

SOA stands for specific flexibility of components which offer services via standardized interfaces in an open environment such as the Internet. In particular, services can enter and leave the market place without permission of a central service (openness), they can be called dynamically, and several of them may be integrated into a compound service. For the time being though, SOA is primarily used within closed in-house systems. This can be accounted for by the fact that they are easier to control from a security point of view, whereas many of the security problems that come with open systems have not yet been solved satisfactorily. In closed systems however, the integration of external services rises security problems that have to be taken into account, too.

3.1 Openness

Open systems are not under central control. In open systems, cooperation partners must be regarded as autonomous persons who cannot be forced by technical means to behave according to the agreed cooperation rules. There are three aspects of openness. Firstly, customers may enter the market place to use the SOA service the first time. Services must be able to serve these customers appropriately. Secondly, providers may enter the market place to offer new services. Users must be able to trust these services, that is, these services must be reliable and accountable to responsible persons. Thirdly, customers may want to use different services in different combinations. They must be protected against negative effects of this complexity and diversity.

3.2 Dynamic configuration and integration of components and composite services

A specific feature of SOA is the possibility of dynamic configuration and integration of components. A user may come across a new service (for example, a music download) and might wish to combine it with the specific payment system which he is used to. The combination of these two SOA services to one reliable transaction is a challenge both for technology and responsibility of persons. A very first step may be the transparent declaration of business policies. However, what is more challenging is the enforcement of transitivity of responsibilities and functions. For example, the further transfer of personal data is clearly regulated by privacy laws. This is not equally clear with the transitivity of service responsibility. Who is liable if combined payment fails? And the other way round: who takes care of new security holes that arise from the combination of two services which are secure as long as they are used alone?

3.3 The integration of external services

Within a closed user group, service providers follow common rules on interface standards, policies and transparency. When anything goes wrong, there are responsible persons to deal with the problem. Security policies can be implemented and enforced by central control.

This is not the case when external services are allowed to be used in combination with internal services. Beyond standardization and transparency, which must be translated to and by the external service, security rules are difficult to export and import.

One approach to security in open systems consists of encryption and digital signatures between end-users in an insecure network. Public-key infrastructures can handle cross-certificates of public keys in order to extend proofs-of-authenticity. Therefore, confidentiality can be enforced beyond closed systems. Furthermore, accountability can be extended by common legal rules on the basis of digital signatures.

The transparency of privacy policies by P3P works across different service domains, too. However, the enforcement of privacy policies and the enforcement of dependability, as well as the controllability of services cannot easily be extended to external services. In a first step, bilateral agreements between any new external service and the closed service group must be concluded. The automation of security enforcement into open systems is a research issue.

4 WEB SERVICE SECURITY

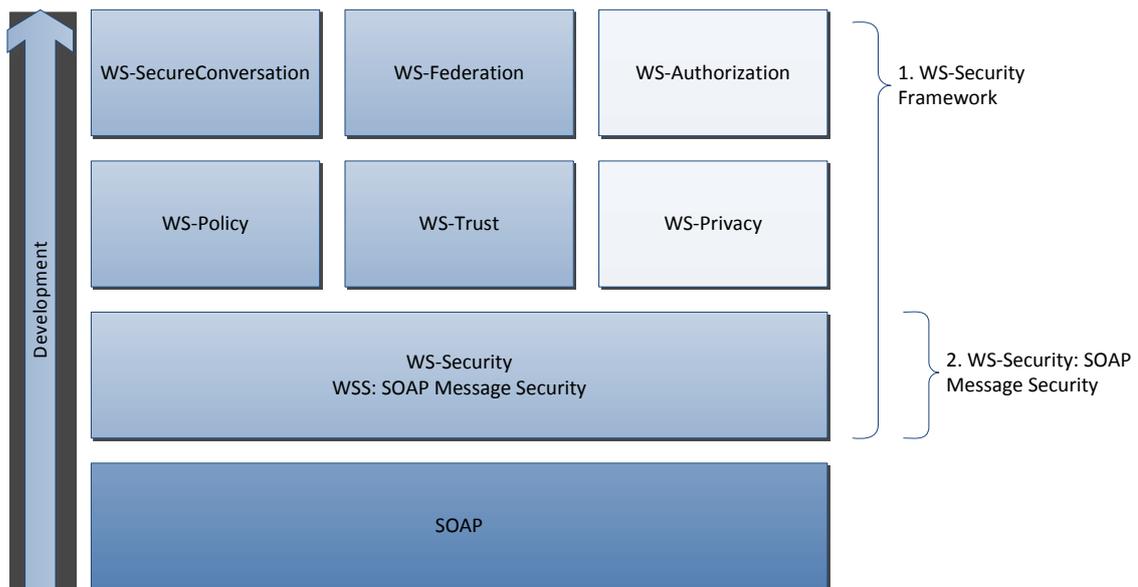
After security requirements and challenges have been shown in the last sections, in this section we will take a closer look at the current technical possibilities, how they can be used to approach these challenges and where they are still insufficient.

4.1 Overview of WS security

As the name indicates, the main goal of WS-Security (or Web Service Security) is to secure Web services, for instance to address the threats discussed in section 3.

The term “WS-Security” is used for two related concepts:

1. A whole framework of interrelated specifications for Web Service Security (**WS-Security Framework**, also known as “the WS-* specifications”).
2. The first of these specifications with the full name “Web Service Security: SOAP Message Security” [1], which is the foundation for all other specifications in the same framework (see **WS-Security: SOAP Message Security**)



Legend:

- WS-Security Standard under Development
- Published WS-Security Standard
- Other Standard Used as Foundation

Figure 3: Overview of the WS-Security framework

WS-Security: SOAP Message Security was originally developed by IBM, Microsoft, and VeriSign, but since a few years ago, it is edited by OASIS Open. The current version, WS-Security 1.1, was published on 1 February 2006 [12].

Before we can discuss the WS-Security standards we have to consider related W3C standards: XML Signature, XML Encryption and P3P.

4.1.1 XML Signature

XML Signature [13] is a W3C Recommendation for digitally signing an XML document, a part of an XML document, or an external object.

If we look at the structure of the XML source code, there are three forms of signatures:

- Enveloping Signature – The signature wraps the object which is signed.
- Enveloped Signature – The signature is contained in the object which is signed.
- Detached Signature – The signature and the signed object are separate elements, either in the same XML document or with the signature in the XML document referencing an external resource.

The process of creating an XML signature consists of two steps:

Reference Generation – the items that are being signed are processed to calculate a digest value:

1. Apply the transformations (such as XPath statements) to determine which part of the XML document will be signed.
2. Calculate the digest value over the resulting data objects.
3. Create a Reference element including the calculated DigestValue and information about how it was created (the referenced resource, the used digest method). An example for the result is shown in Figure 4.

```
<Reference URI="http://www.foo.com/securePage.html">
  <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
  <DigestValue>60NvZvtdTB+7UnlLp/H24p7h4bs=</DigestValue>
</Reference>
```

Figure 4: A Reference Element [14].

Then the **Signature Generation** creates the Signature itself:

Create the SignedInfo element including the Reference element created before plus info about the CanonicalizationMethod and the SignatureMethod.

```

<SignedInfo>
  <CanonicalizationMethod
    Algorithm="http://www.w3.org/TR/2000/WD-xml-c14n-20001011" />
  <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-
    sha1" />
  <Reference URI="http://www.foo.com/securePage.html">
    <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"
    />
    <DigestValue>60NvZvtdTB+7UnlLp/H24p7h4bs=</DigestValue>
  </Reference>
</SignedInfo>

```

Figure 5: A SignedInfo Element [14].

