NORTH ATLANTIC TREATY ORGANISATION

RESEARCH AND TECHNOLOGY ORGANISATION

AC/323(IST-061)TP/52

www.rto.nato.int

RTO TECHNICAL REPORT **TR-IST-061**

Secure Service Oriented Architectures (SOA) Supporting NEC

(Architecture orientée service (SOA) gérant la NEC)

This Report documents the findings of

NATO/RTO Task Group IST-061.

Published January 2009

Distribution and Availability on Back Cover

NORTH ATLANTIC TREATY ORGANISATION

RESEARCH AND TECHNOLOGY ORGANISATION

AC/323(IST-061)TP/52

www.rto.nato.int

RTO TECHNICAL REPORT **TR-IST-061**

Secure Service Oriented Architectures (SOA) Supporting NEC

(Architecture orientée service (SOA) gérant la NEC)

This Report documents the findings of

NATO/RTO Task Group IST-061.

Editors

Peter-Paul MEILER, TNO, Netherlands
Michael SCHMEING, FGAN, Germany

Chapter Authors

Anders EGGEN, FFI, Norway
Raymond HAAKSETH, FFI, Norway
Marek MAŁOWITZKI, MCI, Poland
Rolf RASMUSSEN, FFI, Norway
Jean-Yves SAVERY, EADS, France
Marc SLIMANI, Thales, France

Additional Authors

Ole-Erik HEDENSTAD, FFI, Norway
Dominique GEORGEL, Thales, France

The Research and Technology Organisation (RTO) of NATO

RTO is the single focus in NATO for Defence Research and Technology activities. Its mission is to conduct and promote co-operative research and information exchange. The objective is to support the development and effective use of national defence research and technology and to meet the military needs of the Alliance, to maintain a technological lead, and to provide advice to NATO and national decision makers. The RTO performs its mission with the support of an extensive network of national experts. It also ensures effective co-ordination with other NATO bodies involved in R&T activities.

RTO reports both to the Military Committee of NATO and to the Conference of National Armament Directors. It comprises a Research and Technology Board (RTB) as the highest level of national representation and the Research and Technology Agency (RTA), a dedicated staff with its headquarters in Neuilly, near Paris, France. In order to facilitate contacts with the military users and other NATO activities, a small part of the RTA staff is located in NATO Headquarters in Brussels. The Brussels staff also co-ordinates RTO's co-operation with nations in Middle and Eastern Europe, to which RTO attaches particular importance especially as working together in the field of research is one of the more promising areas of co-operation.

The total spectrum of R&T activities is covered by the following 7 bodies:

- AVT Applied Vehicle Technology Panel
- HFM Human Factors and Medicine Panel
- IST Information Systems Technology Panel
- NMSG NATO Modelling and Simulation Group
- SAS System Analysis and Studies Panel
- SCI Systems Concepts and Integration Panel
- SET Sensors and Electronics Technology Panel

These bodies are made up of national representatives as well as generally recognised 'world class' scientists. They also provide a communication link to military users and other NATO bodies. RTO's scientific and technological work is carried out by Technical Teams, created for specific activities and with a specific duration. Such Technical Teams can organise workshops, symposia, field trials, lecture series and training courses. An important function of these Technical Teams is to ensure the continuity of the expert networks.

RTO builds upon earlier co-operation in defence research and technology as set-up under the Advisory Group for Aerospace Research and Development (AGARD) and the Defence Research Group (DRG). AGARD and the DRG share common roots in that they were both established at the initiative of Dr Theodore von Kármán, a leading aerospace scientist, who early on recognised the importance of scientific support for the Allied Armed Forces. RTO is capitalising on these common roots in order to provide the Alliance and the NATO nations with a strong scientific and technological basis that will guarantee a solid base for the future.

The content of this publication has been reproduced
directly from material supplied by RTO or the authors.

Published January 2009

ISBN 978-92-837-0069-2

Table of Contents

List of Figures/Tables

Figures **Page**

Tables

List of Acronyms

API	Application Programmer Interface
C2IEDM	C2 Information Exchange Data Model
CFBL	Combined Federated Battle Laboratories
CFBLNet	Combined Federated Battle Laboratories Network
CIS	Communication and Information System
COI	Community of Interest
COTS	Commercial Off The Shelf
CWID	Combined Warrior Interoperability Demonstrator
DPN	Data Publishing Node
ESS	Enhanced Security Services
FFI	Forsvarets Forsknings Institutt (Norwegian Defence Research Establishment)
FGAN	Forschungsgesellschaft für Angewandte Naturwissenschaften (Research Establishment for Applied Science)
FTP	File Transfer Protocol
GW	Gateway
HTTP	Hypertext Transfer Protocol
IEG	Information Exchange Gateway
IETF	Internet Engineering Task Force
IP	Internet Protocol
IS	Information System Technology
IST	Information System Technology
JMS	Java Messaging Service
LAN	Local Area Network
LCC	Land Command Centre
LDAP	Lightweight Directory Access Protocol
MCC	Maritime Command Centre
MCI	Military Communications Institute
MIME	Multipurpose Internet Mail Extensions
MIP	Multilateral Interoperability Programme
NATO	North Atlantic Treaty Organisation
NEC	Network Enabled Capabilities
NNEC	NATO NEC
NNEC FS	NATO NEC Feasibility Study
NO	Norway NATO Open Systems WG
OASIS	Organisation for the Advancement of Structured Information Standards
PKI	Public Key Infrastructure

RFC	Request for Comment
RTA	Research and Technology Agency
RTG	Research Task Group
RTO	Research and Technology Organisation
SMTP	Simple Mail Transfer Protocol
SOA	Service Oriented Architecture
SPC	Security Protection Component
STARS	STructure d'Accueil pour le Renseignement et la Surveillance
TNO	Nederlandse Organisatie voor Toegepast Natuurwetenschappelijk Onderzoek
UAV	Unmanned Aerial Vehicle
UDDI	Universal Description Discovery & Integration
URL	Uniform Resource Locator
WAN	Wide Area Network
WS	Web Services
WSDL	Web Services Description Language
XML	Extensible Markup Language
XMLSEC	XML Security

Programme Committee

CHAIRMEN

- Torleiv MASENG (FFI, Norway)
- Gilbert MULTEDO (Thales, France)

EDITORS

- Peter-Paul MEILER (TNO, Netherlands)
- Michael SCHMEING (FGAN, Germany)

CHAPTER AUTHORS

- Anders EGGEN (FFI, Norway)
- Raymond HAAKSETH (FFI, Norway)
- Marek MAŁOWITZKI (MCI, Poland)
- Rolf RASMUSSEN (FFI, Norway)
- Jean-Yves SAVERY (EADS, France)
- Marc SLIMANI (Thales, France)

ADDITIONAL AUTHORS

- Ole-Erik HEDENSTAD (FFI, Norway)
- Dominique GEORGEL (Thales, France)

Authors of Demonstrator Specification

These people have contributed to the specification of the CWID 2006 demonstrator:

Anders EGGEN (FFI, Norway) – Editor
Morten ANDREASSEN (FFI, Norway)
Dominique GEORGEL (Thales, France)
Ole Erik HEDENSTAD (FFI, Norway)
Raymond HAAKSETH (FFI, Norway)
Herve JAMET (Thales, France)
Anders LANGMYR (FFI, Norway)
Marek MAŁOWITZKI (MCI, Poland)
Rolf RASMUSSEN (FFI, Norway)
Pierre SASSUS (Thales, France)
Geir SLETTEN (FFI, Norway)

Secure Service Oriented Architectures (SOA) Supporting NEC

(RTO-TR-IST-061)

Executive Summary

In military scenarios, especially in joint and combined operations at the tactical level, interoperability among different units is increasingly becoming a key factor for success. Current military computerized systems on the other hand are often designed as stand-alone systems with no or only proprietary non-standard network connections. Additionally, similar systems of different nations or even systems of forces of the same nation often can not interoperate. A technology called "Service Oriented Architecture" (SOA) has been identified as a possible solution to create interoperability between such systems. NATO/RTO IST-061 RTG-027 on "Secure Software Oriented Architectures (SOA) Supporting NEC" has examined the possibilities and challenges arising from these technologies and presented a demonstration of a possible implementation.

The basic concept of SOA is built around services offered by Service Providers and consumed or used by Service Consumers. A Service Registry allows Service Consumers to locate the offered services. The objective of the group has been to develop a SOA-based solution for challenges arising in three technical areas when interconnecting tactical systems of different nations. These areas were the publication and localization of available services; the efficient dissemination of information between information producers and consumers and the security of the communication. At CWID 2006, the group demonstrated an implementation of the solution.

During the demonstration it became clear that current standards and implementations are not yet mature enough for operational deployment on a large scale. Even though products may claim to support a given standard, interoperability with other implementations of that standard is not always possible. Also, quite a few standards in the area of SOA are not yet stable or lack important features such as security considerations. Nevertheless, it has been shown that the basic principles of SOA provide a sound foundation for future military communication networks intended to interoperate in joint and combined scenarios. SOA provides a flexible, comparatively easily deployable and scalable networking concept for the tactical environment. As SOA has been identified by the NATO Network Enabled Capabilities Feasibility Study as the way to achieve its objectives, the work of IST-061 is in line with the study.

The idea of not only creating theoretical work but to create an actual implementation has provided input that can be used for the estimation of effort required to field systems based on the considered principles. Several problems for example with interconnecting systems or with incompatibilities between implementations have only been discovered during the implementation of the demonstrator. On the other hand, the situation has been complicated by the fact that the individual contributions were located in different rooms and had to adhere to different national security policies. The provision of a room dedicated to RTO demonstrations would have made things easier.

Architecture orientée service (SOA) gérant la NEC

(RTO-TR-IST-061)

Synthèse

Dans les scénarios militaires, particulièrement dans les opérations interarmées et interalliées au niveau tactique, l'interopérabilité entre les différentes unités est devenu un facteur clé du succès. D'un autre côté, les systèmes informatisés militaires actuels sont souvent conçus en tant que systèmes autonomes, sans connexion ou connectés qu'à un réseau spécialisé non standard. De plus, des systèmes similaires de différentes nations, voire même des systèmes de forces d'une même nation peuvent ne pas interopérer. Une technique dite « Architecture orientée service » (SOA) a été identifiée comme solution possible d'interopérabilité entre de tels systèmes. Le groupe RTG-027 IST-061 du RTO de l'OTAN sur les « Architectures logicielles sécurisées orientées logiciel (SOA) gérant la NEC » a examiné les possibilités et défis posés par ces techniques et fait la démonstration d'une mise en œuvre possible.

L'idée de base du SOA est construite autour des services proposés par les FAI (Fournisseurs d'Accès Internet) et consommés/utilisés par les clients/consommateurs. Un annuaire des services permet aux consommateurs de situer les services proposés. L'objectif du groupe a été de développer une solution basée sur la SOA pour les défis posés par trois domaines techniques lorsqu'on interconnecte les systèmes tactiques de différentes nations. Ces domaines sont : la publication et localisation des services disponibles, la dissémination efficace des informations entre producteurs et consommateurs, et la sécurité des communications. Au CWID 2006, notre groupe de travail a fait la preuve de sa possibilité et mis en œuvre la solution.

Durant cette démonstration il est apparu que les normes actuelles et mises en œuvre n'ont pas encore la maturité nécessaire à un déploiement opérationnel à grande échelle. Même si les produits prétendent gérer une norme donnée, l'interopérabilité avec d'autres mises en œuvre de cette même norme n'est pas toujours possible. Bon nombre de normes en matière de SOA ne sont toujours pas stables ou manquent même de fonctions importantes comme sécurité/fiabilité. Néanmoins, il a été montré que les principes de base de la SOA constituent une base saine pour les réseaux de communications militaires futurs, destinés à interopérer dans des scénarios interarmées et interalliés. La SOA permet à l'environnement tactique de bénéficier d'un concept de réseau souple, relativement aisé à déployer et modulaire. Comme la SOA a été identifiée par l'Etude de Faisabilité des Capacités Réseau de l'OTAN comme moyen de parvenir à des objectifs, le travail du IST-061 est dans la logique de cette étude.

L'idée de non seulement créer une tâche théorique, mais aussi de mettre en œuvre réellement, a généré une impulsion utilisable pour l'estimation de l'effort nécessaire pour installer ces systèmes sur le terrain, fondées sur les principes considérés. Lors de la mise en œuvre de cette démonstration, seulement quelques problèmes, par exemple d'interconnexion entre systèmes ou d'incompatibilité entre les mises en œuvre, ont été découverts. D'autre part, la situation a été compliquée par le fait que les individus étaient dans des pièces différentes et que chacun devaient contribuer différemment selon des politiques sécuritaires nationales différentes. L'attribution d'une pièce spéciale pour les démonstrations du RTO aurait facilité les choses.

Chapter 1 – INTRODUCTION AND BACKGROUND

1.1 INTRODUCTION

The IST-061/RTG-027 Task Group was established in December 2004 with a planned duration of two years, until the end of 2006. The following companies and organizations participated in the Group's work:

- Thales (France, Norway): http://www.thales.com/;
- Norwegian Defence Research Establishment (Norway): http://www.ffi.no/;
- European Aeronautic Defence and Space Company (France): http://www.eads.com/;
- TNO (Netherlands): http://www.tno.nl/;
- Research Establishment for Applied Science (Germany): http://www.fgan.de; and
- Military Communication Institute (Poland): http://www.wil.waw.pl/.

The lead nations were France and Norway. During the Task Group work, the original topic, "Information Infrastructure Supporting Net-Centric Warfare Communications", was changed to "Secure Service Oriented Architectures (SOA) Supporting Network Enabled Capabilities". The rest of this chapter describes the background, the scope of work, and the achieved results.

1.2 SCOPE OF WORK

The actual scope of work is defined in the "Terms of Reference", published in Annex B of this report.

The original scope of work considered the functional and technical architecture of an "information communication infrastructure" integrating all the communication services necessary for systems conforming to the principles of Network Enabled Capabilities (NEC). The design was to be performed through identifying services at the **Network level and the Middleware level.**

Issues related with the network level considered, among other things, connectivity services (e.g., route selection), Quality of Service (QoS), network-level security (Virtual Private Networks (VPNs), firewalls), application communication services, etc.

The middleware level required the analysis of message oriented services, data distribution and object management services, data management services, Directory and Naming services and, finally, Security services.

However, it soon became obvious that handling both levels within a single Task Group was too ambitious. In fact, each of the levels would require a single, focused activity, and the interactions between middleware and network could likely require yet more effort. The Task Group decided to focus on the middleware level and skip the rest. Thus, the final project was a compromise but the results were still satisfying. The revised "Terms of Reference" resulting from this change of focus have been included in this report.

1.3 BACKGROUND OF SERVICE ORIENTED ARCHITECTURES

Current military C2, ISR, Combat Management systems are based on a federation of dedicated and heterogeneous systems, and because of that their operational integration deals with the following difficulties:

- Lack of operational interoperability;
- Weak integration of Information Systems services for Situation Awareness (from the strategic level to the tactical level); and
- Lack of global system management, prohibiting dynamic (re-)configuration of systems.

In the NATO NEC Feasibility Study (from NC3A), Service Oriented Architecture (SOA) was identified as a possible solution to the above-mentioned problems. Moving towards a Service Oriented Architecture (SOA) is a way to achieve the seamless information and service sharing required in a future NATO NEC.

SOA is a paradigm for organizing and utilizing distributed capabilities that enables the creation of applications that are built by combining loosely coupled and interoperable services to support the requirements of the business processes. In a SOA environment resources are made available as independent services that can be accessed without knowledge of their underlying platform implementation [19]. The key is independent services with defined interfaces that can be called to perform their tasks in a standard way, without the service having pre-knowledge of the calling application, and without the application having or needing knowledge of how the service actually performs its tasks. In this way, SOA supports the integration and consolidation activities within complex enterprise systems.

A SOA consist of Service Providers who offer their service(s) to Service Consumers by making a description of their services available for discovery, thereby making it possible to match the needs of service consumers with capabilities provided by service providers. Service Consumers can then find (discover) these services. After discovering suitable services, Service Consumers can start using the service in accordance to the specified service contract. A service provides access to one or more capabilities using a prescribed interface. The access is exercised consistent with constraints and policies as specified by the service description. The policies are described in a contract. Policies potentially apply to many aspects of SOA: security, privacy, manageability, Quality of Service and so on. Beyond such infrastructure-oriented policies, participants may also express business-oriented policies – such as hours of business, return policies and so on.

Services can be combined to form the desired application or system: A service consumer can use different services from different service providers and aggregate them into a new service offered to potential service consumers. SOA allows for services to be shared, but still to be under the control of a specific owner.

The Organisation for the Advancement of Structured Information Standards (OASIS) [3] is a not-for-profit, global consortium that drives the development, convergence and adoption of e-business standards. OASIS is developing the SOA reference model, which is currently the most practical alternative to be used as basis for deployments of SOA and the closest thing to a standard available.

A service-oriented architecture may be implemented using a wide range of interoperability standards, including Remote Procedure Calls, DCOM, ORB or Web Services (which were chosen in this project, see Section 3.1). Web Services are promising technologies for implementing SOA allowing for dynamic information sharing between military units. Among the many interesting features of Web Services are dynamic discovery of services and information exchange using event driven models like publish/subscribe. These services inter-operate based on a formal definition (or contract, e.g., WSDL, see 3.x) that is independent of the underlying platform and programming language. SOA is applied to implement (parts of) the middleware level as discussed in Section 1.2. COTS solutions to implement SOA currently exist. This implies the acceptance of SOA, making it a viable path to implement military solutions.

Some problems are not solved by SOA. Dynamic security solutions and bandwidth constraints in tactical networks are some great challenges that have to be solved in order to fully deploy a Service Oriented Architecture supporting NATO NEC.

1.4 TECHNICAL SPECIFICATION AND DEMONSTRATION

The Task Group cooperated to develop a technical specification of a secure SOA (Service Oriented Architecture) using Web Services, with dynamic discovery and replacement of services, effective information exchange through request/response and publish/subscribe communication patterns, and end-to-end security. The specification defines interfaces, protocols and functionality, which must be implemented to achieve interoperability between national systems that wish to share tactical pictures. A shortened version of the specification is contained in Annex A. The demonstration consisted of the application of before mentioned technologies to make these tactical pictures available and to access and share them. Additionally, to verify whether the specification is complete and "just works", the Task Group agreed to implement an experimental system and demonstrate it during the Coalition Warrior Interoperability Demonstration 2006 (CWID'2006) in Lillehammer, Norway.

1.5 SCOPE OF THE REPORT

This report contains the following chapters:

- Chapter 2, "Application of SOA to Support NEC", describes high-level ideas of how tactical communication in coalition environments using SOA, SOAP and XMLSEC can be achieved;
- Chapter 3, "Service Oriented Architectures Technologies Used", provides an overview of the technologies that are the building blocks for the project;
- Chapter 4, "The Demonstrator Scenario", describes the demonstrator scenario including the relevant communication events and processes;
- Chapter 5, "CWID'2006 Demonstrator and Realization", summarizes the specification, describes the actual setup at CWID'2006 and lists experiments that were conducted;
- Chapter 6, "Results and Lessons Learned", describes the results from the experiments and the lessons learned; and
- Finally, Annex A, "Demonstrator Specification", contains a shortened version of the demonstrator's specification.

Chapter 2 – APPLICATION OF SOA TO SUPPORT NEC

This chapter describes how SOA and the associated technology can be applied to NEC. NATO has adopted the principle of Network Enabled Capabilities (NEC) as a major guideline for future development work. Among the most important core functionalities within NEC is dynamic sharing of information. Nations must be willing to share information and the solutions must be available that makes dynamic information sharing possible.

Service Oriented Architectures (SOA) is a concept that enables resources to be provided and consumed as services. The services are published in a service registry, which announces their availability and indicates where they can be invoked. Web Services are promising technologies for implementing SOA allowing for dynamic information sharing between military units. Among the many interesting features of Web Services are dynamic discovery of services and information exchange using event driven models like publish/subscribe. Dynamic security solutions and bandwidth constraints in tactical networks are some great challenges that have to be solved in order to fully deploy a Service Oriented Architecture supporting NATO NEC.

2.1 NATO NEC FEASIBILITY STUDY

If we look at today's military solutions for information exchange, the information flow is typically defined by a set pattern from sender to recipient. It is pre-planned and preconfigured, and changes normally require manual assistance. The solutions used are often stove piped, tailored for certain applications within one military service and with no interoperability with other types of systems. Another problem is the limited exchange of information between security domains. If we look at the NATO NEC ambitions of seamless information exchange between units, there is certainly a requirement for more flexible and dynamic concepts of information sharing.