Canonicalize the SignedInfo using the specified CanonicalizationMethod. This ensures that documents with identical content but different formatting and encoding will be treated identically.

From the canonicalized SignedInfo calculate a hash using the method specified in DigestMethod.

From that hash value calculate the SignatureValue by using the algorithm specified in SignatureMethod.

Construct the Signature element including SignedInfo, SignatureValue, Object, and KeyInfo.

The process to validate an XML Signature is similar, except that it occurs in reverse. By calculating and comparing hash values the Reference Validation ensures that the referenced objects have not been changed. The Signature Validation ensures that the SignedInfo block has not been changed and the correct key has been used for signing.

4.1.2 XML Encryption

XML Encryption [15] provides the encryption of an XML document or a part of an XML document.

In XML Encryption the main element is EncryptedData that contains the information that is encrypted.

With XML Encryption it is possible to encrypt different parts of an XML document so, that they can be seen by different recipients.

To increase the efficiency of the encryption, the data is usually encrypted with a *shared* (symmetric) key. This shared key is then encrypted using the *public* (asymmetric) keys of the recipients and stored as an EncryptedKey element.

4.1.3 P3P

P3P (Platform for Privacy Preferences Project) [16] is an international standard format that provides an automatically exchange and validation of privacy information. P3P helps Internet user to quickly recognize the policies of data protection (collection and use of entered data) in a machine- and human-readable format.

A service provider defines a data protection statement in a text format as well as in P3P format. An Internet user defines his preferences in his own P3P agent, which might be implemented as an add-on tool for his web browser. An example for such preferences could be the blocking of contents or cookies.

While a user retrieves information from the service, his P3P agent compares the user preference with the data protection statement of the service provider. The P3P agent informs the user about differences between user preferences and service provider statement and gives a summary of the relevant data protection information.

4.1.4 WS-Security: SOAP Message Security

WS-Security: SOAP Message Security [12] uses XML Signature and XML Encryption (see above) to suggest extensions to the SOAP messages [17] exchanged during the usage of Web services. So it does not invent new security mechanisms, but provides structures to integrate existing security mechanisms with SOAP. To do this it invents a SOAP security header that contains:

- Security tokens (such as username/password data, an X.509 certificate, a Kerberos ticket or an SAML assertion)
- One or more Signatures
- Information about encrypted parts of this message (`EncryptedKey` or `ReferenceList` element)

WS-Security focuses on message-level security and protects the exchanged messages. In contrast to this, there are other options to secure the communication within a service-oriented architecture, for instance TLS [18] can be used to secure the HTTP transport [19]. Both options have advantages and disadvantages (see Table 1).

	HTTP Transport Layer Security	Message-level Security
Advantages of Transport Layer Security	Mature technology, proven to be working in large scale applications	Immature standard and tools
	Supported by most servers and clients and well understood by system administrators	
	Generally simpler than message-level security	Complex: encompasses many other standards including XML Encryption, XML Signature, X.509 certificates, and many more

Advantages of Message-level Security	Point-to-point: messages are left in the clear after the endpoint	Persistent: Allows the message to be self-protecting
	Waypoint visibility: Cannot have partial visibility into the message	Selective: Portions of the message can be secured to different parties
	Granularity: Cannot have different security for messages in and messages out	Flexible: Different security policy can be applied to request and response Transport independent
	Transport dependent: Applies only to HTTP	

Table 1: Comparison of transport layer and message-level security [14].

4.1.5 WS-Policy

WS-Policy [20] provides mechanisms to exchange requirements between the provider of a Web service and the user/client of this Web service. Some policy assertions specify traditional requirements and capabilities that will ultimately manifest on the wire (e.g., authentication scheme, transport protocol selection). Other policy assertions have no wire manifestation yet are critical to proper service selection and usage (e.g., privacy policy). [21]

4.1.6 WS-Trust

WS-Trust [22] defines extensions that build on WS-Security to provide a framework for the following tasks:

- Requesting, issuing and validating security tokens
- Establish, assess the presence of and broker trust relationships

Within the Web Services Trust Model, a requestor (for instance the client of a Web service) has to authenticate its requests by providing security tokens. A security token is a set of claims (statements) about the client (name, identity, key, group, privilege).

If the requestor does not have the necessary tokens as required by the Web service, it can obtain them from a Security Token Service. These security token services form the basis of trust by issuing tokens which can be used to broker trust relationships between different trust domains.

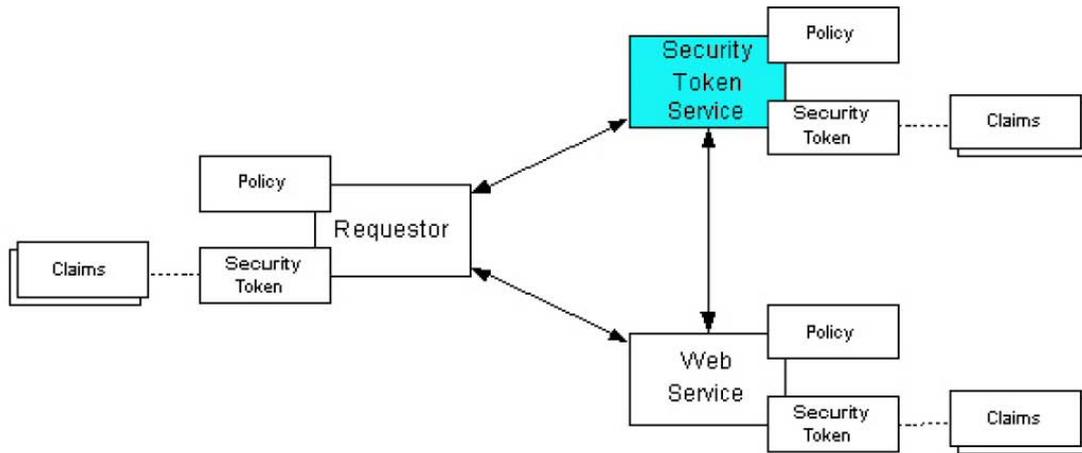


Figure 6: The Web Services Trust Model [22].

This is a general model which subsumes several specific security models, such as identity-based authorization or access control lists. It can be implemented using available and tested technologies such as X.509 certificates, Kerberos tickets or password digests.

4.1.7 WS-SecureConversation

WS-SecureConversation [23] builds on WS-Security and WS-Trust to provide secure communication across one or more messages. Specifically, this specification defines mechanisms for establishing and sharing security contexts, and deriving keys from established security contexts. [21]

Similar to the secure communication channel provided by SSL/TLS on the transport layer, WS-SecureConversation does establish secure communication across a whole sequence of messages on the SOAP message level.

To establish a security context the WS-SecureConversation standard defines a `SecurityContextToken` element. There are three ways to establish such a context:

- The security context token is created by an external Security Token Service, whom the partners trust. This concept is defined by WS-Trust.
- The security context token is generated by one of the partners and propagated with a message.
- The security context token is generated using a challenge response process.

4.1.8 WS-Federation

The target of WS-Federation is to construct federate trust relationships using WS-Policy, WS-Trust, WS-Security and WS-SecureConversation.

This can be used for the mutual use of Web services between a group of enterprises. To do this, the organisations have to agree on a common set of technologies for identification and authentication for the purpose of translating different types of security tokens.