The NATO NEC Feasibility Study (NNEC FS) [4] is an 18 month study funded by 12 nations organized within the context of NC3B, and conducted by the NC3A. The aim of the study was to develop the scope and vision for NATO NEC and to establish the necessary context for developing the C3 aspects of NEC. The document focuses on a strategy and roadmap for identifying and proposing specific courses of action to deliver NEC CIS infrastructure capabilities including interoperability between NATO, NATO nations, partnership countries and potential coalition members. In the study NEC is defined as follows: "Network Enabled Capability (NEC) involves the seamless linking together of sensors, decision makers, and weapon systems, as well as multinational military, appropriately linked with governmental and non-governmental agencies in a collaborative, planning, assessment and execution environment. The NEC must provide for the timely exchange of secure information, utilizing communication networks which are seamlessly interconnected, interoperable and robust, and which will support the timely collection, fusion, analysis and sharing of information."

The NNEC FS has been well received in NATO and nations and the recommendations chapter of the document has been endorsed by the NATO C3 Board. This endorsement will have an impact on future work of the working groups under the NATO RTO and NATO C3Board in that their work will have to be in line with the recommendations of the NNEC Feasibility Study. One of the recommendations is "to utilize a component based 'Service Oriented Architecture' (SOA) approach in implementing the NII (Networking and Information Infrastructure)". The work of this RTG is therefore in line with the directions of the NATO C3 Board.

2.2 USE OF SOA TO SUPPORT NEC

In addition to the current concept of "need to know", NEC requires the new concept of "need to share". The aim of NEC is to increase mission effectiveness by networking military entities, enhancing the sharing of information and situation awareness. The key prerequisite of shared situation awareness is increased access to and sharing of information. The information infrastructure has an important role as an enabler, organizer and provider of information access and information sharing. By using SOA as a foundation for the Information Infrastructure, military resources may be made available as services that may be published and utilized over a communication infrastructure. The service itself is defined using a well-defined interface that exposes the functionality and hides the underlying implementation details. Services may be aggregated, by either the service provider or service consumer, to create more advanced services as further described at the beginning of Chapter 3 SERVICE ORIENTED ARCHITECTURE TECHNOLOGIES USED. This modularization makes dynamic reconfiguration of functionality easier. It opens up the possibilities of ad-hoc organizations of units without requiring much administrative overhead. The Services are characterized by metadata for easier discovery and published in the network.

Another aspect of SOA is the loose coupling of entities that allows for the dynamic and flexibility required in NEC. In a SOA, Consumers do not need to know in advance where the Service Providers are located (only where the Service Registry is). The Service Providers do not need to know in advance where the Consumers are located (only where the Service Registry is). And the Service Registry does not need to know in advance where neither the Consumers nor Providers are located. This makes ad-hoc organization of entities easier since the requirement for preplanning is minimal.

The concept of publish/subscribe as further explained in Chapter 3 SERVICE ORIENTED ARCHITECTURE TECHNOLOGIES USED has its strength in more efficient information exchange, avoiding information overflow. Simply speaking, publish/subscribe means that you will only receive the information that you have subscribed to. This concept utilizes a combination of push and pull. As opposed to a general "push" mechanism there are benefits in that you may select the information sent to you. Also, pure "pull" principles are not able to notify listeners when events occur.

Even if new and better technologies come along there will always be legacy systems in which nations have invested a lot of money and which have a defined calculated lifetime. These systems need to be integrated in a way that makes them an integrated part of SOA. Integrating systems in an SOA requires (amongst several things) the use of (or mapping to) a common data model and enabling the legacy systems to use SOA publishing and data exchange. As stated in the SOA reference model [5] "A SOA service brings together needs and capabilities, but the provider of the underlying capability may not be the same entity that provides the service which accesses that capability". As an example, when integrating a legacy system in the SOA, a bridge may be used to map the functionality of the legacy system the to the SOA technology used. The legacy system may be managed by different people than the people responsible for the SOA service management. Because administration of services can be done independently of the capability, the integration of functionality over heterogeneous technologies can be made easier.

The benefits of using SOA to support NEC may be summarized as follows:

- Military resources are made available as services over a communication network;
- Efficient discovery of and subscription to as well as downloading of relevant information;
- Faster deployment of new technology and functionality;
- Dynamic reconfiguration of functionality within a relatively short timeframe;
- Integration of functionality over different networks and heterogeneous technologies;
- Minimal pre-planning required – loose coupling of systems; and
- Migration – Legacy systems may be integrated with the new technology.

2.3 FLEXIBLE SECURITY MECHANISMS

The "need to share" concept of SOA requires changes not only related to technical solutions, but maybe even more at the organizational level. The concept of sharing information with others in such a flexible manner may be perceived as loosing control compared to the preplanned and preconfigured solutions. When the consumer in addition may be someone that you do not know in advance, this becomes a challenge. The willingness to share information requires that the provider can be sure that the information is protected all the way during transport and also when received by the consumer.

The increased information sharing in SOA may lead to increased vulnerability if security is not properly integrated. The situation of today is that separate networks protect information of different classification using physical, cryptographic and administrative separation. Introduction of security mechanisms which allow for dynamic and seamless exchange of information between units will be a challenge in NEC. IP level security will give confidentiality between systems, but will not prevent unauthorized access within the systems or LANs. Computer Network Attacks (CNA) will focus on attacks behind the firewalls (crypto devices) within the LANs/Systems. Therefore, end-to-end security services are required in order to secure the information inside the NEC systems and LANs, and to make sure the security is not broken in proxies or servers.

Security is a challenge with respect to NEC, making seamless sharing of information a bit more difficult. In our experiment we have focused on end-to-end security, which is also highlighted as the long term goal in the NATO NEC Feasibility Study. The use of end-to-end security solutions does not exclude additional use of traditional network and transport level security. To obtain end-to-end security we have been experimenting with XML and Web Services security solutions with the addition of XML Security Labels and a Public Key Infrastructure. Even though we have shown that the technology of today is improving, there is a way to go before trusted implementations are available that may be used to protect information of higher classifications like NATO CONFIDENTIAL or higher. More work needs to be conducted in this area.

2.4 DISADVANTAGED GRIDS CHALLENGES

Besides security, bandwidth restrictions at the tactical level are a limitation when it comes to fully deploying NEC principles. Those restrictions will in most cases remain, and as a means of reducing network traffic, the concept of publish/subscribe may be valuable for a consumer to be able to restrict the information being delivered. Other measures to reduce the use of bandwidth could be:

- More efficient protocol solutions (for example, the use of multicast);
- Compression;
- XML schema recoding;
- Data Model recoding; and
- Use of Proxies and cashing techniques.

This Task Group has only been evaluating some compression techniques, which are described in Chapter 8 of the Demonstrator Specification (see Annex A).

More work need to be conducted in this area.

Chapter 3 – SERVICE ORIENTED ARCHITECTURE TECHNOLOGIES USED

This chapter provides an overview of the SOA technologies used and evaluated by IST-061. Service Oriented Architecture (SOA) is, as explained in Section 1.3, a powerful but simple architectural principle inspired by the way business is performed. Simplified, a SOA consist of Service Providers who offer their service(s) to Service Consumers by publishing it in a registry. Service Consumers find (discover) these services by using the Service Registry. After discovering suitable services, Service Consumers can bind to the Service Producer, that is, start using the service in accordance to the specified service contract. A simplified overview of SOA is presented in Figure 3.1. The arrowed lines indicate communication between the entities.

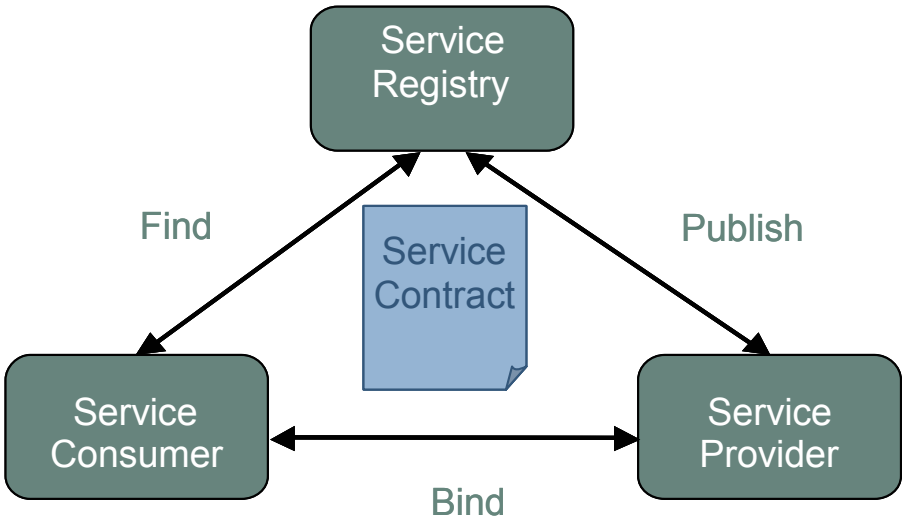

Figure 3.1: Service Oriented Architecture.

A SOA may best be defined as a collection of services that communicate with each other [1]. A service encapsulates standalone functionality which may be delivered across a network. It is well defined by a contract (see Section 3.2). Services can be combined to form the desired application or system: A service consumer can use different services from different service providers and aggregate them into a new service offered to potential service consumers.

It is important to remember that SOA is only an architectural principle, and is not tied to a given technology. A number of technologies such as CORBA, Java RMI and others can be used when implementing a SOA. However, the most popular approach to implementing SOA is the use of Web Services. This is also chosen for the work performed in this group. The most important Web Services technologies used is described in the following sections.

3.1 WEB SERVICES

Web Services are currently the preferred technology for implementing a Service Oriented Architecture and are quickly becoming the de-facto standard. There are many, often ambiguous, views on what a Web Service is. In this context we use the definition provided by the W3C: "*A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web*

service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards" [2].

Web Services are essentially a set of XML based standards used to implement a SOA. From these one can extract three basic specifications:

1) The WSDL specification, used for service descriptions and contracts;
2) SOAP used for transport of messages; and
3) The UDDI service registry specification.

These are explained in more detail in separate sub sections below. The most important standardisation organizations for Web Services standards include the *Organization for the Advancement of Structured Information Standards (OASIS)* [3] and the *World Wide Web Consortium (W3C)* [2]. The *Internet Engineering Task Force (IETF)* [9] is also maintaining some Web Services related standards.

It is important to note that the use of Web Services technologies does not automatically imply a SOA implementation; one can use Web Services technology and not adhere to the SOA principles. Web Services, although quickly becoming the de-facto standard, are not the only technology that can be used to implement a SOA.

3.2 WSDL

The Web Services Description Language (WSDL) is used to define the service contract for a Web Service. When describing services using WSDL it is possible to create a formal, machine-readable description which can be used to invoke the Web Service. The use of XML is the common denominator with all Web Services technologies, so it is also used by WSDL. A WSDL document is an XML document, and as such the Web Service description can be mapped to any implementation language, platform, object model or messaging system.

The description provided in a WSDL document includes definitions of messages and types. Type definitions in turn include definitions of port types, bindings and services. A WSDL definition consists of the following elements:

- The message element. It describes the XML payload to use. Message elements are described using XML schema types, either built-in or complex, which are either embedded in the WSDL document or included/imported from external sources.
- The port type element. This provides an abstract definition of the interface of the Web Service and the operations provided, including which messages to use.
- The binding element. This describes a particular mapping between the abstract port type element to a given protocol (e.g. SOAP over HTTP) and encoding style.
- The service element. It provides the final mapping from the abstract definition to a given address where the service can be reached, e.g. an URL.

The abstract definitions provided by a WSDL document can be extended with different port types and service definitions, thus providing the opportunity to describe the same service with different implementations.

During this work, WSDL v1.1 was used. This is the current version and is hence the most used. Within the W3C there is ongoing work on a WSDL v2.0 [18], but this initiative is still only a candidate recommendation.

3.3 SOAP

SOAP used to be the abbreviation for *Simple Object Access Protocol*. Today it is just the name of the messaging protocol that has become the de-facto standard used with Web Services, but has no specific meaning beside that. SOAP defines how to do service invocations in a standardised way, using XML.

A SOAP message consists of the three basic XML structures Envelope, Header and Body. The Envelope wraps the optional Header and required Body element. The Body element is used to convey the actual XML data (that is the payload) to be transported. The SOAP Header is optional, but is an extensible mechanism used to convey additional information that is not application payload; this can for example be control information and other application specific data.

SOAP messages can be delivered over a number of application, transport and network protocols. The most commonly used binding is SOAP over HTTP, which was also used during the work described in this document. The SOAP over HTTP binding describes how HTTP can be used to transport SOAP messages between hosts. Other possible protocols that can be used to transport SOAP messages include bindings to Simple Mail Transfer Protocol (SMTP), File Transfer Protocol (FTP), Java Messaging Service (JMS), Internet Inter-ORB Protocol (IIOP) and other proprietary protocols.

SOAP is available in both versions 1.1 and 1.2. Even though the SOAP version 1.2 is available and recognised as a W3C Recommendation (also known as standard) version 1.1 is still the most widely used one. During the work described in this document both versions were used.

3.4 SERVICE REGISTRY – UDDI

An important part of an SOA is the Service Registry. When using Web Services the most common standard chosen is the *Universal Description Discovery and Integration (UDDI)* from OASIS [3]. The UDDI provides mechanisms for publishing and discovery of services, primarily implemented using Web Services. Services implemented using other technologies can also be published, even though this is not usual. UDDI can be used for both design time and run time discovery of services. The most common pattern of use is however design time discovery. Design time discovery is utilised by client-software developers when designing and implementing client software. By run time discovery we mean service discovery performed during execution of a system. When using UDDI, run time discovery often involves finding running instances of a service with a known technical fingerprint, and not discovery of unknown types of services.

The data model provided by UDDI also provides means to describe and register service providers such as businesses and organisational units, as well as relationships among them. A description of organizational entities is named businessEntities in the UDDI data model (see Figure 3.2). This entity is one of the four core entities of UDDI, and contains zero or more businessService entities. The businessService entity is a high level description of services. It is used to outline the purpose of the services, and include metadata used for service discovery. The businessService entity is a non-technical description of the services and contains zero or more bindingTemplate entities which describe the technical implementation of the service. The information contained in the bindingTemplate is used by a client to bind to and interact with the given service. The bindingTemplate is basically a collection of references to tModels, also known as Technical Models. tModels are the final core entity and are used within UDDI to represent unique concepts or constructs like specifications, transports and protocols. The businessService entity together with the referenced tModels forms a technical fingerprint of the service. Nevertheless, tModels are not confined to describing technical fingerprints. They can also be used to describe other concepts like categorization schemes for businesses and services, identifier schemes and other concepts.

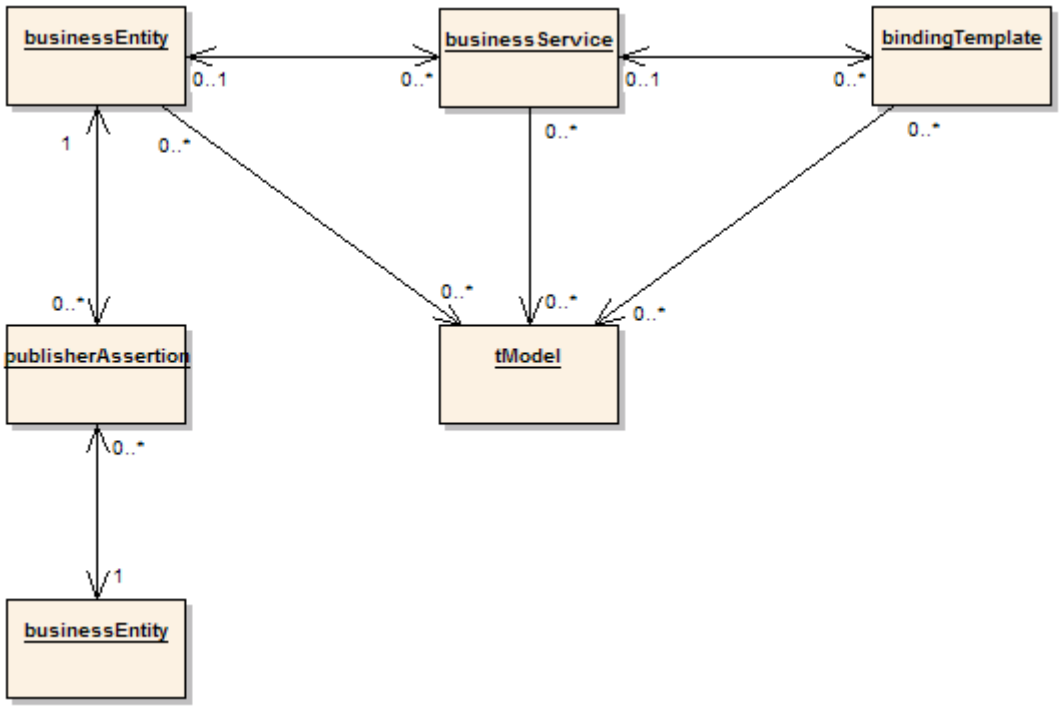

Figure 3.2: UDDI Data Model.

It is important to note that UDDI is itself implemented as a Web Service with advanced interfaces and using SOAP for message exchange. This includes APIs for publishing, inquiry, subscription, security policy, custody and ownership transfer as well as the value set API which is optional and used to allow external validation of data. The UDDI also describes an API used for replication of service registry content. Even though UDDI essentially started out as a centralised service registry, newer versions of the standard include the ability to create federations of registries and integration of registries. The ability to do replication allows for improved scalability and to a certain extent better protection against single-point of failure problems.

Alternatives to UDDI include ebXML Registry [11], which is similar to the UDDI specification. The use of more or less centralised registries is by far not the only way to perform service discovery. Decentralised alternatives like the WS-Discovery [15] specification are also available. We chose to rely on the UDDI standard for this work due to the fact that UDDI is a de-facto standard and it has got strong vendor support.

UDDI has evolved to version 3.0, which was ratified as an OASIS standard in February 2005. Even though few implementations of this specification are available yet, we chose to rely on this version for the work described herein.

3.5 PUBLISH/SUBSCRIBE – WS NOTIFICATION

Publish/Subscribe, often abbreviated to pub/sub, is a well known communication pattern for event-driven, asynchronous communication. Publish/Subscribe makes it possible to link together publishers (data producers) and subscribers (data consumers) into loosely coupled, scalable and dynamic networks. As shown in Figure 3.3, the publish/subscribe pattern fits well into the SOA and Web Services architecture. When fitting pub/sub into SOA, a service provider is equal to a publisher, and a service consumer is equal to a subscriber.

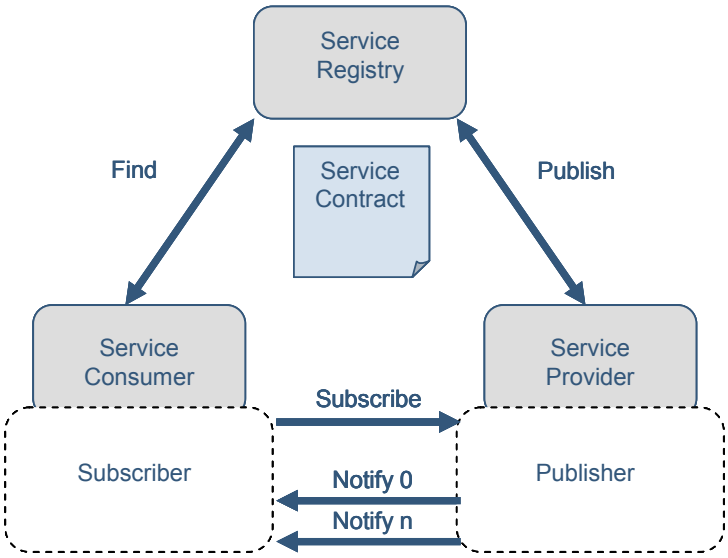

Figure 3.3: Publish/Subscribe Basic Elements in a SOA.

To implement the Publish/Subscribe pattern different protocols and standards can be used. We chose to rely on the specifications provided by OASIS, denoted WS-Notification [13]. This is actually a collection of the three specifications WS-BaseNotification, WS-BrokeredNotification and WS-Topics. For the work presented in this document we utilised the WS-BaseNotification and the WS-Topics specifications. Using WS-Notification terminology, a service that publishes data (publisher) at a specified Topic is called a *NotificationProducer*. Topics are a way to group together, represent and categorize items of interest. The data format of each topic is defined by an XML schema. A client, called a *NotificationConsumer* (subscriber), first creates a subscription to the service. The client will subsequently receive notifications as they are produced by the NotificationProducer. Since WS-Notification is a Web Services specification, all messages are exchanged using SOAP.

Version 1.3 of the WS-Notification, WS-BrokeredNotification and WS-Topics specifications has been recognised as OASIS standards in October 2006. However, during the work described in this document we relied heavily on the use of version 1.2 of these specifications, because no implementations where available at the time.

Other alternatives to WS-Notification have also been proposed, most notably the WS-Eventing specification [14]. This specification is the product of an effort by an industry initiative, and has the status of a “Member Submission” to the W3C as of March 2006. This specification has not been subject to evaluation from this group.

3.6 XML SECURITY

When using Web Services, all the specifications are based on XML and the use of SOAP messages. Therefore XML security specifications can be used to secure the different Web Services components. Many specifications have been written for securing XML documents and some of them have become standards. Most notable of these are the XML Digital Signature (XML Dsig) which has both been ratified as a W3C Recommendation [17] and published as an IETF RFC [6], and XML Encryption (XML Enc), which has been published as a W3C Recommendation [16].

The OASIS WS-Security standard specifies how to extend the SOAP message header in order to achieve message integrity, confidentiality, authentication of originator and replay protection. It is based on the use

of the security standards XML Digital Signature and XML Encryption and supports a variety of security token formats (e.g. X.509, SAML [12] and XrML [7]).

A security label is often used in association with information to provide an indication of the security policy, sensitivity, compartments, and other handling caveats. An official XML-based standard for security labeling of information objects does not yet exist. On the other hand, security label specifications have earlier been developed for X.400 messaging (X.411) and SMTP (IETF S/MIME ESS [8]). These have been used as a basis for the development of the XML Security Label specification used in the demonstrator.

In addition to the use of XML Security specifications and standards to secure the SOAP messages exchanged, we used a security infrastructure to provide and distribute security tokens, including user privileges. We used a PKI and an LDAP Directory to provide the security infrastructure for exchange of certificates and certificate revocation lists. LDAP synchronization was performed using WS-Notification.

3.7 MIP/C2IEDM

The data model is not a core SOA technology as such, but it is included here because data exchange on an interoperability level is important. For this, we chose to reuse the data model defined by the Multilateral Interoperability Programme (MIP) [10]. This is an effort towards providing a common understanding of the battle space between different countries. It is independent of doctrines, procedures and tactics. The MIP model has been developed over many years of work. It started as a land model and is currently being extended to cover joint environments. The aim of the MIP is to achieve international interoperability of Command and Control Information Systems (C2IS) at all levels, in order to support multinational operations.

We chose to use the C2 Information Exchange Data Model (C2IEDM) from MIP Baseline 2. This data model is primarily intended to be used for database replication using the MIP Data Exchange Mechanism (MIP DEM). Instead of using database replication as defined in DEM, we are using Web Services as the information exchange mechanism, using the object-oriented XML representation of the C2IEDM. The main reason for this choice is the fact that XML is well aligned with Web Services.

The C2IEDM is a large and complex data model with approximately 240 entities. As a consequence of this we defined a subset that we used for this work. This subset consists of 30 “core” entities which covers our need for a data model. It should be emphasized that the model in it self was not changed but only reduced, so the data model is still valid.

Chapter 4 – THE DEMONSTRATOR SCENARIO

The demonstration has gathered five nations featuring six platforms in Jørstadmoen Camp in Lillehammer (Norway) during CWID 2006. The names of the participating companies or organisations as well as the demonstration name and Identification number as they were referenced by CWID authorities are listed hereafter:

- EADS for France: EADS demonstrator of the French demonstration Secured Architecture for Information Sharing (SAIS) ID#49;
- FFI for Norway: SecSOA demonstration Secure SOA supporting NEC ID#69;
- NC3A for NATO: IEG Functional Services demonstration #ID104;
- Safelayer for Spain: Advanced Trusted Information Interoperability demonstration ID#106;
- Thales for France: SecSOA of the French demonstration SAIS ID#49; and
- MCI for Poland: Polish SOA WS system demonstration ID#103.

These demonstrators were each located in a separate room according the nationality of company or organisation in the CWID building and were connecting to the CTF-NRF network. This network used the infrastructure of the Combined Federated Battle Laboratories (CFBL) Network (CFBLNet). This network is classified Mission Secret and is also known as the purple Network. (See Figure 4.1)

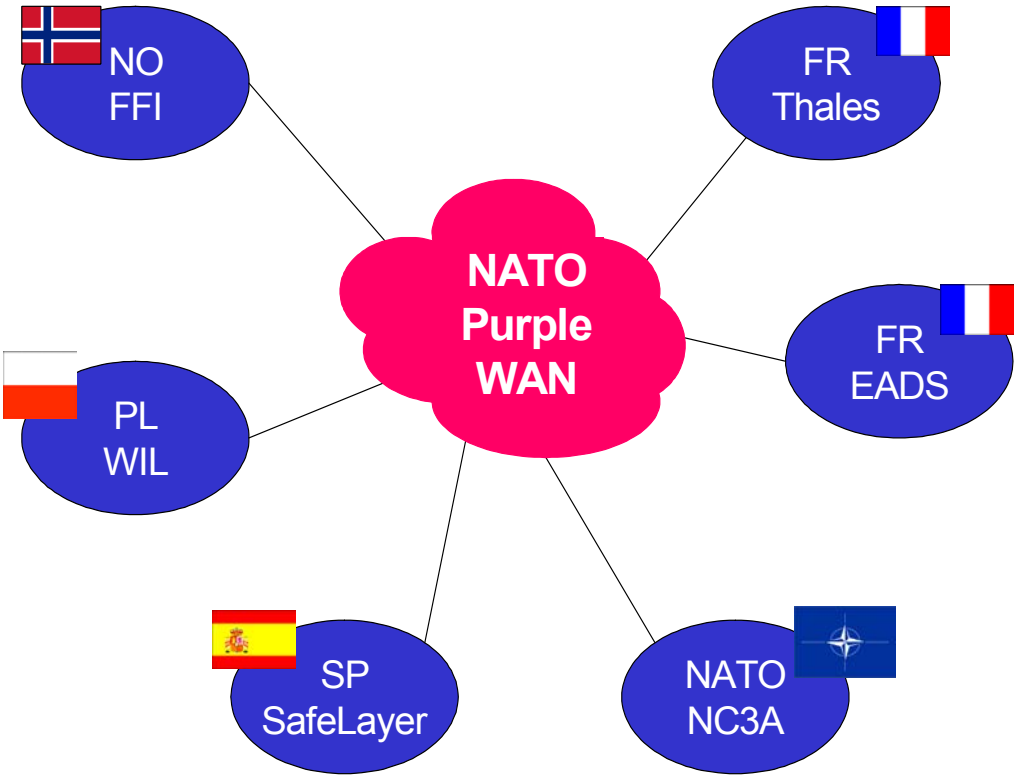

Figure 4.1: Demonstration Participants.

The purpose of the demonstration is to show how nations can share information and capabilities concerning two Communities Of Interest (COI):

- The C2 COI: Nations expose Operational Picture services through which they provide pictures. Picture services are invoked on demand to retrieve pictures from other nations; and
- The ISR COI: Nations expose ISR services through which they can accept Sensor requests to be executed by their Sensor systems. Processed information from Sensor systems can be shared between platforms.

The area chosen for the demonstration scenario is shown in Figure 4.2. It is located in the south of UK near Southampton and called "Greenport".

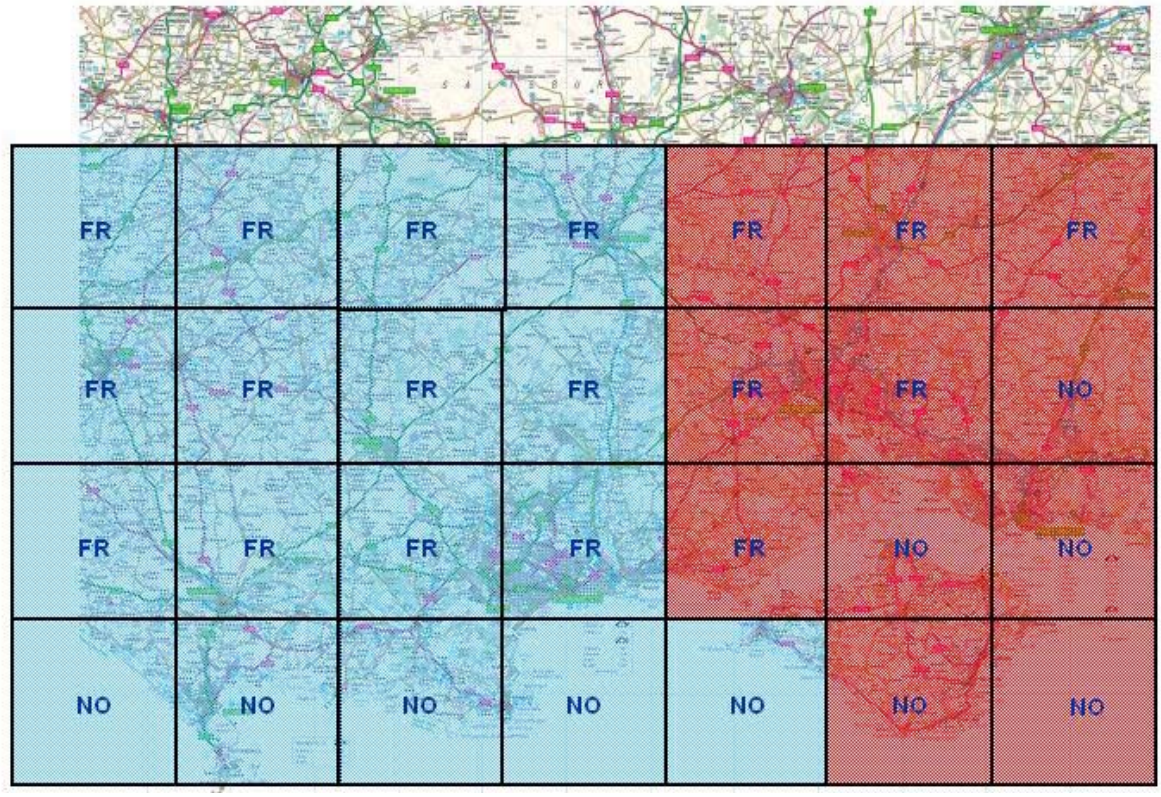

Figure 4.2: Demonstration Area.

The area is segmented in sub-areas that are each under the responsibility of a single nation. The responsible nation positions Blue Forces (in blue squares) and locates Opponent Forces (in red squares) in their sub-areas. Only the responsible Nation is authorised to put elements in its squares. This method has the advantage of not needing a "God's eyes view" for the simulation. It also precludes the need to synchronize the scenario between nations. A limitation of this method is that it is not possible to perform a fusion of data from different nations, but this was not the focus of the demonstration. (See Figure 4.2)

The interoperability story (the scenario) between the nations could be summarised as follows:

- A Land force (FR) is conducting operations in a coastal area performing detection, location, and identification of opponent forces:
  - C2 COI: LCC provides a Land picture to other forces; and
  - ISR COI: The land ISR assets contribute to the detection of opponent forces. Support is requested from Naval ISR assets.

- A Naval force (NO) is performing littoral surveillance:
  - C2 COI: MCC provides Maritime picture to other forces; and
  - ISR COI: A Naval UAV system provides detection of Land forces. Its base station personnel receives sensor requests (service invocation), processes them and decides on changes in the UAV sensor tasking and/or flight plan.

Figure 4.3 is an example on the dialog between the French platform and the Norwegian platform.

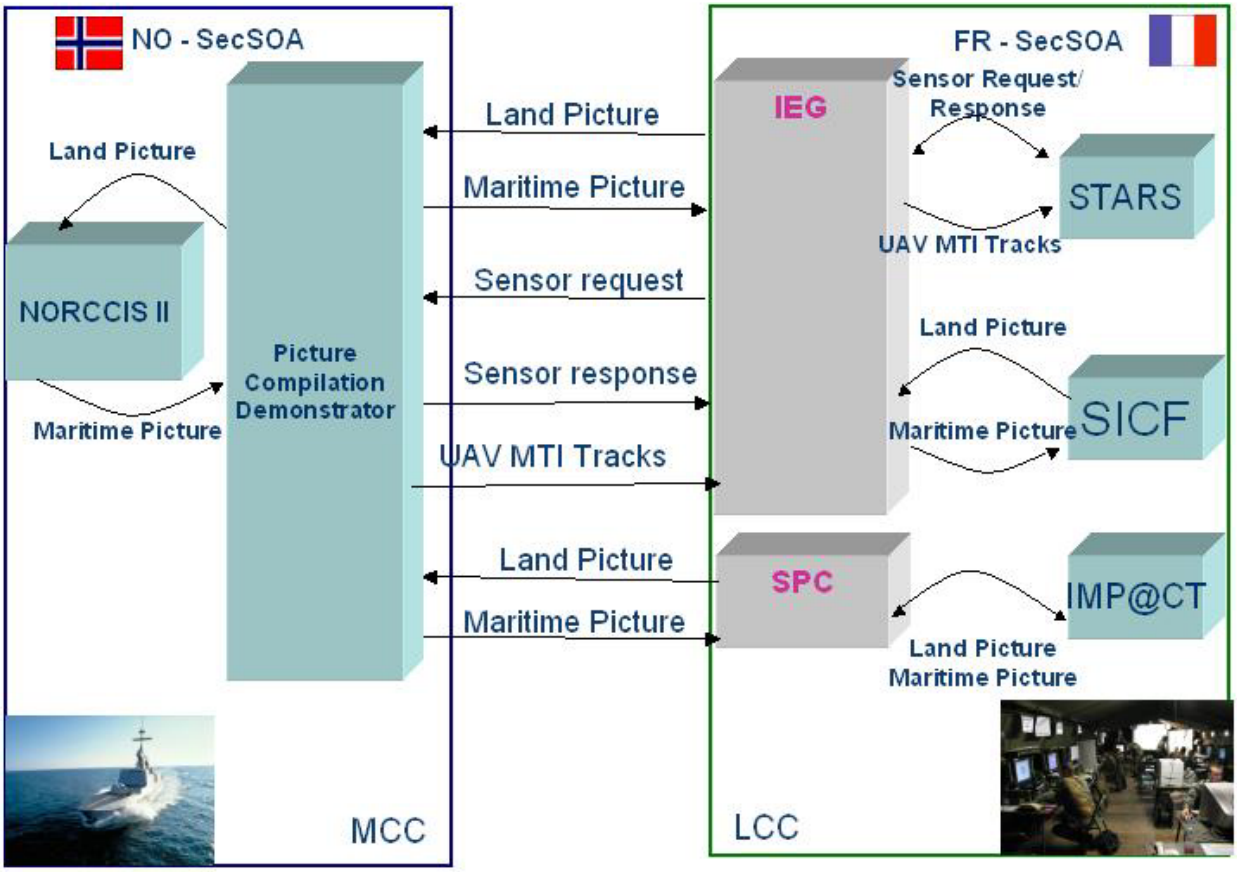

Figure 4.3: Exchange between Norway and France.

Table 4.1 below shows the participants' intent as providers or consumers of services.

Table 4.1: Services Providers and Consumers

Services	France Thales	France EADS	Norway FFI	Poland MCI	Spain Safelayer	NATO NC3A
Land Picture	**P**	**P**	**C**	**C**	**C**	**C**
Maritime Picture	**C**	**C**	**P**	**C**	**C**	**C**
Sensor Request	**C**	**C**	**P**	**C**	**C**	**C**
UAV MTI Tracks	**C**	**C**	**P**	**C**	**C**	**C**

P: Provider **C**: Consumer

Prior Interoperability, Platforms have to connect each other according a three-step process in order to get capabilities of browsing the Service Registry and then Service Invocation:

1) **Planning:** Exchange of necessary Security Certificates for directory replication and exchange of addresses of Platform Gateways through trusted files;
2) **Assembling:** Platforms replicate Directories of Security Certificates, hence all necessary Certificates, for enabling Service invocation through the Network, have been retrieved; and
3) **In operation:**
   - Platforms can publish services that they are willing to expose in the NATO Services Registry. This Registry is hosted on the Norwegian Platform; and
   - Platforms can browse the NATO Services Registry and subscribe to selected services. They will receive updates of information from the services to which they have subscribed.

In the end, the platforms could provide and consume the services as depicted in Table 4.1 Service Providers and Consumers above.

Summarily, the main demonstrated technical components are:

- **Dynamic Service Discovery:** The services are accessible on the Network, platforms do not need prerequisite knowledge of their location except the Services Registry location.
- **Publish-Subscribe Service:** This method allows a service consumer to subscribe to a delivery information service that has been published in the service registry. In the demonstration the Land and Maritime pictures were distributed following this method.
- **Request-response Service:** This method allows a platform to request a service and then to receive the response. The Sensor Request Service uses this method.
- **Services Registry:** A central Services Registry is used for sharing information about services and their publishers.
- **Security Certificates Directories Replication:** Platforms can be synchronised and Certificates can be exchanged.
- **Secured Exchanges:** All exchanges between platforms are performed in a secure way using signature, labeling and ciphering.
- **Security Certificates Revocation:** A certificate can be revoked “on the fly”.
- **Data Exchange Format:** All exchanged Messages are XML-based. The Land and Maritime pictures abide by a sub-set of the C2IEDM-XML Object-oriented Data Model. The UAV MTI Tracks are specified in a proprietary XML schema as there is no standard for sensor data.

Chapter 5 – CWID 2006 DEMONSTRATOR AND REALIZATION

The previous chapter described the general scenario of the demonstrator created by the group. Based on this description, the current chapter introduces its realization as it has been deployed at CWID 2006. The abstract view of the architecture consists of a set of software components for each nation. This set of software components supports secured Web Services exchanges in both publish-subscribe and request-response modes.

The high level architecture proposed by the working group was implemented in CWID as it is described in Figure 5.1.

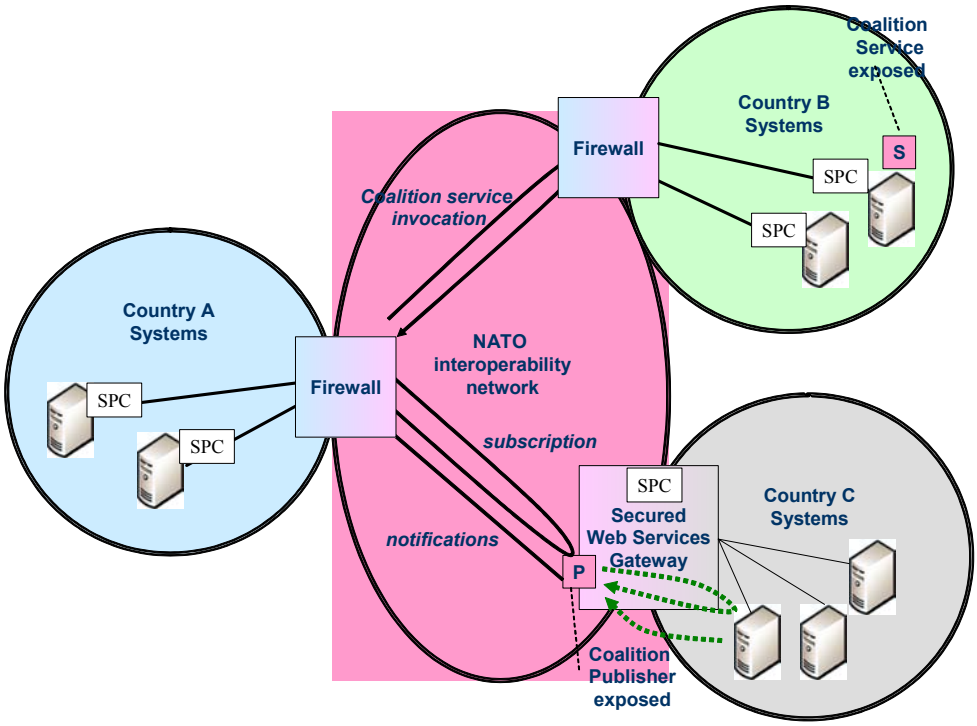

Figure 5.1: Demonstrator General Overview.

Each nation was connected to the CWID purple WAN. Norway and France were service providers and consumers. Poland, Spain and NC3A were service consumers as it was mentioned in the previous chapter.

5.1 GLOBAL ARCHITECTURE

The global architecture consists of the following "building blocks", each of which will be described in the following section. They are:

- Service registry;
- LDAP;
- SPC; and
- Publish / Subscribe mechanism.

5.1.1 Service Registry

The service registry is based on UDDI (Universal Description, Discovery and Integration).

During the trials, each country had the ability to discover new services thanks to the service registry. The service registry was itself a service provided by Norway during CWID.