4.1.9 WS-Authorization

WS-Authorization deals with the access control for Web services and in what form access control information will be specified and handled. This standard is still under development and no published standard exists at the time of this writing.

4.1.10 WS-Privacy

WS-Privacy is a specification which is still under development. The idea is that a service provider publishes his privacy policies in a structured form. Concerning the content of such privacy policies WS-Privacy will mainly be influenced by P3P (Platform for Privacy Preferences Project) [16].

By using mechanisms of WS-Security, statements about privacy (claims) can be attached to messages. With WS-Trusts these statements can be evaluated to check them against user preferences or organisational privacy policies.

Since the privacy policy can be published together with other policies about the service, WS-Privacy has to be integrated with the mechanisms of WS-Policy.

4.1.11 WS-Reliability and WS-ReliableMessaging

There are *two* OASIS specifications from different authors dealing with reliable messaging between Web services. Both specifications are edited within an OASIS process:

- WS-Reliability [24] with Fujitsu, Oracle, Sun, and Novell involved
- WS-ReliableMessaging [25] from BEA, IBM, Microsoft, and TIBCO which claims to be more integrated with WS-Security und WS-Policy.

However, it seems that the differences are limited to some syntax or technical details [26] and the two specifications are almost equivalent: They provide reliable message delivery with acknowledgement between Web services.

4.2 Impact of WS-Security and related technologies on SOA

4.2.1 Confidentiality

Confidentiality on a per-message-level can be enforced by the encryption mechanisms of WS-Security. By the encryption of SOAP messages or parts of SOAP messages, unauthorized partners are excluded from listening (“eaves-dropping”).

In addition, the digital signature mechanisms in WS-Security, adopted from XML Signature, ensure that exchanged messages originated from the apparent communication partner. This prevents spoofing attempts, for instance the attempt to trick a Web service into sending confidential information to an attacker by counterfeiting a legitimate request for such information.

Another technology that enforces the confidentiality within SOA is TLS, which can be used to protect protocols such as HTTP (see Table 1).

4.2.2 Privacy

The Web service standard most relevant for the protection of privacy is WS-Privacy. This allows a Web service provider to publish statements on the privacy policies in effect for his service. A user/client can then check whether a Web service matches his privacy preferences.

Privacy in the SOA context can be implemented by mechanisms like P3P-Statements (see 4.1.3). In SOA environments different types of services are integrated. With the information provided by P3P it is possible to ensure the compatibility between these services.

Another privacy related mechanism is the ability of WS-Security to encrypt parts of a message in a way that they can be read by different involved parties. By this it becomes possible to selectively protect privacy relevant data against unwanted analysis by some of the communication partners, but keep the rest of the message accessible to them.

4.2.3 Accountability

Accountability of actions and processes requires knowledge (logging), non-repudiation of origin (signature) and understanding (policies and presentation).

Accountability is measured by the respective partners. Therefore, the first action to support accountability is a reliable log of all actions on the communication interfaces on either side of the cooperation.

In order to prove actions of cooperation partners, the XML Signature schema is available. It offers these functions to Web services:

- The integrity of the signed information is ensured by the calculation of a hash value.
- Non-repudiation of origin is implemented by the digital signature covering the hash value.
- Non-repudiation of delivery can be implemented by requiring the receiving communication partner to countersign and acknowledge a received message with a digital signature. In a similar way, the reliable delivery of messages by WS-Reliability or WS-Reliable Messaging can be used.
- It allows for multiple, distributed and selected signatures.
- XML signatures can cover message content and message headers.

It is not sufficient to implement signatures on the cryptographic level of SOAP messages. Additionally, digital signatures on the message-level must be mapped to responsible persons. This requires the certification of public verification keys to belong to a specified person. This requires also that a SOAP message and the information presented to the signer during the signature process (and to the verifier during the verification process) are equivalent and that the consequence of the signature is known to the users.

Understanding of actions can be supported by clear policies and their comprehensive presentation to the users. This can be implemented in SOA-Environments based on the WS-Policy standard and a concrete implementation like P3P.

4.2.4 Controllability

The goals of controllability such as handling of complexity, handling of diversity, ergonomics and economy of resources cannot be fulfilled by simply following a certain specification. Their implementation lies partly beyond the model of WS-Security.

There is ongoing research on these “soft factors” to influence security, such as social engineering. However, there are some basic rules to be enforced by any Web service to ensure controllability. Number one is a balanced level between complexity of functions versus transparency of the related processes and their composition.

On the design and development level, controllability can be supported by using good software development practices and organisational/economic approaches. For instance the provider of a Web services has to make sure, that his clients get the service that they are expecting and that the amount charged for a service is within the expected limits.

4.2.5 Dependability

Functional reliability of Web services (and their clients) can be supported by a good design of the systems. However, a user or service has no guarantee that a remote cooperation partner is designed to be dependable. In this case, accountability functions such as logging, signatures and comprehensible policies and their local presentation help to compensate for dependability failures.

In order to protect one’s own functionality against broken input formats from remote partners, input data across communication interfaces must be checked for consistency, correctness and reliable origin.

Another dependability property is the ability of a distributed system to continue operation despite the failure of a single Web service (failover). This can be supported by:

- dynamic binding between required functionality and the particular service providing this functionality, for instance by using UDDI directories;
- multiple servers should be available that provide the same functionality. These servers have to be announced, for instance by listing them in the UDDI Directory [27].

These requirements can be fulfilled in ASG: the monitoring solution described in C-5 [35] can notice the failure of a service and initiate a re-planning of a service composition (as described in C-4 [36]), so that the same work that the failed service was supposed to do can be done by some other (redundant) service or even a set of services.

The dependability of a system is threatened by Denial of Service (DoS) attacks. A dependable Web service should be able to detect and try to resist a DoS attack. Should it be impossible to maintain operation, the services should shutdown in a controlled and predictable fashion. In particular, it should never be an option to temporarily disable security mechanisms to increase performance. E.g., a DoS attack that causes a load increase could be detected by monitoring in ASG, which could then lead to appropriate counter-measures such as intercepting the access for the particular client.

4.3 Remaining challenges

4.3.1 WS-Security alone not sufficient

If a web service communication is protected by WS-Security, single messages are protected, but not their communication context. An attacker might still be able to draw information from the analysis of the exchanged communication, even if the single SOAP messages are protected on a per-message-level. For instance, the mere appearance of a SOAP conversation or an increased frequency of exchanged messages might reveal significant information, such as the IP addresses and indirectly the identities of the communication partners.

Replay attacks which reuse captured messages are another threat which is not encountered by protected singular SOAP messages. The additional steps beyond WS-Security include the encryption of whole communication contexts, for example with the help of VPN on the basis of IPSec and SSL. The mapping of data, actions, and organisational subgroups to responsible persons must accompany implemented technical security measures. A typical organisational security measure to ensure data integrity on a business level is the separation-of-duty mechanisms introduced by the Clarke-Wilson-model [37].

4.3.2 Compatibility of specifications

The WS-* family of standards and technologies is evolving but there is no complete set of standards that can be recommended. There are many standards and specifications dealing with the security of Web services, published by several different organisations such as OASIS, the W3C, or software vendor consortia. When choosing a set of specifications, users have to be careful to compose a set of compatible, non-overlapping technologies [27].

4.3.3 Integration of external legacy systems

Even if the Web service architecture of an organization is secured with the appropriate technologies from the WS-* family of standards, the borders of the Web service architecture have to be considered, for instance, when integrating legacy systems.