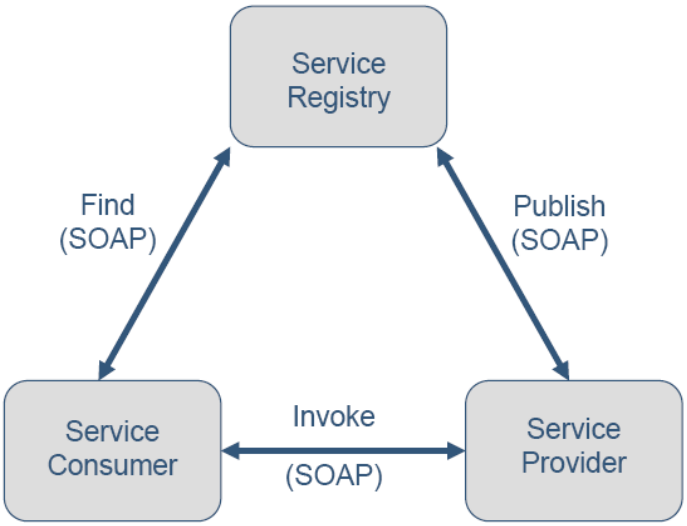

Figure 5.2: Service Registry.

UDDI V3 was chosen for the trials. During CWID, Systinet UDDI V3 COTS was used.

5.1.2 LDAP

The LDAP (Lightweight Directory Access Protocol) directory allows PKI exchange between Nations. The directory relies on the OpenLDAP software.

LDAP is synchronized directly between different nations. The directory synchronization tool was provided by Thales France.

The LDAP in each national demonstrator is directly used by the Security Protection Component (SPC) for authentication as shown in Figure 5.3.

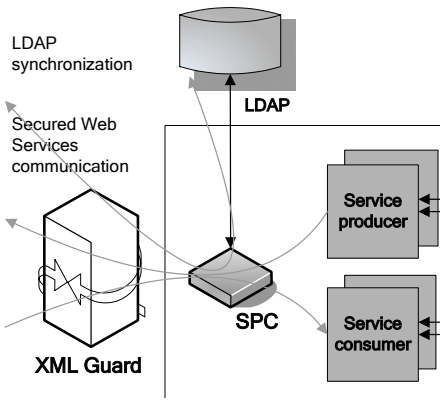

Figure 5.3: LDAP Interaction with SPC.

5.1.3 Security Functionality

The security functionality is placed in the end systems or in a Secured Web Service Gateway.

In Figure 5.4 below, you have a representation of different types of possible architectures. For example, Nation 1 has an SPC for each Service visible by other nations while Nation 2 has a unique Secured Web Services gateway integrating security. The implementation of security is specific for each nation and will be described later in this chapter.

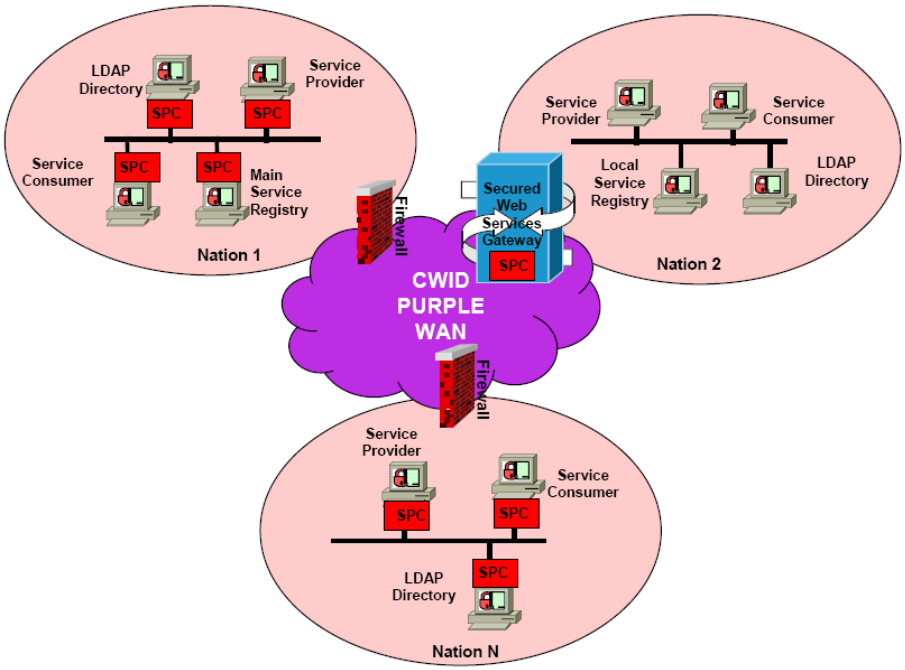

Figure 5.4: Security in the Global Architecture.

5.1.4 Publish / Subscribe Mechanism

The publish / subscribe mechanism used in the demonstrators is based on the Ws-Notification standard.

During the trials, each country had the responsibility to implement Ws-Notification. It will be described more precisely later in the chapter for each country (particularly for the service providers). Figure 5.5 below describes the general architecture for the use of publish/subscribe mechanism linked to the use of a service registry.

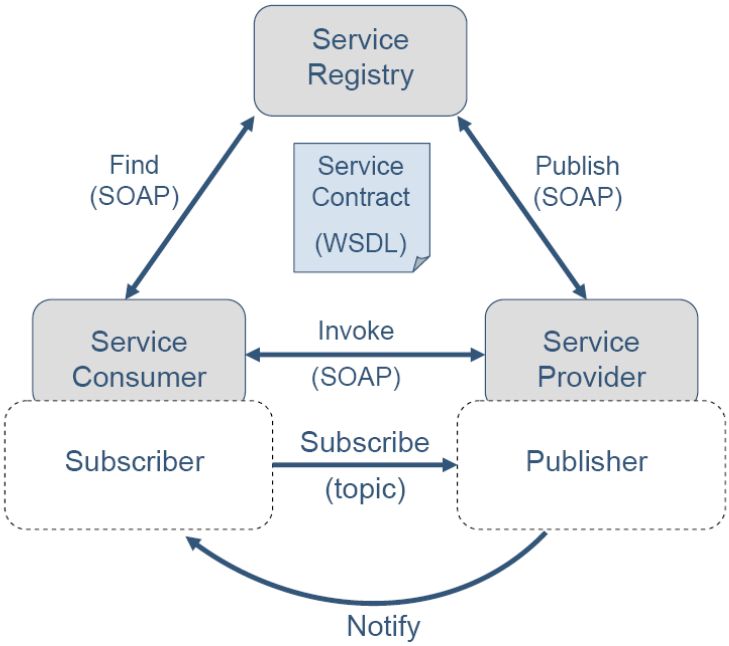

Figure 5.5: Publish/Subscribe Mechanism.

5.2 PARTICIPANTS' DEMONSTRATORS

This section describes the individual demonstrators from the participating nations.

5.2.1 EADS (France)

The EADS part of the French demonstrator represents a platform as described in Figure 5.6.

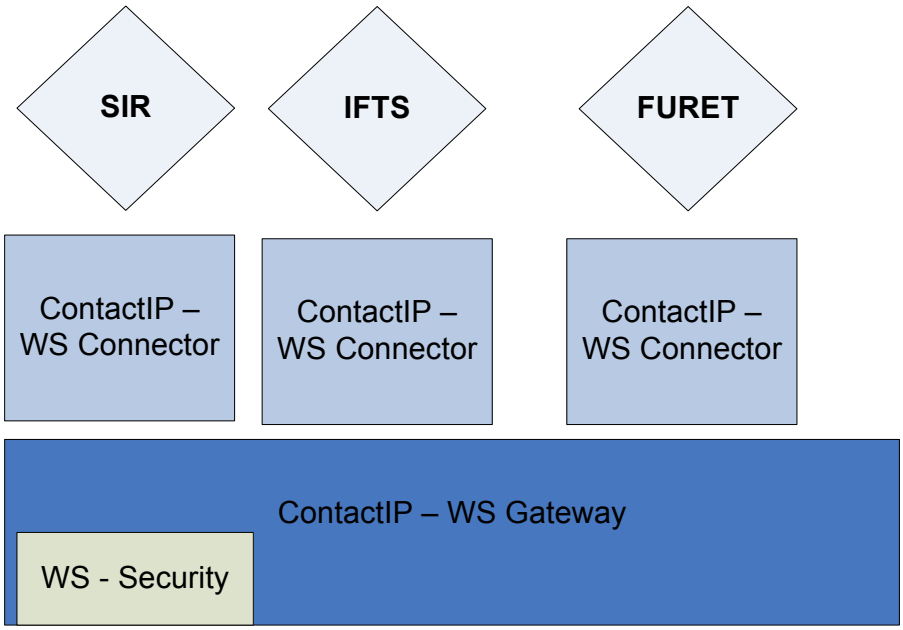

Figure 5.6: Demonstrator Architecture.

This architecture permits to include legacy systems such as:

- IFTS: Interim Force Tracking System deployed for NATO in Kosovo,
- SIR: "Système d'Information Régimentaire", the French regiment level system, and
- FURET: French production chain of tactical intelligence,

within the SOA platform.

This platform is composed of:

- "Contact IP – WS Connector" that allows implementing specific features to wrap existing systems to these new architectures needs.
- "Contact IP – WS Gateway" which is the main block of this architecture. It allows request/ response, publish/subscribe (WS-Notification) mechanisms.
- "WS Security". This block wasn't developed by EADS. As all the Web Services developed rely on WS standards such as WS-Notification and as EADS had decided to put more effort on web service implementation than on security, the demonstrator used the SPC developed by Norway.

5.2.1.1 Demonstrator Architecture

For the demonstration purpose, we chose to use IFTS' Imp@ct information system to test the whole ContactIP-WS architecture at CWID.

The demonstrator was composed of

- IMP@CT:
  - To visualize MIP situation provided by any nation, and
  - To visualize MTI tracks provided by any nation.
- Simulator:
  - To simulate movement of units that will be provided as changing MIP situation.
- "ContactIP – WS Gateway" based for this demonstration on Globus Toolkit:
  - That allows WS-Notification, and
  - With an additional development provided by Norway for WS-Security.
- "Contact IP – WS Connector:
  - That provides MIP tactical situation,
  - That is client to MIP situation, and
  - That is client to MTI tracks.

Figure 5.7 provides a very simple example of how the different elements of our demonstrator interact with other national systems:

- To provide a MIP situation to all authorized nations; and
- To get MIP situation from Norway and Poland.

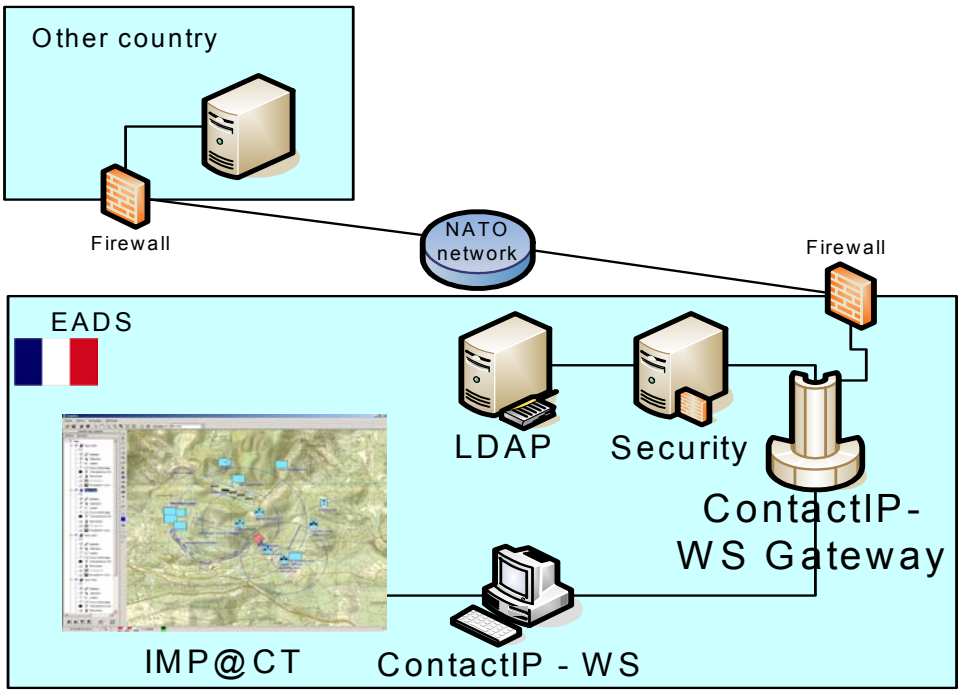

Figure 5.7: Demonstrator Interoperability Example.

5.2.1.2 Demonstrator Implementation

The demonstrator was created in an incremental development:

- Adaptation of the ContactIP framework: creation of the WS gateway;
- Implementation of a connector for legacy systems in conformance with standards to allow use of these web services in any infrastructure;
- Development of the security component; and
- Implementation of subscribing/unsubscribing for the UDDI registry.

As the time for development became shorter, we finally only made the 2 first steps of the demonstrator before CWID.

5.2.1.3 Interoperability before and during CWID

Before CWID, EADS provided a C2 Web Service to all other nations on Internet. That was an easy way for EADS to test all the ContactIP-WS Connectors created, and for other countries such as Poland or NC3A to test their implementations before moving to Lillehammer.

During CWID, EADS provided a C2 Web Service to all participants. Complete secured interoperability between IMP@CT-C2 Web Service and Polish and Norway systems were tested with success. The situation provided by IFTS-Imp@ct was visible on NORCCIS and on the Polish C2 Demonstrator.

CWID was also the opportunity for EADS to test its IMP@CT-C2 client by connecting to NORCCIS and to test its IMP@CT-MTI Client by connecting to the Norway Compilation Demonstrator which are described later in this chapter. The complete secured interoperability between Norway and the EADS clients were tested with success. The situation was viewable on the 2 EADS systems.

5.2.2 Thales (France)

The Thales part of the French demonstrator represents a Platform composed of an IEG Gateway and an Application Infrastructure. The Gateway is the public part of the architecture whereas the Application Infrastructure is hidden. (See Figure 5.8)

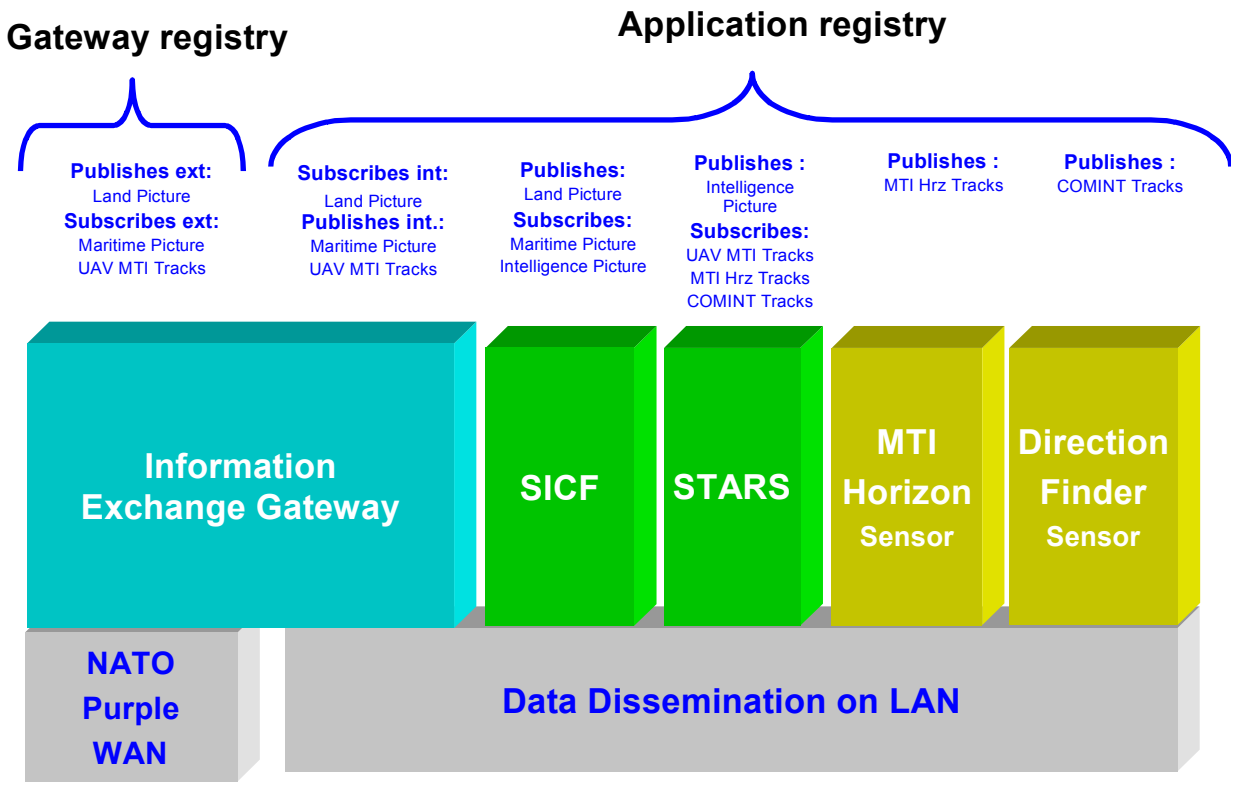

Figure 5.8: Thales Part of French Demonstrator.

The Main Components of the demonstrator are:

- An Information Exchange Gateway: This is a bridging component between the external world and the internal part of the Demonstrator.
- An Application Infrastructure comprising:
  - SICF: "Système d'Information et de Commandement des Forces" Command and Control Information System to display both land and maritime pictures and to generate the land picture;
  - STARS: "STructure D'Accueil pour le Renseignement et la Surveillance" Intelligence and Surveillance Framework for Tracks handling and fusion from different sensors;
  - MTI Horizon Sensor: Simulation of heliborne MTI radar tracks;
  - Direction Finder Sensor: Simulation of COMINT tracks; and
  - Data Dissemination Layer: Communication node distributing data within the LAN.

The demonstrator includes legacy systems such as SICF, STARS and Sensors. These legacy Systems have been integrated in the SOA architecture although they were not designed to be deployed in this way.

5.2.2.1 IEG

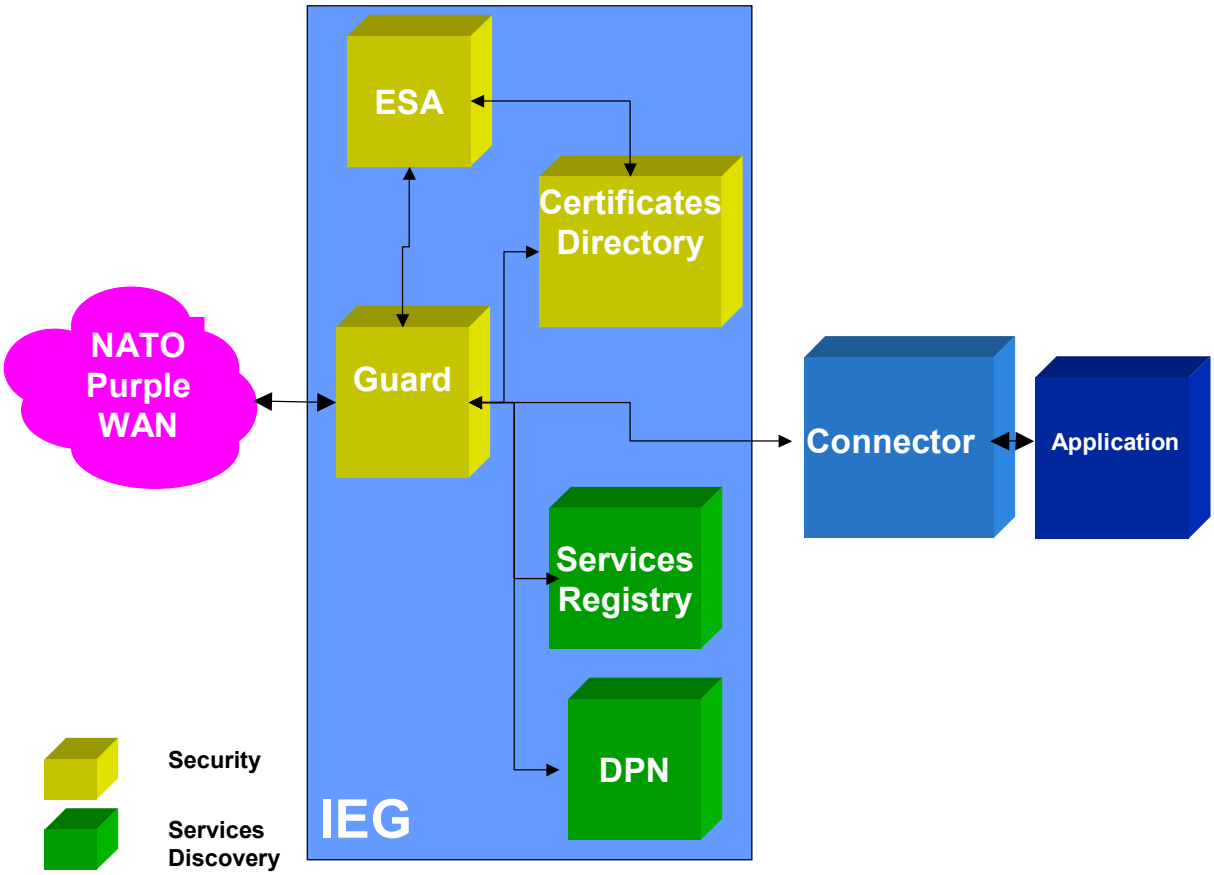

Figure 5.9: French (Thales) IEG.

The Gateway comprises services for information exchange with other members of the coalition according to two patterns: "Publish/Subscribe" and "Request/Response". Technical services in the IEG are:

- Information Assurance Services:
  - The Guard checks all outgoing information according to a Label with an appropriate security level and a signature. Allowed outgoing messages are transmitted. The integrity of incoming messages is verified;
  - The Encryption and Signature Authority (ESA) encrypts and signs outbound messages, and decrypts inbound messages coming from the Guard; and
  - The directory of Security Certificates provides Synchronisation Capability between Platforms.
- Discovery Services:
  - The registry of services that are exposed to the NATO Coalition; and
  - The Data Publishing Node (DPN) supporting the publish-subscribe mechanism for the Coalition Network.

The Connector bridges the application infrastructure with the Gateway.

5.2.2.2 Application Infrastructure

Behind the IEG, the Application Infrastructure relies on a Data Dissemination Layer where operational systems can provide and consume Data.

A user of the Data Dissemination layer can affiliate itself as a producer or consumer of a list of flows. Several of these flows are aggregated into dissemination channels. Each channel and each flow within a channel has a unique name. Here are some examples for possible channels with their respective flows from the present demonstration:

- Dissemination channel: C2 Channel:
  - Information flow: Land Picture; and
  - Information flow: Maritime Picture.
- Dissemination channel: ISR Channel:
  - Information flow: MTI Horizon Tracks;
  - Information flow: COMINT Tracks; and
  - Information flow: UAV MTI Tracks.

The connector within the IEG is a user of the Data Dissemination Layer. Hence it can transfer information between the internal part of the demonstrator and the external NATO Network. The connector is a subscriber of the national services (internally visible and exposed in the internal registry but hidden to the external world) through the Data Dissemination Layer. The national services are in turn published on the NATO registry as NATO services (externally visible and exposed by the Gateway Registry and/or NATO registry, but hidden to the internal part). In opposite operation, the connector is a publisher on the NATO registry of services that it has previously subscribed to on the Data Dissemination Layer.

5.2.2.3 Demonstrator Capabilities

The French Thales Demonstrator has mainly interoperated with the Norwegian demonstrator. From an Interoperability point of view, the demonstrated capabilities have been:

- Browsing and Discovery of Services in NATO Registry;
- Providing the Land Picture provision service. Publishing the existence of the Land Picture service provider in the NATO Registry;
- Subscribe to the Maritime Picture provider via address received from the NATO Registry;
- Send a Sensor Request to the Norwegian Demonstrator;
- Subscribe to the UAV MTI Tracks provider using address received from the NATO Registry;
- Exchange XML-based messages using the security mechanisms;
- Conversion of XML Messages between MPIA format (Modèle Pivot Inter-Armées: Standard of French Armies) and C2IEDM XML format; and
- Security Certificates Directories Synchronisation.

5.2.2.4 Demonstrator Implementation

The demonstrator has required the development of several components. To reduce development overhead, several COTS components (commercial and Open Source) have been included. If required, a wrapping to

adapt them to the specifications of the demonstrator was created. Table 5.1 below lists some of these components with their wrappers.

Table 5.1: Components of the Demonstrator

Components	COTS Inside	Standard
Guard	XML-Security	WS-Security
ESA	XML-Security Xerces	WS-Security
Certificates Directory	OpenLDAP	LDAP
Services Registry	Systinet Registry	UDDI
DPN	Axis TOMCAT	WS-Notification SOAP
Connector	Axis TOMCAT	SOAP
Data Dissemination	–	Proprietary

The French Thales Demonstrator has a different implementation from the other demonstrators. For the interoperability, the other demonstrators rely mainly on a Norwegian component, namely the Security Protection Component (SPC), which has been developed by Thales Norway. The SPC uses a COTS component, namely the Globus Toolkit 4 (GT4), for the Interoperability based on Web Services. The SPC component is used by all participants (except by the FR Thales demonstrator). As a consequence of the use of the Globus toolkit, there was a Synchronous behaviour on the NATO Network due to the fact that the GT4 acknowledges each terminating transaction.

The intent of the French Thales demonstrator was to demonstrate a platform operating on a battlefield using a radio Network. This intent requires the use of an asynchronous Network. To reduce the effort required to integrate with the other partners, it was decided to keep the NATO Network synchronous and to wrap the Thales Demonstrator behind a component that bridges the synchronous and asynchronous behaviours. The structure of the resulting network is shown in Figure 5.10.

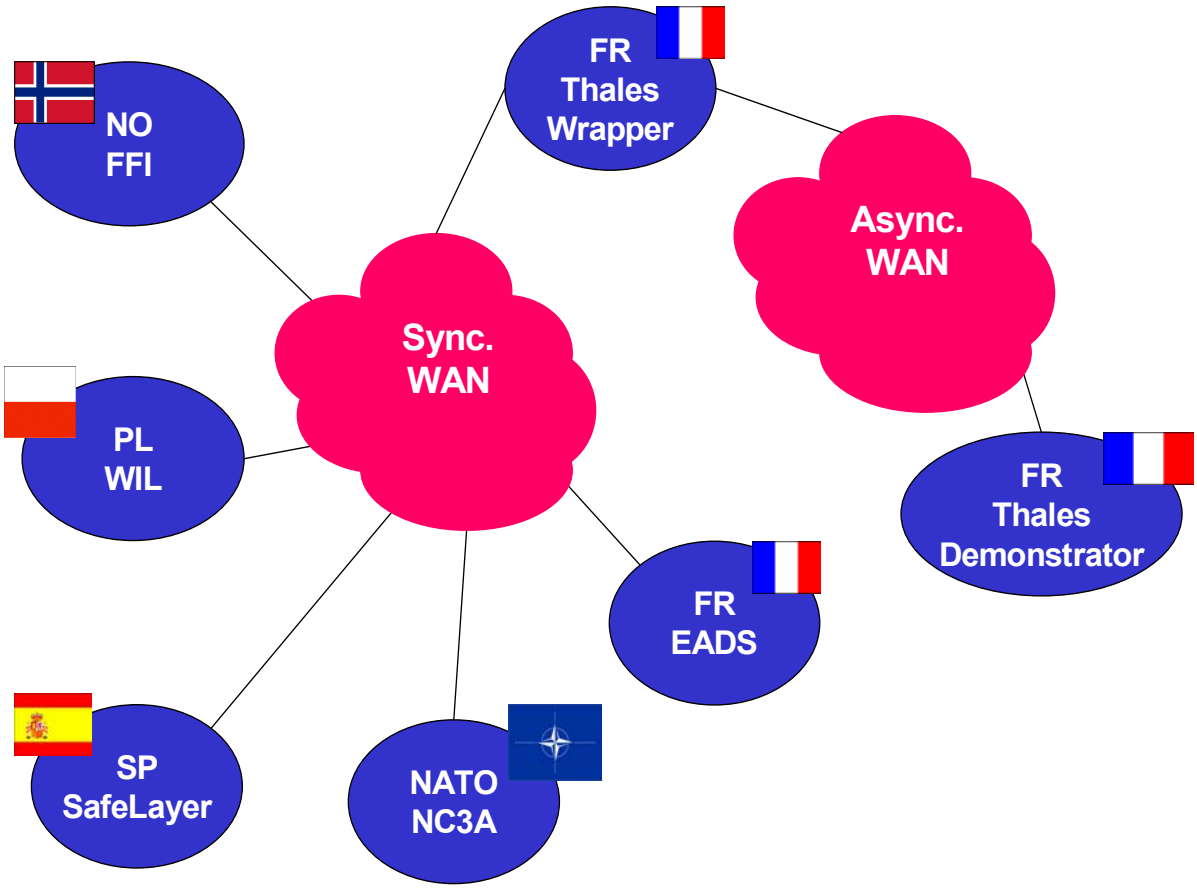

Figure 5.10: Asynchronous vs. Synchronous.

5.2.2.5 Demonstrator Flexibility

A main feature of the Demonstrator is its flexibility. From an external point of view, the Gateway exposes operational services that are dynamically added to or removed from a registry. The connector in the IEG allows bridging to the external world with another architecture, which may be service-oriented or a legacy one.

In the present case, the internal architecture relies on a data dissemination layer. Legacy Systems can plug in this layer through an adaptation connector. Systems can be easily added or removed. Additionally required development consists only of the Adaptation connectors and possibly a Data model.

This flexibility allows building generic components suitable to different platforms. Moreover the insertion of legacy systems, even legacy infrastructures, will be the future challenge for meeting NEC requirements in a progressive migration.

In a summary but not comprehensive, the Demonstrator has the following features:

- Adding/Removing of active services in operation;
- Adding/Removing of Systems providing/consuming services in operations when appropriate adaptation connectors are ready; and
- Insertion of legacy systems or infrastructures.

5.2.3 FFI / Thales (Norway)

This section describes the architecture of the Norwegian SecSOA Demonstrator at NATO CWID 2006. It has been developed by FFI, with substantial contributions from Thales Norway, especially on the security components.

The FFI Demonstrator is a distributed system consisting of several loosely coupled modules deployed across different physical machines. Figure 5.11 illustrates the CWID06 deployment of the Demonstrator. The Demonstrator is made up of the following logical parts:

- Simulation environment;
- A set of DPNs (Data Publishing Nodes);
- Service Registry based on UDDI; and
- Certificate server based on LDAP.

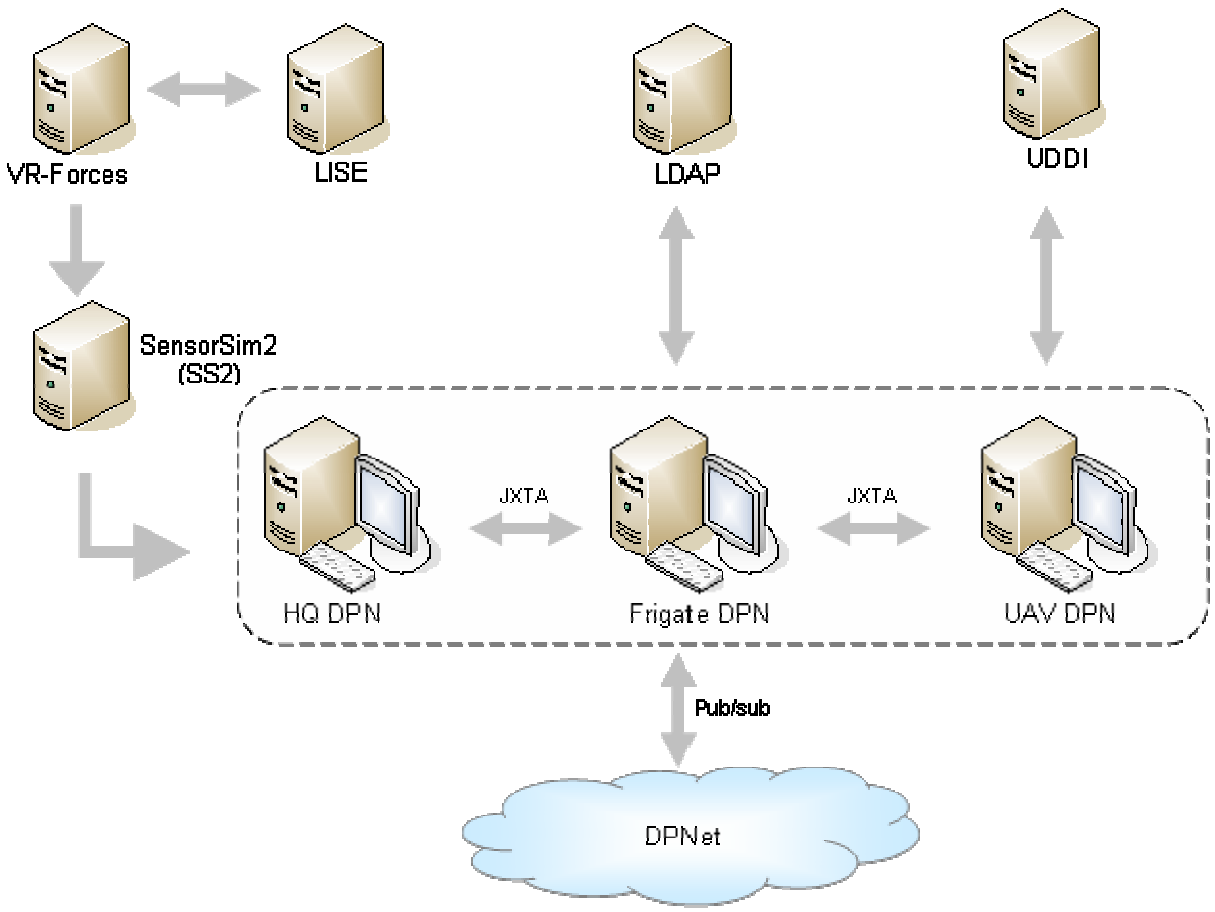

Figure 5.11: Demonstrator Deployment CWID 2006.

All components are interconnected in a national LAN, and reachable from the Data Publishing Network (DPNet), which represents the outside world that we interact with.

The simulation environment consists of three physical machines, VR-Forces, LISE and SS2. VR-Forces is a commercial application, and it requires a separate license server on the LISE machine, in order to run correctly. VR-Forces simulates object movements of both friendly and hostile units, and sends these to

SensorSim2 (SS2). SensorSim2 is an application which adds sensor simulations (e.g. radar simulation) to data coming from VR-Forces. SensorSim2 sends the simulated objects and sensor data to the DPNs.

Each of the DPN nodes represents a physical unit having a set of sensors attached to it, with an ability to communicate to other DPNs. Three national DPNs were used at CWID06:

- HQ DPN, which represents the national headquarters;
- Frigate DPN, representing a frigate; and
- UAV DPN, representing an unmanned aerial vehicle (UAV).

The internal DPN-to-DPN communication within the national domain is realized using the JXTA peer-to-peer technology. Each DPN also implements the Publish/Subscribe (pub/sub) mechanism. Publish/subscribe allows a DPN to communicate with other members of the DPNet, either by acting as a data producer or data consumer:

- A DPN can produce data to other DPNs; and
- A DPN can subscribe to and receive data from other DPNs.

At CWID06, the Demonstrator was integrated with the Norwegian Command and Control System NORCCIS II, enabling information exchange based on Web Services. Security functions in the form of a System Protection Component (SPC), was implemented in each DPN, and in the Service Registry. The SPC was developed by the Norwegian team, relying heavily on the contributions from Thales Norway. For CWD06, the SPC was made available to all participating nations for re-use.

The Service Registry is based on a commercially available UDDI registry from Systinet, with added functionality for security and extended search. It allows the DPNs to register their services so they can be discovered by other DPNs. At CWID06, the Norwegian Service Registry was centrally available for use by all nations participating in SecSOA.

5.2.3.1 End-to-End Security

Our implementation uses a combination of several security mechanisms in order to achieve the goal of end-to-end security at the information object level. The following bullets outline the security functionality developed as part of the demonstrator:

- All SOAP messages were attached with a security label, encrypted and signed;
- All advertisements in the Service Registry were attached with security labels and signed before storage;
- Before any notifications or UDDI records were released to a requestor, their security privileges were checked against the security label of the information objects; and
- A PKI and an LDAP Directory were used for providing the security infrastructure for exchange of certificates and certificate revocation lists.

As shown in Figure 5.4, all components that provide or consume services contained the SPC (System Protection Component). This component handled all parts of the security processing, i.e., performed certificate validation, created and validated signatures, encrypted and decrypted, and did access control based on the security labels. Thus, in our architecture security was handled in an end-to-end fashion.

5.2.3.1.1 SOAP Security

All information exchange was done using SOAP messages. The security of the SOAP messages was based on the use of the OASIS WS-Security standard with extensions in order to include an XML Security

Label. The security label (and other important fields) was bound to the SOAP message by a digital signature. The content of the SOAP messages was compressed, encrypted, labeled and signed before transmission. Upon arrival the security was validated and the originator was identified in order to see if the message came from a reliable source.

5.2.3.1.2 Securing the UDDI Registry

All records stored in the UDDI registry were labeled and signed in order to indicate their sensitivity and to protect them from being changed during storage. UDDI v.3 defines APIs for access to the data within the service registry. Two of these are the Inquiry API, which is used for searching for records, and the Publish API, which is used for insertion and updates of records. In order to secure these interfaces and enforce differentiated access control on the stored records, we used the System Protection Component (SPC) as part of the Security Abstraction Layer in front of the UDDI APIs. This security abstraction layer performed the WSS related security processing of the SOAP messages (authentication, signature handling and encryption), in addition to performing differentiated access control on the UDDI records based on the security labels of the UDDI records and the privileges in the user certificates.

5.2.3.1.3 Securing Subscriptions and Notifications

When the Notification Producer had a notification to distribute, the NotificationProducer matched the InformationSecurityLabel in the SOAP message of the notification against the User Privileges registered for each subscription. For each of the subscriptions for which it identifies a match, it released the Notification to the Subscriber associated with the subscription.

5.2.3.1.4 Security Infrastructure

The security infrastructure consists of a Public Key Infrastructure (PKI) used for handling X.509 certificates and CRLs, and a distributed LDAP system. Each nation had a Certificate Authority (CA) and trust was distributed amongst the systems via the exchange of the National PKI's Root certificates. The Root certificates were placed into a "trust file," which the systems used during digital signature verification. Only the PKI Root CA certificate was needed in the trust file because the CA certificate and the certificate of the system that performed the signature were available from the LDAP Directory together with the Certificate Revocation Lists (CRLs). Smartcards were used to store user certificates. The replication of the LDAP information was done periodically by exchanging LDIF files using Publish/Subscribe functionality. By subscribing to the periodical update of each national LDAP server (using the WS-Notification specification), the LDAP information replicated was protected by the SOAP security functionality. This also shows how a non-XML legacy system like LDAP may be included using Web Services technology.

5.2.4 Safelayer (Spain)

SP-ATI2 stands for Advanced Trusted-Information Interoperability and it is the formal title of the Spanish Demonstrator for CWID 2006.

This document describes the Spanish Secure SOA demonstrator at NATO CWID 2006, including the results and the lessons learned.

5.2.4.1 Demonstrator Description

The Safelayer' SP-*ATI2* demonstrator:

- Illustrated how to secure the data exchange between Nations using the new Web services technology in a trustworthy common environment;

- Allowed data exchange in a trusted environment, providing authentication mechanism, data confidentiality and the data integrity by means of using strong encryption technology; and
- Guarantied the interoperability and the global data exchange, establishing a *global trust environment between different Nations*.

For this purpose SP-ATI2 brought the concept of the PKI-Based Trust Service Provider into practice. The PKI-Based Trust Service Provider:

- Implements PKI-based security functions to Web applications demanding digital Signatures and data Protection, alongside authentication, authorization and audit functions; and
- Implements a full policy-based system in which the execution of a service is always guided by the rules of a concrete applicable service Policy.

Safelayer's TrustedX Web services Platform were used to implement a PKI-Based Trust Service Provider.

5.2.4.2 Demonstrator Architecture

As we have seen previously, the main purpose of the demonstrator is to show interoperability among different Nations using standard Web services in an application to securely interchange data. The chosen paradigm is a publish/subscribe schema in which a Nation will subscribe to different topics that another Nation will publish. All interchange messages will be secured in a standardized way. For more details see others chapters in this document.

The main goals of the SP-ATI2 Demonstrator are two fold:

- Technical interoperability among different implementations of standard Web services, namely, WS-Notification, WS-Security, SOAP, WSDL, etc.
- Technical interoperability in security aspects surrounding PKI mechanisms.

On the other hand, along side the aforementioned goals, it is also a primary target to test the technical interoperability of Safelayer product called "TrustedX secure Web services Platform".

Figure 5.12 shows the basic architecture of SP-ATI2 depicting all the participating components.