A legacy system has its own security methods, for instance an authentication by passwords. The security mechanisms of the external system have to be mapped to the Web service implementations. This integration can be seen in two directions:

Web services adapts to external system: A payment service might require that web services which use its functionality accept its policies and security methods, e.g. for the handling of user IDs, passwords and e-mail addresses. ASG's Dynamic Supply Chain scenario prototype is an excellent example, as it successfully integrates PayPal as a payment service.

External system adapts to web service: A music download service, which will use the web service interface of a music distribution system like Potato has to accept and handle the security mechanisms suggested by PotatoSystem (<http://www.potatosystem.com>), such as the used XML Tokens.

4.3.4 Privacy

The only existing privacy enhancing technology in the context of WS Security ready for immediate implementation is P3P. However, it solves only a very small part of privacy problems. P3P is not even sufficient to make privacy policies transparent and comprehensible to users. The composition of policies is not covered by P3P. For example, if a service A relies on two services B and C with two different privacy policies: how can the privacy policy of service A be validated against those of B and C? Complexity of compound services is not only a security challenge for controllability, but also for privacy.

Moreover, it is not sufficient to publish the privacy policy currently in effect for a Web service. Mechanisms are needed to enforce basic privacy principles for the collection, processing and transfer of personal data, such as data minimization, purpose-binding, informed consent, choice without discrimination, and responsible protection of those data. The direct relation of policies to their enforcement is not a technical issue alone. It is also an organisational matter and may be supported by accountability and controllability measures.

The chances of Web services for the improvement of transparency of services can be exploited to enhance privacy issues as well.

This research question is in the focus of the succeeding project “SOAinVO”, performed by the authors of this deliverable in cooperation with other ASG partners (Steffen Staab, Uni Ko-Ld) and the German Federal Ministry for Education and Research (BMBF), during 2007.

5 BEST PRACTICE EXAMPLES

Up to now, SOA security problems as well as possible solutions have been shown. While these were rather theoretical, an insight into real-life practice shall now be given, starting specifically with the ASG platform prototype and leading to a survey on how SOA system users and providers in general deal with some security issues.

5.1 The ASG middleware platform (JBossWS)

The ASG platform prototype is based on JBoss. In this section, we investigate the new Web service stack developed by the JBoss community [38]. It is called *JBossWS*, and we refer to the version 1.0.3G which can be downloaded from the official JBoss site.

JBossWS replaces the previous WS-stack WS4EE, the currently deployed WS-stack at C-5 which has not been supporting security services for Web services. *JBossWS* implements many important features promoting the building of interoperable as well as portable Web services and clients thereof. This includes

- J2EE endpoint development model for EJB and Java (JSR-109). JSR-109 standardises the development of Web services, hence of ASG proxies, in a J2EE container. Consequently, the transformation of the C-3 tool chain can be simplified considerably as with JSR-109 the deployment amongst platform-specific models of an ASG-proxy that is generated by C-3 does not differ substantially. In this context, a platform constitutes a concrete application server that is J2EE compliant.
- J2EE client development model (JSR-109) for ASG proxies. The programming model standardises three mechanisms for invoking an external Web service.
- Web Service Metadata (JSR-181). This specification reduces the amount of information that is needed to setup a Web service, such as an ASG proxy, to a large extent. With this approach, the EJB session bean implementing an ASG proxy interface (*Service Endpoint Interface*, SEI) can be annotated directly with Web services metadata. Consequently, the transformation rules of the C-3 tool chain could be simplified further with the use of metadata defined in JSR-181.

Unfortunately, Web services security annotations (metadata) have not been standardised as yet. From an engineering perspective, also a standard client and service endpoint programming model for security does not exist either. Consequently, the building of portable secure Web services is still a pie in the sky. However, if current WS-Security specifications are implemented by the underlying middleware, secure yet interoperable Web services can be developed and generated from the C-3 tool chain, respectively.

Subsequently, we outline the security features that are supported by the new WS-stack of JBoss. In particular, we focus on WS-Security features that might be relevant and tangible for ASG service integration, namely data confidentiality, integrity, data as well as data origin authentication.

5.1.1 Confidentiality

Sample systems that can be used to employ cryptography of SOAP messages are symmetric-key as well as public-key cryptography. Both systems can be used to encrypt and decrypt data, respectively. The detailed operation of each process is controlled by the algorithm and a key.

An entity is authorised to read sensitive data if it contains the key that is necessary to decrypt the ciphertext. The key represents the secret property. In symmetric cryptography, a shared key is used for encryption and for decryption. Here, the message receivers, such as intermediaries and ultimate external Web service, that share the same key are able to decrypt the corresponding ciphertext.

In asymmetric cryptography, two different but mathematically related keys are utilised – a public key as well as a private key. The public key of an entity can be used to obscure (i.e. encrypt) the data from all entities but from the owner of the paired private key. The private key can be utilised to render the ciphertext intelligible.

Due to performance reasons, among others, hybrid architectures can be required: symmetric cryptography may be used to ensure the confidentiality of a SOAP message that is exchanged between an ASG proxy and the external Web service. The shared symmetric key may be generated at runtime, for a very limited lifetime. It is transmitted through a SOAP message to the ultimate receiver of the SOAP message, here: the external Web service. It is kept confidential as well, through public-key cryptography. It is encrypted with the public key of the external service. The security implementation of the C-3 supports this pattern, as described in [28] (see also figure 28 in [28]). The implementation is interoperable as it conforms to WS-Security [34].

In *JBossWS*, SOAP messages can be kept confidential as specified in WS-Security [34]. Consequently, ASG proxies employing the WS-Security mechanisms supported by *JBossWS* remain interoperable. The complete list of supported and unsupported features as well as configuration features regarding confidentiality can be obtained in [42]. Unfortunately, as yet the configuration of confidentiality is very specific to JBoss. Detailed instructions on configuration of the proprietary policies for different scenarios, similar to the turnkey scenarios implemented with the Web Service Enhancement (WSE) of Microsoft, are lacking yet.

JBoss uses generic JAX-RPC handlers for security computation. This technique is in line with the architecture of the interceptor-based approach that has been implemented by C-3 and introduced in [28], section 5.3. Lastly, information on the implementation of the above mentioned encryption scenario with *JBossWS* can be found in [29]. The schema definition of the security policy supported by *JBossWS* can be obtained in [30].

5.1.2 Integrity and Authenticity

From the service integration point of view, the following security properties might be required, especially if sensitive (business) data are exchanged between ASG (proxies) and external Web services:

- Integrity: Checksums can be used to provide integrity from accidental changes of SOAP messages that are exchanged between ASG (proxies) and external Web services.

- **Data authentication:** In addition to integrity, message authentication also protects against active attacks. In an active attack, sensitive data of a SOAP message are altered. The checksum is changed as well so as to match the change in the data. Message authentication codes can be used here likewise as digital signatures.
- **Data origin (signer) authentication:** Digital signatures can be used to assert that the identity of the signer (e.g. of the ASG platform or even of an ASG proxy) is as claimed [31]. This is due to the fact that someone who does not know the private key cannot forge the correct signature [32].

These properties must be realised if impersonators and forgers need to be excluded from the message exchange between the ASG platform and external Web services. Message authentication and signer authentication are essential ingredients of non-repudiation [33]. A non-repudiation service provides proof of facts to defend against an opponent's effort to avoid a transaction, e.g. of a digital order or of a reservation. These properties correspond to the third requirement accountability.