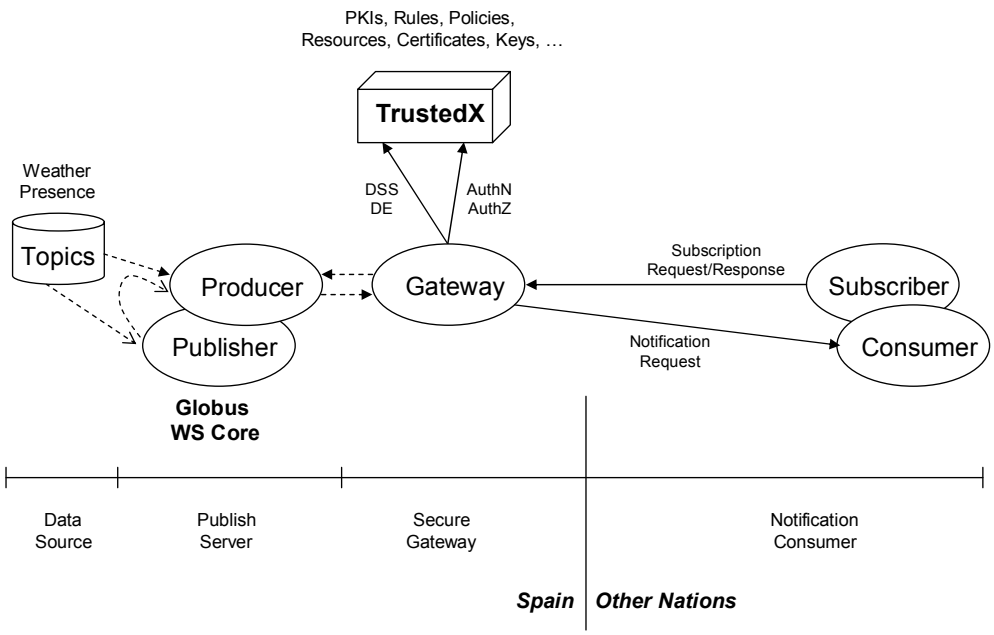

Figure 5.12: Architecture of SP-ATI2.

TrustedX is central in the performance of security services and also, very important, in the management. The idea is that TrustedX services are accessed by a gateway that intercepts the messages of the final Web service provider, in this case, a Notification service. In doing so, TrustedX will perform:

- The Authentication of the originating entity (the subscriber) when a subscription request message comes in, as well as the integrity of the message. For that, the SOAP message header containing a WS-Security compliant Digital Signature will be verified.
- An Authorization process (access control enforcement) so that the subscriber entity has the right to consume the subscription service in general, and if it has the right to subscribe to the requested Topic in particular.
- A Decryption process in order to decrypt the SOAP request message body.
- An Encryption process in order to encrypt the SOAP response message body.
- A Digital Signature generation process in order to protect the SOAP response message.

TrustedX takes care of unwrapping and wrapping the security elements of the request and response messages to/from the final provider service (Producer) and the consumer (Subscriber). However, and most importantly, TrustedX semantically verifies and checks the security elements following the locally defined policies. In this way, TrustedX provides to other components of the Service Oriented network a centralized container and processor of:

- All trusted PKI elements, that is, Certification Authorities, Validation Authorities, and Time Stamping Authorities.
- A policy-based mechanism to define both
  - i) Which are the allowed entities (Authentication Policies); and
  - ii) That have the right to access a set of defined resources (Authorization Policies).
- A policy-based mechanism to establish trust development in digital signatures (authenticity and non-repudiation) and the plethora of trusted third parties defined in the system (Digital Signature

Policies). Trust development is done in a semantic way, that is, a policy is applied to the digital signature in order to get a measurable Trust diagnostic by processing all security elements (including some environment-related elements, like time, IP addresses, authentication mechanism, etc.).

- A policy-based mechanism to protect data (Data Encryption Policies) so that data can be encrypted and decrypted taken into account security and environment-based elements.

TrustedX also provides a trusted point of robust security by centralizing sensible key material in different degrees as well as certificates, revocation information, etc. All accesses to information, whether sensible or not, are controlled by a policy enforcement mechanism that will require the right credentials to access the resource. Moreover, data communication is also secured by the right protocol, for instance, SSL.

It is also worth to shortly describe how the Gateway component works for outgoing Notification messages (from Producer to Consumer):

The Producer application (Globus Toolkit version 4) sees the Gateway as an HTTP Proxy, thus the Gateway will forward the protected message to the suitable end point.

The Gateway is configured to authorize the use of TrustedX services to a Producer application with a fixed IP address. Nevertheless, inner traffic is supposed to be friendlier than outer traffic.

The Gateway uses the SOAP message header wsa:To endpoint reference to select the right encryption certificate/key.

The Gateway signs the SOAP message using the name (X.509SubjectName) of the Producer application.

5.3 LIST OF TEST CASES

The test cases described below are the ones agreed by this working group:

1) Enhanced end-to-end WS-Security

   Show that all SOAP messages exchanged between nations are secured using PKI-based end-to-end object level security mechanisms.

2) Information delivery using Publish/Subscribe

   Show that services are made available to others by publishing, and that efficient delivery of updates is achieved by subscribing to an information delivery service.

3) New services made ready for use

   Show that a new instance of a known service interface, or a new service with a not previously defined data format, can be published and used.

4) COI Cooperation

   Show NetCentric cooperation between the C2 and ISR COIs using the object oriented MIP data model.

5) Access control at the object level

   Show that the information objects (WS-notifications or UDDI records) can be securely marked and that only users with the right security privileges can access/receive them.

6) Distributed Security Management

   Show that Certificates/user privileges can be issued or revoked, and evaluate the time that is needed until the change has been propagated to all nations involved.

7) Dynamic Service Replacement

   Show that a broken service can be automatically replaced.

Chapter 6 – RESULTS AND LESSONS LEARNED

This chapter describes the results achieved by IST-061. This is done by first describing the view of each participating nation on the national outcome of the testing done at NATO CWID 2006. References to CWID Test Case definitions and results are given for each nation, as well as results of interactions with participating non-members of IST-061.

Second, the results are described on an IST-061 group level. Besides the very important Specification document, resulting from an integration of national contributions and technical discussions in the group to arrive on proposed solutions for "Secure SOA supporting NEC", the perceived group-level results are summed up. As has been mentioned above, a shortened version of this specification document is presented in Annex A of this report. Some of the material contributed to the specification is strictly limited in distribution to the members of the group and had therefore to be removed before the specification could be included.

Finally there are comments on what may be considered "lessons learned" form this work.

6.1 RESULTS FROM PARTICIPANTS

This section highlights the results from the perspectives of the individual participating organizations and companies in no particular order.

6.1.1 Results from Thales (France)

The transformation of the armed forces is leading to a redefinition of both, the operational requirements and the way in which the armed forces are organised. This impacts on how system architecture should be organised as well. A Thales department, the Battlespace Transformation Centre (BTC) is working on assessing how the transformation impacts System Architecture and Operational Concepts each other. This is why it was interesting in experimenting and getting proof of the relevance of SOA principles and their implementation on Web Services in that context.

The targeted operational capabilities were defined according to drivers such as an enhancement of dynamic connectivity, the security issues within a coalition, and two tenets of NCW that are the information sharing and the shared situational awareness. The results from an operational point of view were the achievement of the following capabilities:

- Dynamic creation of ad hoc COI on the battlefield: Services were in topics regarding two Communities of Interest (COI) (.i.e. C2 COI and ISR COI). COI Members on different platforms may publish/subscribe to their topics. Hence, they send/receive relevant information to/from appropriate members.
- Sharing of services providing situation pictures: Platforms disseminated information over the WAN through services providing Land and Maritime Pictures.
- Sharing of Sensor resource: The platform requested information from a Sensor on a remote platform and a change of its tasking.
- Dynamic Management of Security: The Security Certificates given at planning time allowed managing the initial connection between Platforms. When connected, directories of security certificates were synchronised and updates of security parameters were processed with an immediate effect. Then platforms managed access authorisation to information "on the fly" by revoking security certificates.
- End-to-end secure exchange with different level of security.

The system is composed of two main sub-systems: An IEG and an Application Infrastructure (For more details, please refer to Section 5.3.2). The IEG contains the information assurance and discovery services. The Application Infrastructure features systems offering information delivery services from situation pictures of C2IS to data sensors. The system was built according to assumptions that were verified during the experiment:

- In that architecture all security functions are located in the IEG on the edge of the platform. A central location allows facilitating security management such as the configuration management of security certificates as well as giving a unique access point to the remote platforms. The experiment showed this choice brought an easier buildability of the system and therefore a better maintainability.
- As the discovery services are in the gateway, the platform can only expose those services to the coalition that the platform operators are willing to provide. There is a strict segregation between the internal registry containing all platforms services and those that are exposed to the coalition.
- The application Infrastructure relies on a Data Dissemination Layer, which allows plugging in systems supporting services in an easy way. It allows an easy adding/removing of systems as well.

The benefit of such a system architecture is to get the capability to present an SOA interface to the coalition side of the system while internally we can insert legacy architectures or legacy systems. Additionally it gets a great flexibility because the number and the type of systems can be modified in an easy and affordable way thanks to the plug in on the data dissemination layer.

The technical services such as discovery and information assurance services were enablers in the interoperability of platforms. Despite their different implementation, the participants had the same services providing the same functionality and sharing common standards.

The implementation of the demonstrator was based on Open Standards and used COTS implementations whenever feasible. There was no showstopper although some COTS take some liberties with respect to the standards they implement. The demonstrator implementation used some of the standards recommended in the NNEC Feasibility Study. However, some discrepancies about the used versions have to be noted.

The result is a set of technical components abiding by WS Standards, UDDI and LDAP. It would not have been possible to develop this demonstrator within a so short period of time without the availability of products from the commercial market and the open source community (For more details, please refer to Section 5.3.2.5).

However, the development showed weaknesses of some of these standards in terms of their maturity and the scanty number of implementation. For instance, WS-Notification has few implementations. Moreover, when it is implemented, the COTS did not fully abide by this standard. An example of such an incomplete implementation is the Globus Toolkit version 4 (GT4). Nevertheless, when many of these problems were occurring, often they were fixed thanks to information and source code solution available on the Web.

The issue is not the difficulty or the cost to get the mastery of Web Services. The price is not so high. It is rather the relevance of the choice of standards. The standards are not mature enough, and they may evolve. Hence some of them may become obsolete.

Despite these considerations, the result of the implementation was satisfactory from a technical point of view. The major point is the capability to make legacy systems and infrastructures available as Web Services with reasonable effort. In that way we get a smooth and affordable transition from legacy to NEC.

There was a significant effort to achieve this implementation. It encompasses a large scope of capabilities when comparing with other CWID06 demonstrations. Of course these capabilities do not cover all the

operational needs for a NEC context, but the work done is a good achievement and can mitigate some future risks, as it has been the opportunity to validate many assumptions and to reject some others.

6.1.2 Results from FFI (Norway) and Thales (France)

For the Norwegian participants in IST-061 and NATO CWID 2006, the work on "Secure SOA supporting NEC" has taken a lot of effort, especially if you include all preparation and specification work. The results of those efforts can be summed up in an extremely valuable learning experience for the participants.

The Norwegian focus of the NATO CWID experiment may be described as the combination of four technical areas:

- Dynamic Service Discovery;
- Publish/Subscribe style information exchange;
- End-to-end security; and
- Use of Object-oriented C2IEDM.

The goal of the experiment was to prove that the combination of the four areas could be implemented, and demonstrate information exchange between nations based on the implementations. Underlying is the strong assumption that technology like this is essential to support Network Enabled Capability.

The Norwegian testing at CWID was formally based on seven Test Cases defined up front. The NATO CWID 2006 Report concludes on the Norwegian SecSOA at an overall level: "The information was retrieved successfully by partners." A closer look into the Norwegian Test Cases is given in Table 6.1 below. The numbering (TC#) refers to the NATO CWID Test Case Tool.

Table 6.1: Results of Norwegian Demonstration

TC#	Heading	Description	Result
615	Information delivery using Publish/Subscribe	Show that services are made available to others by publishing, and that efficient delivery of updates is achieved by subscribing to an information delivery service	Success
616	New services made ready for use	Show that a new instance of a known service interface, or a new service with a not previously defined data format, can be published and used	Partial success (50%) – did not test new data formats
617	COI Cooperation	Show Net Centric cooperation between the C2 and ISR COIs using the object oriented MIP (C2IEDM) data model	Success
618	Enhanced end-to-end WS-Security	Show that all SOAP messages exchanged between nations are secured using PKI-based end-to-end object level security mechanisms	Success
619	Access control at the object level	Show that the information objects (WS-notifications or UDDI records) may be securely marked and that only users with the right security privileges can access/receive them	Partial success (90%) – UDDI records were not secured
620	Distributed Security Management	Show that Certificates/user privileges can be issued or revoked, and evaluate the time that is needed until the change has been propagated to all nations involved	Success
621	Dynamic Service Replacement	Show that a broken service may be automatically replaced	Not tested

As stated in the table, there are some areas where our initial expectations were not met. First, the Test Case on "Dynamic service replacement" was cancelled. Second, "New services made ready for use" was not tested "with a not previously defined data format". And finally, the implementation of security on the Service Registry turned out not to be compatible with the respective implementation on Publish/Subscribe. As a result, we could not fully test object level security in the Service Registry.

These are to a large extent implementation shortcomings, and not limitations of the technical concept given in the specifications. But some of them clearly point in the direction of limited "dynamicity", especially when it comes to the Service Discovery and the use of metadata.

On the national level, working together on-site at CWID 2006 has reinforced relations between the participants of the Norwegian CWID delegation, creating a good foundation upon which an even better CWID participation for the coming years can be built. Also, the SecSOA experience helps FFI and the Norwegian Defence in the evaluation of leading technology and how to use it in upcoming work to implement a national Information Infrastructure.

For FFI, the results can be summed up into experience for the scientists and documentation aimed at internal and external use. Being clearly visible as a CWID 2006 participant is also assumed to be a positive result for FFI.

6.1.3 Results from Safelayer (Spain)

The demonstrator proved that Nations can exchange information in a common and secure way. Using the new Web services technologies and the security PKI services they provided information to authorized Nation-consumers.

The SP-ATI2 scenario proved:

- Nation-consumers were able to find the topics that Nation-producers provided and exchange data.
- The parties were able to secure and authenticate the exchanged data as well as ensured the its integrity and the confidentiality of the exchanged data.
- Data exchange was possible among Nations with different PKI domains, offering the same security guaranties to inter-domain communications.

The CWID Web-based Test Case Tool was used for the planning, execution and documentation of NATO CWID Test Cases. The Test Case Tool is an online, interactive database used to coordinate and document interoperability testing and experimentation in NATO CWID.

Scenario test cases were developed based on Interoperability Test Requirements within the context of the SP-ATI2 demonstrator. Interoperability test cases were created based on trials intentions to test specific system capabilities against the following partners:

- France (IT/ID Name: FR-SAIS);
- Norway (IT/ID Name: NO-SecSOA);
- Poland (IT/ID Name: PL-SOA WS); and
- NC3A (IT/ID Name: NATO-CSSI-IVAS).

Trials used the Test Case Tool to record test plans and results in cooperation with their partners (information providers and consumers). Following the execution of a test case, trial partners agreed on a result (Success, Limited Success or Interoperability Issues), then submitted their recommendations to the Interoperability Assessment Team for review.

SP-ATI2 demonstrator focused only on the security aspects, considering the following tests:

- **End-to-End Message level security**: Show that SOAP messages are encrypted and digitally signed using user digital user certificates. Intended recipients must be able to verify the authenticity and the integrity of the data and then to decrypt the information.
- **Digital certificate/user privilege management**: Show that certificates/user privileges can be revoked. Digital signatures using revoked certificates must not be accepted. No information must be provided to users presenting revoked certificates.
- **Access control at the object level**: Show that the information objects (WS-notifications) may be securely marked and that only users with the right security privileges are allowed to access/receive them.

6.1.3.1 Conclusions

CWID 2005 revealed a growing interest on Web services and XML Technologies. Future CWID editions will offer the opportunity to analyze new fields of application, demonstrate the benefits of these technologies, while maintaining the Web services security and PKI technologies in the mainstream. The NATO RTO/IST-061 Secure SOA demonstrators provided real application-cases testing the best state-of-the-art of the technology and its application to the military field.

On the other hand, new technologies usually mean new security protocols in the arena. XML flexibility may reveal interoperability problems on Web service security protocols which must be specially tested: SP-ATI2 makes focus on security interoperability aspects. Preformed tests showed that despite of using different technologies; demonstrators were able to *interoperate and to securely exchange information* (strongly authenticated, digitally signed and encrypted). Norway-Spain and Poland-Spain demonstrator tests where all successful while partially successful in NC3A-Spain.

Finally, we had the opportunity to analyze different architectural approaches of the demonstrators. While most of the demonstrators used PKI toolkits to incorporate security to application, SP-ATI2 used a completely different approach, based on the concept of the PKI-Based Trust Service Provider. *This approach showed that thanks to SOA and Web services, security services can be independent from the applications*. It also simplifies the integration and allows the audit, changes and security upgrades to be transparent to the applications.

6.1.4 Results from EADS (France)

Challenges and objectives for this demonstration were multiple and at different level. The main challenge was to do real experimentation of all the topics that were discussed in this working group. The realization in a CWID context with real multinational systems was a great experience. EADS' objective in this demonstration was to bring legacy systems to interoperate with other nations in this particular context.

Another good challenge during realization was to always keep in mind that SOA is made for flexibility and all architecture used should always be simple (simple to implement, simple to deploy, simple to use). A lot of effort had been spent in this item even if it is not demonstrative.

The modularity of infrastructure solutions is also a big issue. As web service provider, we were able, theoretically, during the CWID demonstration to use any of the web service infrastructures.

During CWID, one of the big challenges was to being able to interoperate with multiple countries and organization at the same time. This CWID gave us the opportunity to make tests with Norway, Poland, NC3A and Spain. It gave us the feedback of the effort we should spend to realize loose coupled interoperability. This effort should be compared to MIP trials that interoperate with messaging or

database. During tests, we also observed that each team was able to work independently. Only one distant service is sufficient for tests. And when different service providers are connected at the same time, we had a global vision of situations without reconfiguring. In the battlefield, having this possibility to be connected to a secured network and to get the common operational picture fitted to our needs (without browsing a registry, without configuring service providers …) is a first step for network centric warfare.

6.2 GROUP-LEVEL RESULTS FOR IST-061

The primary result from the work in IST-061 is the Specification document. Version 1.0 was agreed on and distributed to the group March 17, 2006. An updated version is to be found as an annex to this report.

A secondary result is the proof-of-concept that was performed as a contribution to NATO CWID in May and June 2006. The viability of the experimental implementations of the specification reinforces the value of the specification document.

The NATO CWID results are good examples of how SOA using Web Services may be a suitable technology for systems that are to support NEC. Military resources are made available as services, accessible from the network. Services are described by metadata that is published on the network.

The aim of the specified solutions is to provide more flexibility and adaptivity. The benefits of using this secure and dynamic SOA solution in a military environment are as follows:

- Military resources can be made available as services that may be accessed over a communication infrastructure;
- Information is characterized by metadata and published in the network;
- Efficient discovery and downloading of and subscription to relevant information;
- Faster deployment of new technology and functionality;
- Dynamic reconfiguration of functionality within a relatively short timeframe; and
- Integration of functionality over different networks and heterogeneous technologies.

In the security area, IST-061 has initiated important work. End-to-end security at the object level is an important contribution to existing security regimes. It is an interesting future solution with a great potential, given adequate security policy and management procedures.

IST-061 has developed an experimental solution for the future that has been federated in a European environment. The work has demonstrated how a UDDI Service Registry can be extended to meet future requirements. A practical way of using XML-based MIP/C2IEDM for multinational information exchange has been shown.

The results achieved clearly indicate that SOA is a good foundation for the NATO Network and Information Infrastructure. SOA has the potential of overcoming the limitations of current “stove-piped” solutions.

It is hard to determine the level of success for an effort like IST-061. Many important goals have been achieved, but in hindsight it is easy to spot parts that could have been improved. Overall, and especially when speaking to the world outside the IST-061 group itself, the many positive achievements certainly qualify for the label “Success”. This group can be proud of its results.

A more detailed discussion within the group will reveal substantial goals at the nation and participant level that were not met. “*Limited dynamics*” and “*Specifications were not detailed enough*” are examples of

limiting factors. This may indicate that the correct label on behalf of the participants should be "Partial success". The lessons learned will indicate some shortcomings that can be read as reductions to the success level. On the other hand, it may be argued that those lessons learned are very valuable results in themselves, and should count positively instead of limiting the success.

6.3 LESSONS LEARNED

The experimental implementations relied heavily on Commercial Off The Shelf (COTS) technologies. COTS does not work reliably yet, and is not always compliant with the standards. Standards are not even stable yet. There are bugs in COTS and open source software.

To ease implementation, there was widespread sharing of software components within the group. This makes it easy for smaller participants to join, which is positive, but you get less independent implementations and less validation accordingly. There were only two fundamentally different implementations of IST-061 during CWID.

The group learned numerous lessons about complexity. To sum it up:

- The volume and complexity of MIP/C2IEDM was a problem;
- XML technology is not mature enough; and
- Compatibility problems create needs for agreements on more than schemas and interface descriptions.

Communicating the results of the work is a challenge. Technical elements like SOA and security are hard to visualise in a demo. Although we had agreed on a Scenario framework with a high degree of independency for each nation, we did not make it to a common "live demo" at CWID. There are different views within the group upon publication of the written results.

At CWID, the group members had to participate from their respective national delegations. It is hard to become visual as a group that way. A wish for future events like CWID is to facilitate presence as a NATO RTO group.

The IST-061 group recommends to other NATO RTO groups to conduct implementations and/or demonstrations – e.g. at NATO CWID – in order to create more results than "just paper". Nevertheless, it should be realized that implementation work is generally much more resource consuming than writing reports. But it is valuable to get detailed knowledge on the subject and to validate or reject the assumptions.

Chapter 7 – SUMMARY AND CONCLUSIONS

This chapter will summarize the Final Report of NATO/RTO IST-061 RTG-027 on "Secure Service Oriented Architectures (SOA) supporting Network Enabled Capabilities". After introducing the general concept of SOA, the report has shown how these principles can be applied to the military concept of Network Enabled Capabilities. This general overview has been followed by a description of which technologies have been used by the group. Building on this, the scenario for the demonstrator of the group has been described. This description contained both, a military "story" and a high-level description of the technical building blocks used to create the demonstrator. The "story" described a sequence of events including the data exchange associated with them.

After the description of the actual realization of the demonstrator at the CWID site, the results of the demonstration have been explained. This has been done first for each participating organization and company and then from the perspective of the group. It was shown that, just as anticipated in the NATO NEC Feasibility Study, SOA is a valid technology for implementing NEC concepts. Especially the relatively low effort required to integrate two additional partners (Safelayer and NC3A) into the demonstration at a very late point in time shows that the expectations concerning the chosen approach are justified.

Nevertheless, it has become obvious that it is difficult to have products from different vendors interoperate, even if they claim to implement the same version of the same standard. This can in part be attributed to the fact that many SOA and Web Services related standards are quite new. Therefore, they still lack the required specificity to ensure interoperability between independent implementations in some places. At the same time, products often do not correctly implement the standards because of misunderstandings or programming failures.

Interoperability among different implementations of the same standard is a crucial factor to ensure seamless integration of military units from different nations. To achieve this, interoperability tests are required. The demonstrations and experiments conducted by this group have shown that (at least for the military community) CWID is a very good place for these interoperability tests.

To further enhance interoperability especially of Web Services (WS), the creation of a military WS profile appears to be a good way. In the civilian world, the "Web Services Interoperability Organization" [20] has specialized in these profiles. Active engagement of the military in this group could help to promote deployment of interoperability profiles in both, the civilian and military domain, benefiting both.

Security is one of the key requirements for NEC. On the other hand, current security technologies are not up to the task for several reasons that have been described in this report. Thus, while the demonstrations of the group showed that security is (in principle) possible, a lot of work is still required for operational deployment. The same can be said for the deployment of SOA and WS technologies in networks operating in resource constraint environments (which are sometimes called "Disadvantaged Grids"): While SOA and WS are the right way to go, the group feels that more research and standardization is required in this area.

One of the reasons SOA and WS are not yet ready for deployment is the status of the Service Discovery and Directory technologies. Especially UDDI, which has been used in the demonstrator, is not flexible enough for the highly dynamic environments common in tactical and operational environments. Especially in tactical environments, services can not be expected to cleanly disconnect from the registry. Network interruptions due to for example terrain, weather or enemy action are quite common. Such circumstances require a service registry and discovery service that offers a keep-alive service and handles disappearing services gracefully. On the other hand UDDI is lacking these features.

The Publish / Subscribe principle used during the demonstration has shown to be a good compromise between pure pull and pure push approaches. While the consumer can decide which topics are relevant for him, he does not have to constantly poll the producer for updates but is informed automatically. This reduces unnecessary network usage significantly.

For the general approach this group has taken, it concludes that the drawbacks such as the significant effort required for the implementation of the demonstrator are more than compensated by the advantages described in this report. This is true as well for the approach of having CWID 2006 as a fixed (and externally defined) date of delivery. Thus, work has been more focused on the central objectives of the group.

The results achieved clearly indicate that SOA is a good foundation for the NATO Network and Information Infrastructure. SOA has the potential of overcoming the limitations of current “stove-piped” solutions.

Even though in hindsight some aspects of the work could have been optimized, many important goals of IST-061 have been achieved. These many positive achievements certainly qualify for the label “Success”. This group can therefore be proud of its results. Even the additional “lessons learned” are very valuable, although not necessarily intended, results.

7.1 RECOMMENDATIONS FOR FUTURE NATO ACTIVITIES

While the group does not recommend a continuation of its complete spectrum in a direct follow-up activity, several topics for future research within NATO/RTO have been identified. The primary topics are security technologies for Web Services and the deployment of SOA in disadvantaged grids. The first topic is already considered in NATO/RTO IST-068 RTG on “XML in Cross Domain Security Solutions”.

As this group has identified shortcomings in the area of service discovery and service registry, it is recommended that this is made the objective of future research activities of NATO. This may be achieved either by creating a dedicated RTG or by including the topic in an existing group.

Additionally, the group recommends that further work on NATO profiles for Web Services is performed in the appropriate NATO forums such as the new Core Enterprise Services Working Group of the ISSC.

Chapter 8 – REFERENCES

[1] Barry, D.K. (2003). Web *Services and Service-Oriented Architectures – The Savvy Managers Guide*; Morgan Kaufman Publishers, San Francisco, USA.

[2] World Wide Web Consortium. (2004). *Web Services Architecture*; http://www.w3.org/TR/2004/NOTE-ws-arch-20040211.

[3] OASIS UDDI. (2006). http://www.uddi.org/.

[4] NATO Network Enabled Capability Feasibility Study, Volume 2.

[5] OASIS Reference Model for Service Oriented Architectures 1.0. (2006). http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm.

[6] Eastlake, D., Reagle, J. and Solo, D. (March 2002). *(Extensible Markup Language) XML-Signature Syntax and Processing*, RFC 3275.

[7] eXtensible rights Markup Language (XrML). (2006). http://www.xrml.org/.

[8] Hoffman, P. (June 1999). *Enhanced Security Services for S/MIME*, RFC 2634.

[9] The Internet Engineering Task Force (IETF). (2006). http://www.ietf.org.

[10] Multilateral Interoperability Programme. (2006). http://www.mip-site.org/.

[11] OASIS ebXML Registry Technical Committee. (2006). http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=regrep.

[12] OASIS Security Services (SAML) Technical Committee. (2006). http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security.

[13] OASIS Web Services Notification (WSN) Technical Committee. (2006). http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsn.

[14] OASIS Web Services Notification (WSN) Technical Committee. (2006). http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsn.

[15] Web Services Dynamic Discovery (WS Discovery). (2006). http://www.w3.org/2002/ws/desc/.

[16] W3C XML Encryption WG. (2006). http://www.w3.org/Encryption/2001/.

[17] W3C XML Signature WG. (2006). http://www.w3.org/Signature/.

[18] W3C Web Services Description Working Group. (2006). http://www.w3.org/2002/ws/desc/.

[19] Wikipedia; Service Oriented Architectures. (2006). http://en.wikipedia.org/w/index.php?title=Service-oriented_architecture&oldid=93400359.

[20] Web Services Interoperability Organization. (2006). http://www.ws-i.org/.

Annex A – DEMONSTRATOR SPECIFICATION

The following document is a redacted version the specification of the CWID 2006 demonstrator. Some participants of the group could not release their contributions to the specification outside IST-061. These parts have been removed. It should be noted that this means that the information in this version of the specification is not sufficient to reproduce the experiments and is thus only provided for informational purposes.

DOCUMENT SUBMISSION FORM

Date: 01/12/2006

Document Reference Number:

NATO RTO/IST-061:

Title: The NATO RTO/IST-061 Secure SOA Demonstrator Specification for CWID 2006

ISSUE: Version 2.0 - modified version

Editor: NATO RTO-IST 061

Purpose:

Classification: NATO UNCLASSIFIED

Document Type: D

D	Finalized Document
WP	Working Paper
R	Record of Meeting
N	Notice of Administrative Nature
A	Agenda

A.1 INTRODUCTION

This document has been written as part of the work of the NATO RTO/IST-061 (Secure Service Oriented Architectures (SOA) Supporting Network Enabled Capabilities). This document specifies the interfaces, protocols and functionality, which must be implemented in order to achieve interoperability between the systems involved in the Secure SOA distributed demonstrator at CWID 2006. Some tutorial text has been included in order to document the chosen functionality and solutions of this demonstrator.

The keywords “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in IETF RFC 2119.

A.2 DEMONSTRATOR ARCHITECTURE

The demonstrator will comprise several national systems or demonstrations interoperating through secured Web Services. Two interactions patterns are demonstrated:

- Publish-subscribe,
- Request-response.

For both interaction patterns the exchanges between nations are secured with XML security technologies.

The development being performed by each participating nations[1] comprises:

- The implementation of this interoperability specification,
- The development and integration of national systems or prototypes to produce and consume data and services. This second part of the development is a national concern not described by this specification.

The logical architecture comprises in each nation a set of software components supporting secured Web Services exchanges in both publish-subscribe and request-response modes (see Section A.4). From a deployment point of view they may be centralised in a single physical gateway or distributed to the end systems (some components being deployed close to the national systems).

Secured Web Services allow:

- Any nation to declare public (coalition) Web Services exposed to other nations,
- One nation to invoke any coalition Web Service provided by an other nation (provided access rights are granted),
- Any nation to declare public (coalition) publishers of information on information Topics,
- One nation to subscribe to a given Topic on a foreign Publisher in order to receive information updates as notifications from that Publisher (push mode).

Examples of such interactions between nations are depicted on the Figure A.1.

[1] Several figures in this document show only two or three nations for simplicity. However, the number of participating nations is not constrained by the architecture. Hence, the five nations participating to the group are potential contributors to the demonstrator. The same applies to NC3A that joined the group in April 2005.

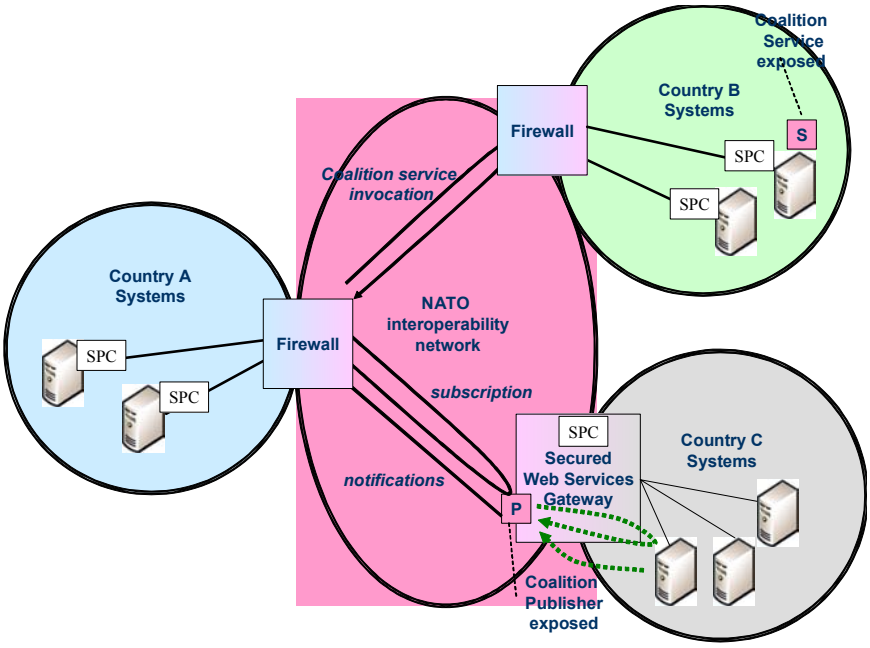

Figure A.1: High-Level Logical Architecture.

Figure A.2 shows the demonstrator developed components.

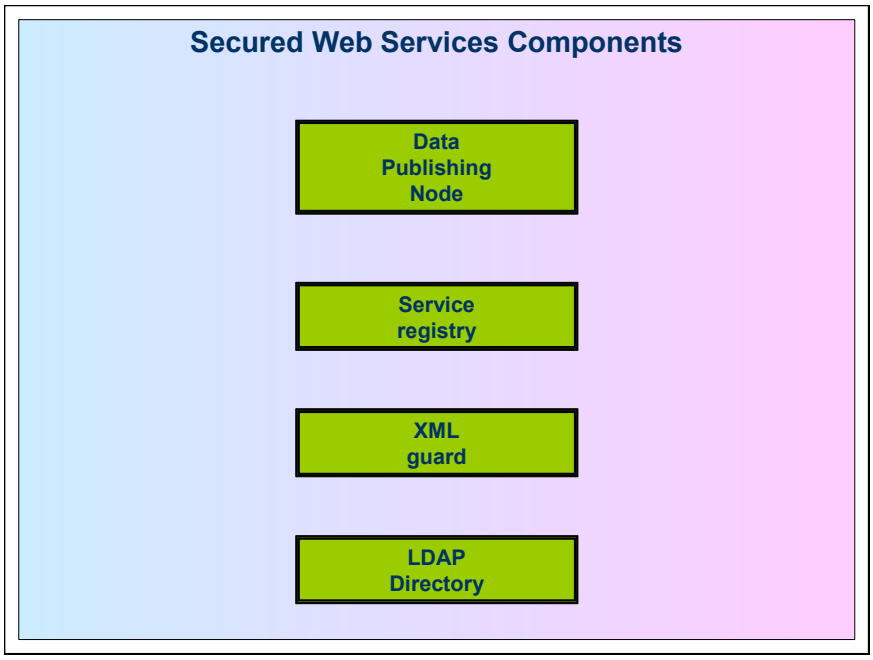

Figure A.2: Logical Components.

The Data Publishing Node supports publish-subscribe exchanges,
The Service Registry provides descriptions of services exposed by a given nation,
The XML Guard checks that inbound and outbound flows comply with security policies,
The LDAP Directory stores certificates and access rights.

The following sections refine the architecture of the demonstrator:

- Section A.3 details the Web Services protocols to be used (addresses technical interoperability),
- Section A.4 describes the publish-subscribe service, its components, protocol and implementation using Web Services standards (addresses technical interoperability),
- Section A.5 introduces the Data Model and Service Model (addresses semantic interoperability),
- Section A.6 details the Service registry architecture and implementation using Web Services standards (addresses technical interoperability),
- Section A.7 details the security architecture, the security features to be demonstrated and their application to Web Services (addresses technical interoperability focusing on security),
- Section A.8 collects information about compression techniques.

A.3 USE OF WEB SERVICES CORE STANDARDS

This chapter describes the Web Services Interoperability specifications to be used in the Demonstrator. The solutions described are based on the use of existing civil or military standards where possible, supported by profiles developed especially for this demonstrator.

A.3.1 SOAP

SOAP version 1.1 using document literal style "Wrapped Mode" SHALL be used. The rational for this is the use of Globus Toolkit. In addition HTTP SHALL be used for transport (SOAP over HTTP).

A.3.1.1 Attachments

Attachments SHALL NOT be used.

A.3.1.2 Potential Issue with SOAP over HTTP

Some of the COTS products used in this demonstration use the HTTP Response message to communicate the result back to the client. This may create problems since some implementations does not wait for the HTTP response to return.

As of the time of writing the extent of this potential issue is not known. Potential solutions include creating wrappers on the client side.

A.3.2 WSDL

WSDL version 1.1 SHALL be used.

A.3.3 Message Transport

HTTP version 1.1 SHALL be used.

A.3.4 UDDI

UDDI version 3 SHALL be used.

See Section A.6 for more details.

A.3.5 Binary XML/Compression Algorithm

See Section A.6 for details.

A.3.6 Publish/Subscribe Specifications

See Section A.4.

A.4 PUBLISH/SUBSCRIBE SPECIFICATION

Publish-subscribe is among the technical services to demonstrate. Participating nations are expected to exchange information using this service. They can be provider (publisher) or consumer (subscriber) of information or both.

Some of the text of this chapter describing architecture details have been removed to meet the Thales publication policy. The places where text is removed is highlighted.

A.4.1 Service Concepts and Features

The publish-subscribe service provides a data–centric interoperability means between nations. Exchanged data are organised into Topics which are identified with a topic name and which refers to a given data structure. Quality of Service parameters could also be attached to Topics. However, this is not envisaged at that stage.

E.g.: a Topic to related IMINT information
Name: "Image_Intelligence"
Data structure:
- intelligence report (free text)
- image co-ordinates (Geodesic co-ordinates)
- the image itself (binary data).

Topics are defined for the scenario exchange needs and are not intended to be changed or extended with new Topics during the experiments. Each nation can subscribe to the defined Topics and can publish information on them.

A.4.2 Service Architecture

The text is removed in this version of the specification

A.4.3 Publish/Subscribe Protocol

Publishers and Subscribers act as representatives of their nation on the NATO Net.

A Publisher interacts with the Service Registry and with Subscribers in the following way:

- A Notification Service publishes itself by sending to the Service Registry a **UDDI Publish message** containing the specification of the Topic.
- Then, it publishes information when it wants to do so. The published information shall comply with the defined data structure for the Topic. If there are Subscribers for the Topic, the Publisher sends a **Notification message** to all the Subscribers. It is the Publisher's responsibility to maintain a list of Subscribers for each Topic on which it publishes information. Several information elements complying with the Topic data structure can be published with one publication action grouping these elements into one **Notification message**. Hence, a **Notification message** includes the Topic name, the publisherId, a list of information elements.
- A Publisher can decide to stop publishing information on a Topic. In such a case, it must unregister from the service for that Topic. An **"update" message** (using the same service key) is sent by the Publisher to the Service Registry.

A Subscriber interacts with the Service Registry services and the Publishers in the following way:

- It first searches the Service Registry to find one or several Publishers that match its information needs. It selects one or several of them and then subscribes to related Publishers by sending them

Subscription message(s). One **Subscription message** can mention one or several Topic(s) of interest. Each subscription is identified by a unique subsciptionId and contains the Topic name and a subscriptionTerminationTime. Publishers are responsible of automatic subscription deletion when this duration is elapsed. A Subscriber can renew a subscription before the subscriptionTerminationTime is elapsed by sending a **Renew message** with the same subscriptionId and the specification of a new subscription duration. Publishers receiving such **Renew messages** should reinitialize their timers accordingly.

- Then, the Subscriber receives asynchronously **Notification messages** from Publishers.
- A Subscriber can decide to unsubscribe from one Topic. It uses the corresponding subscriptionId to do so in an **Unsubscribe message** sent to the related publishers.

A.4.4 Design Constraints, Guidelines and Technologies

The text is removed in this version of the specification

A.4.5 Implementation using Web Services

This section describes the implementation of the described publish-subscribe concepts and protocol using Web Services standards.

The selected standards to do so are the OASIS publish-subscribe standards: WS-Topics and WS-BaseNotification. WS-BrokeredNotification is not used in the current specification.

The security services described in Section A.7 SHALL be used.

The selected versions of those standards are:
WS-Topics: 1.2 draft 01, 22 July 2004
WS-Base Notification: 1.2 draft 03, 21 June 2004

When reference to WS-BrokeredNotification is needed the referenced version is WS-BrokeredNotificatio 1.2 draft 01, 21 July 2004.

There are several reasons for selecting those versions:

- The mechanisms provided by those versions cover most of the demo needs,
- There OSS implementations: version 1.2 Draft 03 is the version implemented by Pubscribe and the Globus Toolkit. Other tools like ServiceMix also provide an implementation of WS-Notification, which is not 1.3 (which 1.2 exact version is not clear).
- WS-BaseNotification v1.3 is very new and no implementation could be found.

This table below provides the mapping between the concepts and protocol described in the previous section and the WS standards for the implementation. When some standard features are not fully used, this is also mentioned in the table below.