With WS-Security [34], the three properties listed before can be implemented for Web services' interactions, in an interoperable manner. As mentioned before, *JBossWS* implements WS-Security. Hence, it supports the security properties mentioned previously. The complete list of supported and unsupported features as well as configuration features with regard to digital signatures can be obtained in [42].

A sample SOAP message with a digital signature that is created by an ASG proxy and targeted toward an external Web service was presented in [28], section 5.3. Here also, default policy configurations for various scenarios concerning integrity and/or data origin authentication would improve the usability but do not exist for *JBossWS* to date. Detailed information on the implementation of digital signatures with *JBossWS* can be found in [39].

Finally, a complex example in which credit card information is encrypted and signed with *JBossWS* is presented in [40] (see also [28], section 5.3).

5.1.3 Summary

With the new Web service stack of *JBoss*, the basic security requirements confidentiality, data authentication and data origin authentication have been implemented. Web services can communicate securely and yet remain interoperable. However, as the security annotation method for Web services is specific to *JBossWS*, portability is not ensured. Also other fundamental security concerns, such as security contexts as per instructions in WS-SecureConversation, have not been implemented. This section concludes with a listing of relevant and emerging WS-security related specifications, giving a snapshot of their current implementation status with *JBossWS* [41].

Table 2: Implementation status of WS-Security related specification

WS-Security related specification	% Complete	JBossWS version
WS-Security 1.0	100%	1.0x
WS-Security Username Token Profile 1.0	90%	1.0x
WS-Security X.509 Profile 1.0	90%	1.0x
WS-Security SAML Token Profile 1.0	0%	
WS-Security 1.1 Extensions (Username Token Profile, X.509 Profile, SAML Token Profile, Kerberos Token Profile, SOAP with Attachments Profile)	0%	
WS-SecureConversation, WS-Trust, WS-SX (Security Exchange)	0%	

5.2 Overview of existing SOA system providers and their customers

To successfully take account of existing SOA-related security problems and their solutions, it was a basic necessity to get at least a general idea of SOA-using and SOA-providing businesses. Additionally, data was required to provide the basis for a subsequent survey. Due to the limited time available, it was not possible to take more than only the very first steps in this regard – more detailed information will be available in the aforementioned SOAinVO project.

The following list shows some users of SOA-based systems we could identify so far:

- Albert-Ludwigs-Universität Freiburg
- Apollo Optik
- AWD-Holding
- Bank-Verlag
- BEA Systems
- BMW Group
- BT Global Services
- Bundesland Sachsen, Landesamt für Umwelt und Geologie
- Covad Communications
- DaimlerChrysler AG
- DB Systems GmbH
- Deutsche Leasing
- Deutsche Luftwaffe

- Deutsches Patent- und Markenamt
- Deutsche Post
- Drägerwerk AG
- Helvetia Patria Versicherungen
- Hewlett-Packard
- Hotelzon
- IBM
- Infraserb Höchst
- Lufthansa Technik AG
- MessageLabs
- Nordea
- Nordzucker AG
- Phoenix Contact GmbH & Co. KG
- Pfizer Global Pharmaceuticals
- RTC - Real-Time Center
- TeleCash
- Volkswagen Financial Services AG

They can be roughly grouped in the following branches:

- Banking / Financial Services
- Communications/Media
- Public Sector
- Retail
- Healthcare
- Insurance
- Manufacturing
- Automotive & Transport
- Computer Hardware
- Consumer Packaged Goods
- Education
- Media & Entertainment
- Pharmaceutical

- Real Estate
- Services
- Software
- Transportation
- Travel / Hospitality
- Utilities / Energy

As can be seen in the above lists, many different companies all over the world use SOA systems. A certain minimum business size can be observed as a common denominator in this list: For small businesses, the costs of employing a SOA system (as opposed to a monolithic solution) outweigh its benefits by far, but in larger businesses, where flexibility is becoming increasingly important, SOA can lead the way to Business-IT Alignment. While it seems that SOA is most popular in branches that work with large amounts of data (e.g., Banking and Financial Services), SOA is also used in other kinds of business, depending on the organisational complexity. Examples like the “Deutsches Patent- und Markenamt” (German Patent and Trademark Office) show that the usefulness of SOA has not gone unnoticed in the public sector, either.

The flexibility of the SOA paradigm can be seen not only in the diversity of branches, but also in the concrete uses: some companies only use SOA within their own company limits (e.g., to connect subsidiaries), others use it for Business-to-Business or Business-to-Customer communication.

As providers of SOA-based software, the following companies could be identified:

- Accenture
- BEA Systems
- HP
- IBM
- Microsoft
- Oracle
- SAP
- sd&m
- Software AG

While some of the SOA providers on this list offer “off-the-shelf” software packages which can be customised by their buyers, others (i.e. Accenture and sd&m, who are otherwise known as consultants) have specialized in installing specific custom-built SOA systems for their customers. The role of Microsoft is somewhat special: Unlike most of their competitors, they do not sell a product package that could be dubbed

“SOA suite”, but have integrated WS capabilities into several of their products (in particular into the .NET platform) in order to provide seamless integration of Web services and other (proprietary) technologies. This fits in with Microsoft’s SOA strategy of slow, incremental transitions as opposed to employing large, business-wide solutions right away [43].

5.3 A survey on SOA security practice

Service-oriented architectures have been on the rise during the last five years. In order to gain a first impression of the usage, security aspects and patterns of distribution of SOA-based services, several companies from the aforementioned list were asked to take a survey concerning the services used or designed.

In particular, we put a focus on the degree of utilisation of services and how the processed data are dealt with. The short questionnaire was sent to companies using SOA systems and consisted of the following questions:

- 1.a Does your organisation use SOA?
- 1.b Which system do you use?
- 1.c What kind of services do you offer/use with SOA?

- 2. Do external users have access to your SOA services?

- 3.a Do you process personal data?
- 3.b Do you offer data protection policies?

- 4. How do you ensure data authenticity and integrity with SOA?

- 5.a Are different SOA services coupled?
- 5.b Do you offer one service in different application contexts?
- 5.c What are the costs and benefits of coupling (5.a) and of multiple-use (5.b)?

- 6.a What is your experience with availability?
- 6.b How do you continue your services in case of break-downs?

SOA System Developers and Providers were asked to answer a different set of questions:

- 1. In which contexts do your customers use your SOA systems?
Are there preferred areas of usage?

- 2.a Which mechanisms do your services offer to support authenticity/integrity?
- 2.b Do your customers use these mechanisms?

3. Can your SOA systems be coupled with other SOA systems?
- 4.a1 How error-prone are the SOA/the web services you offer?
- 4.a2 Is the increase or decrease of errors in using SOA (in contrast to other systems) an issue for your customers?
- 4.b Do you offer bridging services to your customers in case of break-downs?

In the short time-frame of the work on this deliverable, only a few responses were received. Therefore, a thorough evaluation of the questionnaire and subsequent reasoning will be postponed to a later publication.

For the time being, the following highlights can be derived from the first answers: Among the companies that answered so far, personal data and data protection policies were not an issue since some SOA implementations did not even deal with personal data. In other cases where SOA is used in a closed environment, it is considered safe enough to allow tasks that process personal data. In these situations, special data protection policies were not considered necessary, as similar policies were prompted elsewhere.

As far as the rather broad terms of data authenticity and integrity are concerned, specific custom-made solutions are in place, such as end-user authorisation via HTTP. Security standards such as WS-Security are becoming more and more important, with most SOA user either having implemented them or planning to implement them.

6 CONCLUSION

The vision of the ASG project is an open development platform for adaptive and reliable matchmaking discovery, composition, and enactment. As a result, open services can be dynamically coupled to enhance functionality and increase value to their users. Based on semantic specifications of requested services by service customers, ASG discovers appropriate services, creates composed services and enacts them. Security, though, was not part of the original ASG design. However, the vision of an open landscape of dynamic service components which may be composed (service complexity) or which may be used in mutual alternatives (component diversity) provokes a series of security questions. Three main security problems are identified

1. *Openness*: when autonomous services are free to enter and leave the market place, how can binding communication (like contracts, payment, service delivery) be trusted?
2. *Assertions* with (a) complexity and (b) diversity of services: accountability of services can be supported by service and security assertions. (a) How can the transitivity of service and security assertions be enforced in a composition of services? (b) How can service and security assertions remain stable across different alternative services? For example, how do fairness assertions survive the exchange of payment and delivery services within an online purchase chain?
3. *Privacy*: privacy is not even satisfactorily solved in the WWW of single services. It is even more a problem when services are composed. How can purpose binding, data minimization and transparency of privacy policies be enforced in a complex world of free service compositions? How do different service provided from different regulation environments (e.g. the USA and EU) handle privacy?

The ASG middleware platform allows to utilize basic security functions such as signatures and encryption. This is a first step into the right direction in order to provide flexible, open and secure services and composition of services.

However, current SOA applications show that today open services are only used within closed islands which are under central control of big organizations. In order to reduce the complexity problem, SOA must be securely manageable in an open management environment. Otherwise, SOA wouldn't scale. However, in order to open the services to the coupling with external services, the security problem of openness, assertions, and privacy must be solved. Otherwise these applications will not happen. In other words, *security technology* must be developed and implemented which *enables applications* in growing, open environments. This is a challenge for further research. The challenge is to enable complexity and diversity in a usable, scalable, and trustworthy (secure) manner.

REFERENCES

- [1] United States Department of Defense: Trusted Computer System Evaluation Criteria (TCSEC, "Orange Book"), DoD5200.28-TD, Dec.1985. <http://csrc.nist.gov/publications/history/dod85.pdf>
- [2] Whitman, Michael E.; and Mattord, Herbert: Principles of Information Security. Course Technology, ISBN 0-619-06318-1, 2002.
- [3] Avizienis, Algirdas; Laprie, Jean-Claude; Randell, Brian; and Landwehr, Carl: Basic Concepts and Taxonomy of Dependable and Secure Computing. IEEE Transactions on Dependable and Secure Computing, Vol. 1, No.1 Jan-March 2004.
- [4] CC/ISO: Common Criteria, Security Evaluation. Version 3.1, Revision 1, Sep 2006. ISO/IEC 15408:2006. <http://www.bsi.bund.de/cc/> [8.1.2007] and <http://www.commoncriteriaportal.org/> [8.1.2007]
- [5] Dierstein, Rüdiger: Sicherheit in der Informationstechnik - der Begriff IT-Sicherheit. Informatik Spektrum, Volume 27, pp. 343-353, 2004
- [6] Hiles, A.; and Barnes, P.: The Definitive Handbook of Business Continuity Management, Chichester, New York, Weinheim: JohnWiley and Sons.
- [7] U.S. Department of Commerce: Safe Harbor. <http://www.export.gov/safeHarbor/> [10.1.2007]
- [8] EU: 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, http://www.cdt.org/privacy/eudirective/EU_Directive_.html [10.1.2007]
- [9] Cottrell, Lance: Mixmaster & Remailer Attacks. Original from <http://www.obscura.com/~loki/remailer/remailer-essay.html> [not available in original]. German translation available from Donnerhacke, Lutz (2005): Mixmaster- & Remailer-Angriffe. <http://www.iks-jena.de/mitarb/lutz/anon/remailer-essay.html> [22.1.2007]
- [10] Universities of Regensburg and Dresden (2000-2006): Project: AN.ON - Anonymity.Online. http://anon.inf.tu-dresden.de/index_en.html [22.1.2007]
- [11] World Wide Web Consortium (2002): The Platform for Privacy Preferences 1.0 (P3P1.0) Specification. W3C Recommendation 16 April 2002. <http://www.w3.org/TR/P3P/> [23.1.2007]
- [12] A. Nadalin, C. Kaler, R. Monzillo, and P. Hallam-Baker, "Web Services Security: SOAP Message Security 1.1 (WS-Security 2004)," 2006
- [13] D. Eastlake, J. Reagle, and D. Solo, "XML-Signature Syntax and Processing," World Wide Web Consortium, 2002
- [14] J. B. Rosenberg and D. L. Remy, Securing Web services with WS-Security : demystifying WS-Security, WS-Policy, SAML, XML Signature, and XML Encryption. Indianapolis, IN: SAMS, 2004
- [15] D. Eastlake and J. Reagle, "XML Encryption Syntax and Processing," World Wide Web Consortium, 2002
- [16] M. Marchiori, "The Platform for Privacy Preferences 1.0 (P3P1.0) Specification," World Wide Web Consortium, 2002
- [17] N. Mitra, "SOAP Version 1.2 Part 0: Primer," World Wide Web Consortium, 2002
- [18] T. Dierks and E. Rescorla, "RFC4346: The Transport Layer Security (TLS) Protocol -- Version 1.1," IETF RFC4346, 2006
- [19] E. Rescorla, "RFC2818: HTTP Over TLS," IETF RFC2818, 2000

- [20] J. Schlimmer, "Web Services Policy Framework (WSPolicy) -- Version 1.2," 2006
- [21] ASG, "C5 Activity Report -- Web Service Standards (Internal Project Documentation)," 2007
- [22] S. Anderson, "Web Services Trust Language (WS-Trust)," 2005
- [23] A. Nadalin, "Web Services Secure Conversation Language (WS-SecureConversation)," 2005
- [24] K. Iwasa, "WS-Reliability 1.1," OASIS, 2004
- [25] D. Davis, A. Karmarkar, G. Pilz, S. Winkler, and Ü. Yalçinalp, "Web Services Reliable Messaging (WS-ReliableMessaging) -- Committee Draft 04, August 11, 2006," OASIS, 2006
- [26] D. Chappell and R. Cover, "WS-Reliability and WS-ReliableMessaging," in The Cover Pages, 2005
- [27] A. Singhal and T. Winograd, "Guide to Secure Web Services (Draft) NIST Special Publication 800-95," 2006
- [28] Mehr, F., Cabanis, N.: Redefined Design of Service Integration and Development Tools. Adaptive Services Grid Deliverable D3.K-5, March 17, 2006.
- [29] JBoss Wiki: Encryption example:
<http://wiki.jboss.org/wiki/Wiki.jsp?page=WSSecurityEncryptExample>, December 28, 2006.
- [30] JBoss Wiki: Schema definition of the JBoss WS-Security policy:
http://wiki.jboss.org/wiki/attach?page=WSSecurityConfig%2Fjboss-ws-security_1_0.html, December 30, 2006.
- [31] Eastlake, D.; Reagle, J.; Solo, D.: The Web Service Modeling Framework WSMF. XML-Signature Syntax and Processing. W3C Recommendation, February 12, 2002.
- [32] Shirey, R.: RFC 2828 – Internet Security Glossary, May 1, 2000.
- [33] American Bar Association: Digital Signature Guidelines, August 1, 1996.
- [34] Nadaln, A.; Kaler, C.; Hallam-Baker, P.; Monzillo, R.: Web Service Security: SOAP Message Security 1.0 (WS-Security). OASIS Standard, March 1, 2004.
- [35] Flehmig, M.; Troeger, P.; Saar, A.: Design and Integration of SLA Monitoring and Negotiation Capabilities. ASG Key Deliverable D5.II-7, August 2006. https://asg-platform.org/twiki/pub/Internal/D5_II-7/D5.II-7-Color.pdf
- [36] Wu, B.;Zhang, J.Y.: Documented Mechanisms for Meditated Service Process Adaptation. ASG Internal Deliverable D4.IV-3, March 2006. https://asg-platform.org/twiki/pub/Internal/D4_IV-3/D4.IV-3_01.pdf
- [37] Clark, D.; Wilson, D. (1987): A Comparison of Commercial and Military Security Policies. Proceedings of the 1987 IEEE Symposium on Security and Privacy, Oakland, CA. Computer Society Press of the IEEE, Washington DC, 184-194, 1987.
- [38] JBoss Web Service community: JBossWS:
<http://wiki.jboss.org/wiki/Wiki.jsp?page=JBossWS>, December 15, 2006.
- [39] JBoss Wiki: Message Signing example:
<http://wiki.jboss.org/wiki/Wiki.jsp?page=WSSecuritySignExample>, December 31, 2006.
- [40] JBoss Wiki: Complex example:
<http://wiki.jboss.org/wiki/Wiki.jsp?page=WSSecurityComplexExample>, December 31, 2006.
- [41] JBoss Wiki: WS-* specifications implementation status:
<http://wiki.jboss.org/wiki/Wiki.jsp?page=JBossWSSpecStatus>, December 13, 2006.
- [42] JBoss Wiki: WS-Security features:
<http://wiki.jboss.org/wiki/Wiki.jsp?page=WSSecurityFeatures>, December 19, 2006.

- [43] Microsoft: Microsoft Highlights “Real-World” Approach to SOA at SOA & Business Process Conference, <http://www.microsoft.com/presspass/press/2006/oct06/10-04PragmaticSOAPR.msp>, [20.02.2007]

PROJECT CONSORTIUM INFORMATION

Partner	Acronym	Contact
University of Potsdam, Germany		Dr. Regina Gerber Universitaet Potsdam Am Neuen Palais 10 D-14469 POTSDAM Germany Email: rgerber@rz.uni-potsdam.de Tel: +49-331-9771080
University of Leipzig, Germany	 Faculty of Economics and Management Information Systems Institute	Prof. Bogdan Franczyk Universitaet Leipzig Ritterstrasse 26 D-04109 LEIPZIG Germany Email: franczyk@wifa.uni-leipzig.de Tel: +49.341-33720
University of Innsbruck, Austria		Prof. Dieter Fensel Institute of computer science University of Innsbruck Technikerstr. 25 A-6020 Innsbruck Austria Email: dieter.fensel@deri.org Tel: +43-512-5076488
Fraunhofer IESE, Germany	 Institut Experimentelles Software Engineering	Dr. Dirk Muthig Fraunhofer-Gesellschaft zur Foerderung der angewandten Forschung e. V. Hansastrasse 27C, D-80686 MUENCHEN Germany Email: muthig@iese.fhg.de Tel: +49-6301-707161
DaimlerChrysler Research, Germany		DI Jens Weiland DaimlerChrysler AG Postfach 2360 D-89013 Ulm Germany Email: jens.weiland@daimlerchrysler.com Tel: +49-731-5052404
HPI at University Potsdam, Germany		Hasso-Plattner-Institut fuer Softwaresystemtechnik gGMBH Prof.-Dr.-Helmert-Strasse 2-3 D-14482 POTSDAM Germany Prof. Mathias Weske Email: Mathias.Weske@hpi.uni-potsdam.de Tel: +49-331-5509191 Prof. Andreas Polze Email: andreas.polze@hpi.uni-potsdam.de Tel: +49 331 5509 231
NUIG, Ireland	 Ollscoil na hÉireann, Gaillimh National University of Ireland, Galway	Prof. Christoph Bussler National University of Ireland Science and Engineering Technology Building Galway Ireland Email: chris.bussler@deri.ie Tel: +353-87-6826940

Swinburne University, Australia		Prof. Ryszard Kowalczyk Swinburne University of Technology PO Box -218 AUS-3122 HAWTHORN Australia Email: rkowalczyk@it.swin.edu.au Tel: +61-39-2145834
Thüringer Anwendungszentrum fuer Software-, Informations- und Kommunikations-technologie GmbH, Germany		DI Holger Krause Thüringer Anwendungszentrum fuer Software-, Informations- und Kommunikations-technologie GmbH Langewiesener Strasse 32 D-98693 ILMENAU Germany Email: Krause@transit-online.de Tel: +49-3677-845109
NIWA, Austria		DI Alexander Wahler NIWA-WEB Solutions Niederacher & Wahler OEG Kirchengasse 13/1a A-1070 VIENNA Austria Email: wahler@niwa.at Tel: +43-131-9584311
Telenor Communications II AS, Norway		Dr. Rolf. B. Haugen Telenor ASA Snarøyveien 30 N-1331 FORNEBU Norway Email: rolf-bjorn.haugen@telenor.com Tel: +47-900-51101
Siemens AG, Germany		DI Klaus Jank Siemens AG Corporate Technology, Software and System Architecture Otto-Hahn-Ring 6 D-81730 MUENCHEN Germany Email: klaus.jank@siemens.com Tel: +49-89-636-50573
Rodan Systems, Poland		Mariusz MOMOTKO Rodan Systems Spolka Akcyjna Ul. Pulawska 465 PL-02-844 WARSZAWA Poland Email: Mariusz.Momotko@rodan.pl Tel: +48-58-5502024
University Jyväskylä, Finland		Prof. Jari Antti VEIJALAINEN Jyväskylän Yliopisto Seminaarinkatu 15 FI-40014 JYVASKYLA Finland Email: jari.veijalainen@titu.jyu.fi Tel: +358-14-2603021
Telekomunikacja Polska, Poland		Bogdan BANSIK Telekomunikacja Polska S.A. Ul. Obrzeźna 7 PL-02-691 WARSZAWA Poland Email: Bogdan.Banskiak@telekomunikacja.pl Tel: +48-22-6995340

Marketplanet, Poland		Otwarty Rynek Elektroniczny S.A. Ul. Domaniewska 41 PL-02-672 WARSZAWA Poland Email: info@marketplanet.pl Tel: +48 22 576 88 00
ASTECSp. z o.o., Poland		Janusz MICHALEWICZ ASTECSp. Z O.O. Ul. Piaskowa 14 PL-65-209 ZIELONA GORA Poland Email: J.Michalewicz@astec.com.pl Tel: +48-68-3298031
The Poznan University of Economics, Poland		Prof. Witold ABRAMOWICZ Akademia Ekonomiczna W Poznaniu Al. Niepodleglosci 10 PL-60-967 POZNAN Poland Email: W.Abramowicz@kie.ae.poznan.pl Tel: +48-61-8569333
FH Furtwangen, Germany		Prof. Ulf Schreier University of Applied Sciences Furtwangen Rogert-Gerwig-Platz 1 D-78120 FURTWANGEN Germany Email: schreier@fh-furtwangen.de Tel: +49-7723-920153
Polska Telefonía Cyfrowa, Poland		Longin BRZEZINSKI Polska Telefonía Cyfrowa SP. Z O.O. Al. Jerozolimskie 181 PL-02-222 WARZAWA Poland Email: lbrzezinski@era.pl Tel: +48-22-4135808
University of Koblenz-Landau		Prof. Steffen Staab Institute of Computer Science Universität Koblenz-Landau PO Box 201 602 56016 Koblenz Germany Email: staab@uni-koblenz.de Tel: +49-261-287 2761
Erik Lillevold		Erik Lillevold Åsheimneien 33 2016 Frogner Norway Email: erlille@online.no Tel: +47-9134-4641

Bisher erschienen

Arbeitsberichte aus dem Fachbereich Informatik

(<http://www.uni-koblenz.de/fb4/publikationen/arbeitsberichte>)

Rüdiger Grimm, Farid Mehr, Anastasia Meletiadou, Daniel Pähler, Ilka Uerz: SOA-Security, Arbeitsberichte aus dem Fachbereich Informatik 19/2007

Christoph Wernhard: Tableaux Between Proving, Projection and Compilation, Arbeitsberichte aus dem Fachbereich Informatik 18/2007

Ulrich Furbach, Claudia Obermaier: Knowledge Compilation for Description Logics, Arbeitsberichte aus dem Fachbereich Informatik 17/2007

Fernando Silva Parreiras, Steffen Staab, Andreas Winter: TwoUse: Integrating UML Models and OWL Ontologies, Arbeitsberichte aus dem Fachbereich Informatik 16/2007

Rüdiger Grimm, Anastasia Meletiadou: Rollenbasierte Zugriffskontrolle (RBAC) im Gesundheitswesen, Arbeitsberichte aus dem Fachbereich Informatik 15/2007

Ulrich Furbach, Jan Murray, Falk Schmidsberger, Frieder Stolzenburg: Hybrid Multiagent Systems with Timed Synchronization-Specification and Model Checking, Arbeitsberichte aus dem Fachbereich Informatik 14/2007

Björn Pelzer, Christoph Wernhard: System Description: "E-KRHyper", Arbeitsberichte aus dem Fachbereich Informatik, 13/2007

Ulrich Furbach, Peter Baumgartner, Björn Pelzer: Hyper Tableaux with Equality, Arbeitsberichte aus dem Fachbereich Informatik, 12/2007

Ulrich Furbach, Markus Maron, Kevin Read: Location based Information systems, Arbeitsberichte aus dem Fachbereich Informatik, 11/2007

Philipp Schaer, Marco Thum: State-of-the-Art: Interaktion in erweiterten Realitäten, Arbeitsberichte aus dem Fachbereich Informatik, 10/2007

Ulrich Furbach, Claudia Obermaier: Applications of Automated Reasoning, Arbeitsberichte aus dem Fachbereich Informatik, 9/2007

Jürgen Ebert, Kerstin Falkowski: A First Proposal for an Overall Structure of an Enhanced Reality Framework, Arbeitsberichte aus dem Fachbereich Informatik, 8/2007

Lutz Priese, Frank Schmitt, Paul Lemke: Automatische See-Through Kalibrierung, Arbeitsberichte aus dem Fachbereich Informatik, 7/2007

Rüdiger Grimm, Robert Krimmer, Nils Meißner, Kai Reinhard, Melanie Volkamer, Marcel Weinand, Jörg Helbach: Security Requirements for Non-political Internet Voting, Arbeitsberichte aus dem Fachbereich Informatik, 6/2007

Daniel Bildhauer, Volker Riediger, Hannes Schwarz, Sascha Strauß, „grUML – Eine UML-basierte Modellierungssprache für T-Graphen“, Arbeitsberichte aus dem Fachbereich Informatik, 5/2007

Richard Arndt, Steffen Staab, Raphaël Troncy, Lynda Hardman: Adding Formal Semantics to MPEG-7: Designing a Well Founded Multimedia Ontology for the Web, Arbeitsberichte aus dem Fachbereich Informatik, 4/2007

Simon Schenk, Steffen Staab: Networked RDF Graphs, Arbeitsberichte aus dem Fachbereich Informatik, 3/2007

Rüdiger Grimm, Helge Hundacker, Anastasia Meletiadou: Anwendungsbeispiele für Kryptographie, Arbeitsberichte aus dem Fachbereich Informatik, 2/2007

Anastasia Meletiadou, J. Felix Hampe: Begriffsbestimmung und erwartete Trends im IT-Risk-Management, Arbeitsberichte aus dem Fachbereich Informatik, 1/2007

„Gelbe Reihe“

(<http://www.uni-koblenz.de/fb4/publikationen/gelbereihe>)

Lutz Priese: Some Examples of Semi-rational and Non-semi-rational DAG Languages.
Extended Version, Fachberichte Informatik 3-2006

Kurt Lautenbach, Stephan Philippi, and Alexander Pinl: Bayesian Networks and Petri Nets,
Fachberichte Informatik 2-2006

Rainer Gimnich and Andreas Winter: Workshop Software-Reengineering und Services,
Fachberichte Informatik 1-2006

Kurt Lautenbach and Alexander Pinl: Probability Propagation in Petri Nets, Fachberichte
Informatik 16-2005

Rainer Gimnich, Uwe Kaiser, and Andreas Winter: 2. Workshop "Reengineering Prozesse" –
Software Migration, Fachberichte Informatik 15-2005

Jan Murray, Frieder Stolzenburg, and Toshiaki Arai: Hybrid State Machines with Timed
Synchronization for Multi-Robot System Specification, Fachberichte Informatik 14-2005

Reinhold Letz: FTP 2005 – Fifth International Workshop on First-Order Theorem Proving,
Fachberichte Informatik 13-2005

Bernhard Beckert: TABLEAUX 2005 – Position Papers and Tutorial Descriptions,
Fachberichte Informatik 12-2005

Dietrich Paulus and Detlev Droege: Mixed-reality as a challenge to image understanding and
artificial intelligence, Fachberichte Informatik 11-2005

Jürgen Sauer: 19. Workshop Planen, Scheduling und Konfigurieren / Entwerfen, Fachberichte
Informatik 10-2005

Pascal Hitzler, Carsten Lutz, and Gerd Stumme: Foundational Aspects of Ontologies,
Fachberichte Informatik 9-2005

Joachim Baumeister and Dietmar Seipel: Knowledge Engineering and Software Engineering,
Fachberichte Informatik 8-2005

Benno Stein and Sven Meier zu Eißel: Proceedings of the Second International Workshop on
Text-Based Information Retrieval, Fachberichte Informatik 7-2005

Andreas Winter and Jürgen Ebert: Metamodel-driven Service Interoperability, Fachberichte
Informatik 6-2005

Joschka Boedecker, Norbert Michael Mayer, Masaki Ogino, Rodrigo da Silva Guerra,
Masaaki Kikuchi, and Minoru Asada: Getting closer: How Simulation and Humanoid League
can benefit from each other, Fachberichte Informatik 5-2005

Torsten Gipp and Jürgen Ebert: Web Engineering does profit from a Functional Approach,
Fachberichte Informatik 4-2005

Oliver Obst, Anita Maas, and Joschka Boedecker: HTN Planning for Flexible Coordination Of
Multiagent Team Behavior, Fachberichte Informatik 3-2005

Andreas von Hessling, Thomas Kleemann, and Alex Sinner: Semantic User Profiles and their
Applications in a Mobile Environment, Fachberichte Informatik 2-2005

Heni Ben Amor and Achim Rettinger: Intelligent Exploration for Genetic Algorithms –
Using Self-Organizing Maps in Evolutionary Computation, Fachberichte Informatik 1-2005