Concept	**Correspondence to WS standards**
Publisher	NotificationProducer
Subscriber	NotificationConsumer and Subscriber. The WS-BaseNotification standard makes a distinction between the roles NotificationConsumer and Subscriber. In our case, the Subscriber will also be the Notification Consumer.
Topic	Topic of WS-Topics with the following restrictions: - only one data type per Topic, - only root Topics are used.
COI	Topic Space of WS-Topics.
Notification Message	NotificationMessage. Received through the Notify operation.
Operation	**Correspondence to WS standards**
Service Registry::Register Publisher	To be mapped on Service Registry publication API.
Service Registry::Change Publisher	To be mapped on Service Registry publication API.
Publisher::Subscribe	NotificationProducer::Subscribe Parameters: - ConsumerReference: NotificationConsumer EPR - TopicExpresssion= <topic Qname> - TopicExpresssion Dialect= Simple - UseNotify= true (default value, can be omitted) - Precondition= optional, omitted, - Selector= optional, omitted, - SubscriptionPolicy= optional, omitted - InitialTerminationTime= <xsd:dateTime to be set by subscriber>
	NotificationProducer::GetCurrentMessage
Publisher::Unsubscribe	Not defined in selected standards but provided by Pubscribe. Exists also in WS-BaseNotification 1.3. Replacement solution: The unsubscribe message will be published by Subscribers on a built-in Topic called “Subscriptions”. Subscribers that want to terminate their subscription publish in the “Subscriptions” Topic a Subscription object providing the reference of the subscription to terminate. The Publishers must subscribe to that built-in Topic. The Subscription object is defined by an XML Schema.
Publisher::Renew	Not defined in selected standards. Exists only in WS-BaseNotification 1.3. Replacement solution: The renew information (new termination time) to be published by Subscribers on the built-in Topic “Subscriptions”. Subscribers that want their subscription to be renewed publish in the “Subscriptions” Topic a Subscription object specifying a new termination time. The Publishers must subscribe to that built-in Topic. The Subscription object is defined by an XML Schema.
Not used	SubscriptionManager::PauseSubscription
Not used	SubscriptionManager::ResumeSubscription

Subscriber::AcceptNotificationMessage	NotificationConsumer::Notify Notify message= one or more NotificationMessage NotificationMessage: - Topic= <topic Qname> - Topic/Dialect= Simple - ProducerReference= EPR of NotificationProducer - Message= XML fragment complying with our data model XML schema.

A.4.6 Potential Asset-COI Relationships

Chapter 6 describes how Business Entities are modeled in UDDI. Simple relationships are also modeled. Below is a description of a more sophisticated relationship model between Assets and COIs. This model is not realized in this work, but may be a valuable input for future work.

The COIs, the assets allowed to join these COIs and the topics these assets produce or consume for these COIs are described below.

Four relationships between assets and COIs are supplied:

1. The topicSpaces an asset has the right to produce for all the COIs. This relationship is named *"AllowedProduction"*.
2. The topicSpaces an asset has the right to consume for all the COIs. This relationship is named *"AllowedConsumption"*.
3. The topicSpaces an asset produces at a current time for all the COIs. This relationship is named *"RealProduction"*.

The corresponding UML diagram is shown below:

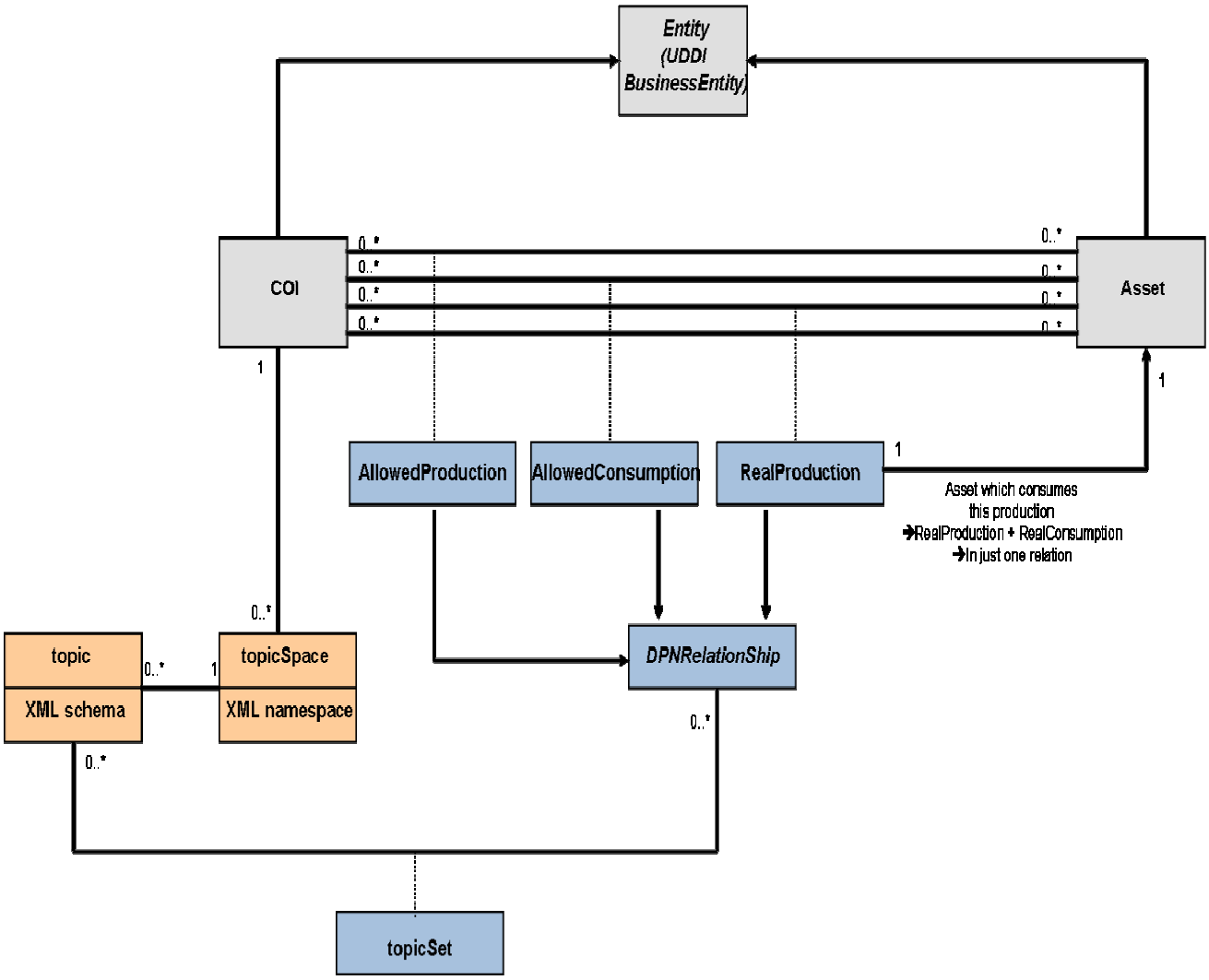

Figure A.3: Relationships between COIs and Assets.

A.5 DATA AND SERVICE MODEL

A.5.1 Data Model

Considering the studied scenario, the data model encompasses only a few data types: maritime and land pictures, Sensor requests/response and MTI Tracks.

- **Maritime and Land Picture**

A subset of MIP XML is defined in Appendix 10.

The Maritime picture reflects the NAVSITSUM and MARINTSUM. The Land picture reflects the OWNSITREP and ENYSITREP.

- **Sensor Request and Response**

SensorRequest

TypeSensorRequest			
Field	Type	Cardinality	Comment
Requester	String	1	Operational Identifier of the requester
Start date	typeDTG	1	
Duration	int	1	minutes
DataNature	String	1	Only one value possible: “MTI”
Area	TypeArea	1	

TypeArea			
Field	Type	Cardinality	Comment
MinLongitude	double	1	Degrees, >0 towards East.
MaxLongitude	double	1	Degrees, >0 towards East.
MinLatitude	double	1	Degrees, >0 towards North.
MaxLatitude	double	1	Degrees, >0 towards North.

SensorRequestResponse

TypeSensorRequest			
Field	Type	Cardinality	Comment
Status	String	1	OK or NOK
RejectionCause	String	1	If NOK, reason for NOK
EffectiveStartDate	TypeDTG	1	If OK, StartDate accepted by sensor system.
EffectiveDuration	int	1	Minutes. If OK, Duration accepted by sensor system.
DistributionTopic	String	1	If OK, Topic on which requested data is published.

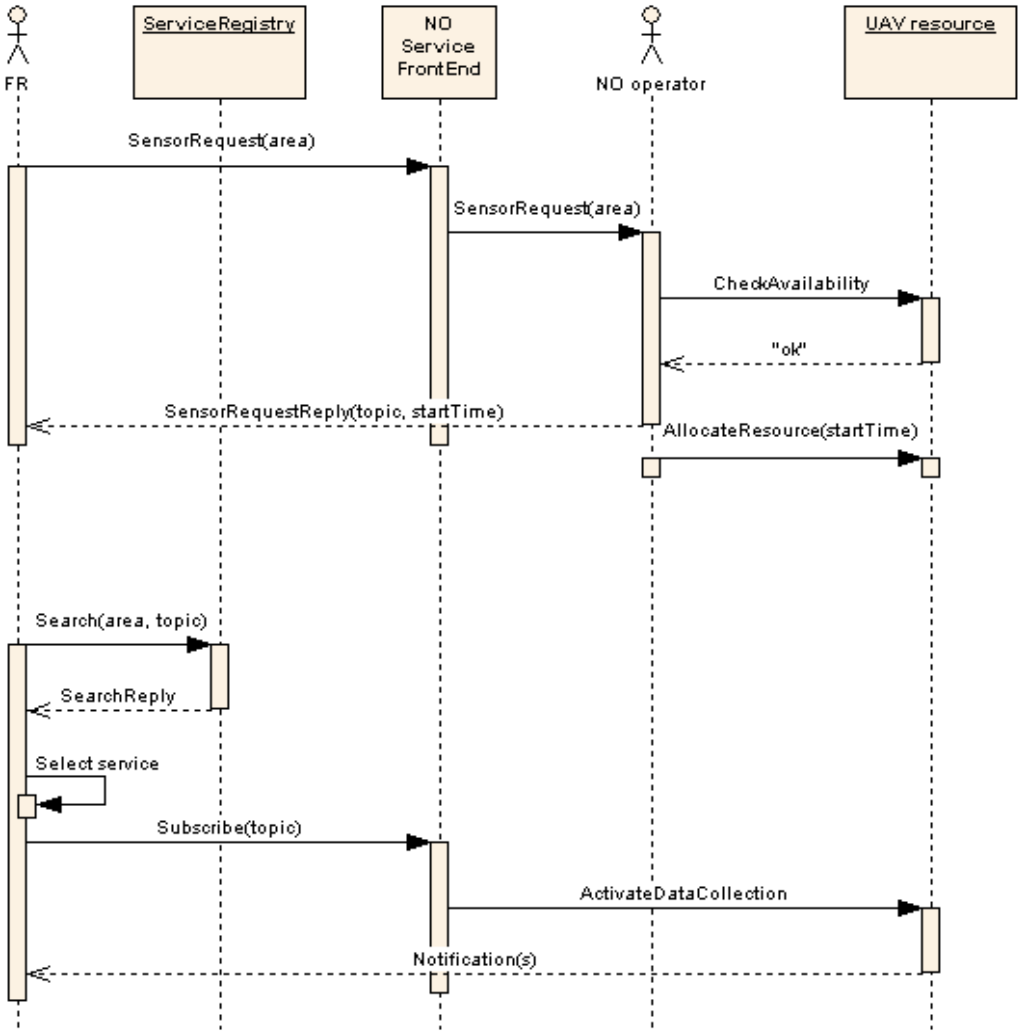

Figure A.4: Sensor Request/Response Sequence Diagram.

- **MTI Tracks**

The MTI Tracks are about exporting data from a sensor system. It represents a snapshot, a technical situation, upon an interval of past time, of which has been detected by the sensor and then processed by the sensor system.

MTI Tracks are contained in a surveillance report.

The MTI Tracks Xml Schema can be found in Appendix 6, Section A6.1.
Additional information on MTI Tracks is in Appendix 6, Section A6.2

Surveillance Report:

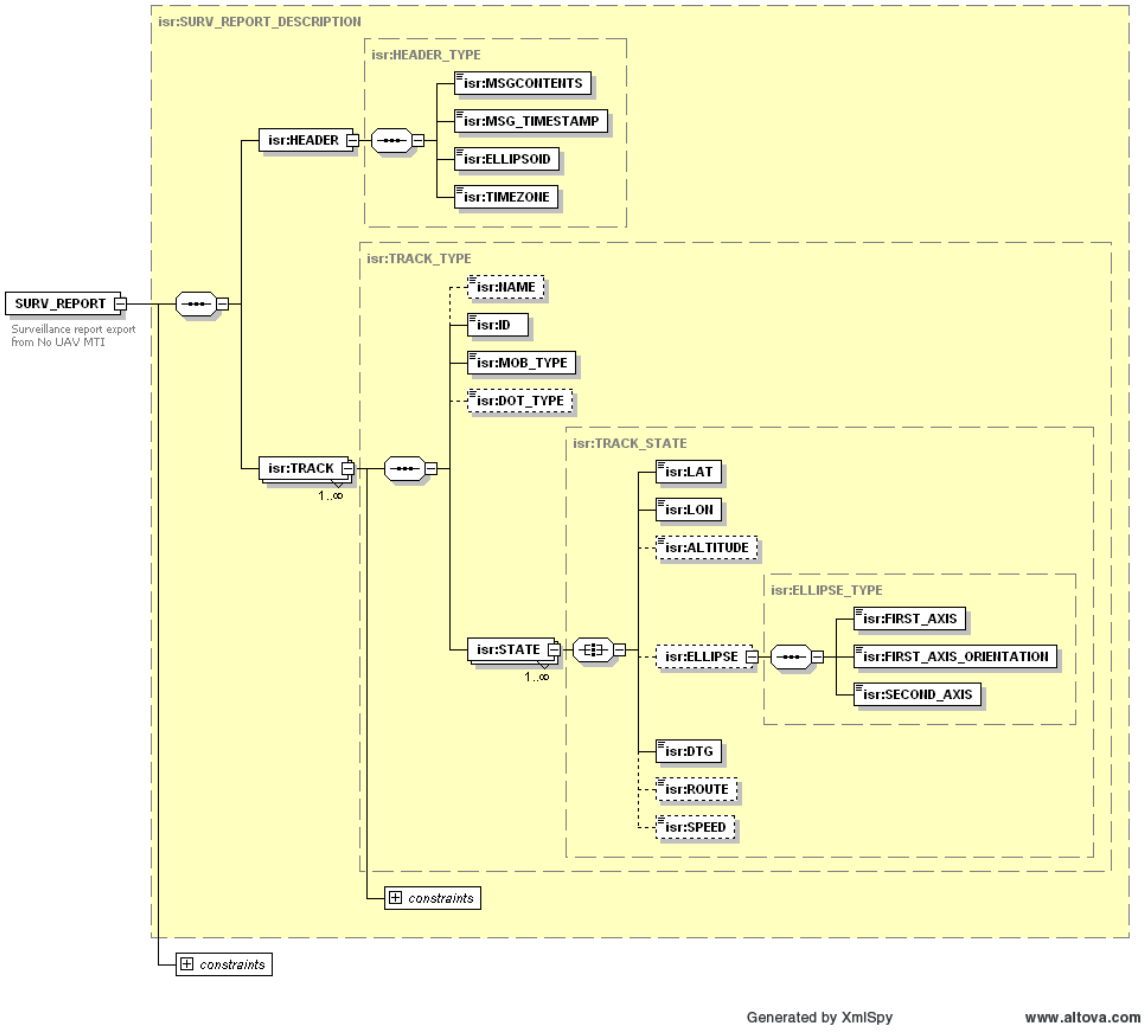

Figure A.5: MTI Tracks xml Schema.

A.5.2 Service Model

The Service Model describes the services provided by each nation and potentially used by others. Services are described by a set of attributes that give enough information about them so that potential users can select and invoke one.

Considering the studied scenario, services are of two kinds:

- publish-subscribe services. The publish-subscribe operations exposed by publishers and subscribers are described in the publish-subscribe section. Publishers are considered as services exposed by one nation to the others. As such, publishers are declared as services in the Service Registry (Subscribers are not). The model of Publishers as services is part of the detailed service model defined in the Service Registry specification.
- intelligence orientation services (one nation provides observation / intelligence service to others that can specify what kind of data or information they expect). The detailed service model is provided is the Service Registry section. The logical (business part of it is defined hereafter -> AcceptSensorRequest operation.

- ***SensorRequestResponse* AcceptSensorRequest(*SensorRequest*)**

AcceptSensorRequest (SensorRequest) allows an entity requiring sensor support to ask another entity providing such service to detail its specific needs. For the demonstration, it is proposed to define an SensorRequest as a geographical area (form to be specified in more details: geodesic "rectangle", polygon, other, … The SensorRequest is exemplified in the SensorRequest service, of which the detailed structure is defined in the Data Model section.

The Service provider receiving the call returns an SensorRequestResponse message through which it can either accept or reject the request. The detailed structure of the SensorRequestResponse is defined in the DataModel section. If the request is accepted then the service provider pushes these messages on a publish-subscribe Topic and provides the Topic name to the caller (that can subscribe to it), along with the expected date of first message publication on the Topic.

In the scenario, this operation is implemented by the Frigate/UAV Station and called by the Land Brigade.

A.5.3 Topics Model

Several Topics are defined to exchange operational data using the publish-subscribe mode. These Topics are:

- "**Land picture**" Topic: transports instances of Land Picture type according to the data format defined above.
- "**Maritime picture**" Topic: transports instances of Maritime Picture type according to the data format defined above.
- "**MTI Tracks**" Topic: transports instances of TypeTacticalPicutre according to the data format defined above.

A.6 SERVICE REGISTRY SPECIFICATIONS

This chapter describes the Service Registry and related specifications to be used in the NATO RTO/IST-061 CWID06 demonstration. It describes the architecture of the registry, in addition to what information to publish about each service and how to publish this information into the Service Registry to make the services searchable.

A.6.1 Architecture

The service discovery architecture will be based on UDDI (Universal Description, Discovery and Integration) and an abstraction layer in front of UDDI. This abstraction layer will handle some aspects of the service registry that are not satisfied by the UDDI registry, such as security, extended search capabilities and service termination policies.

Figure A.6: Service Discovery Architecture.

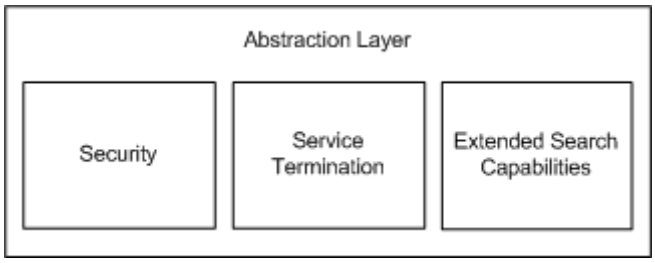

Figure A.7: Abstraction Layer.

The security and service termination components of the abstraction layer are REQUIRED, whereas the extended search capabilities component is OPTIONAL.

For demonstration purposes a central registry will be provided that all nations can use if they so choose, but each nation is free to provide their own service registry. Figure A.8 illustrates this choice. Here nation B provides a local service registry, whereas nation A does not.

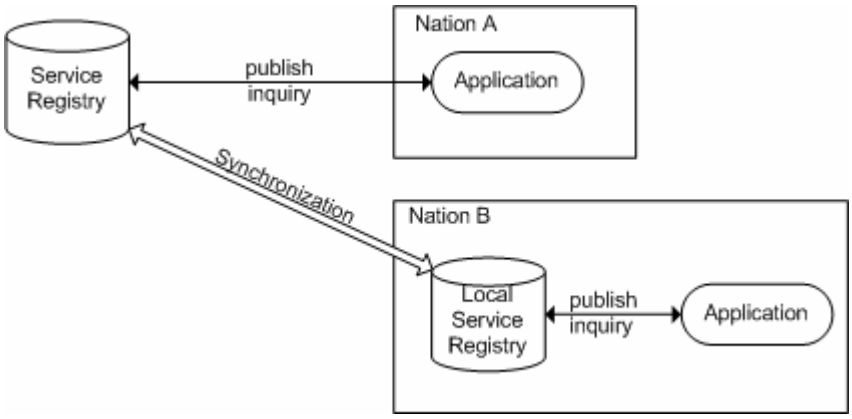

Figure A.8: Central and/or Local Service Registries.

If a nation provides a local service registry this has to be synchronized against the central one, how this is done is a national concern.

A.6.1.1 Which Version of UDDI to Use

UDDI V3 [[1]] will be used as the specification to build on regarding the service discovery architecture in the demonstrator. The UDDI V3 specification consists of several APIs; the Inquiry API set, the Publication API set, the Security API set, the Subscription API set, the Custody and ownership Transfer API set, the Value Set API set and the Replication API set. The service registry MUST at the minimum be compliant with the UDDI V3 Inquiry, Publication, and Security Policy API sets. The other API sets are OPTIONAL.

The differences of UDDI V3 versus UDDI V2 are briefly described in Appendix 1, Section A1.13.

A.6.1.2 Abstraction Layer Programmers APIs

The abstraction layer will implement the necessary UDDI V3 Programmers APIs. This includes the Inquiry API Set, Publication API Set and the Security Policy API Set. In addition the Publication API Set is extended with the special purpose calls of publishServices and resetRegistry. These are explained in Section A.6.7. Other UDDI V3 Programmers API's may be added if new requirements emerge.

Both success and error reporting will be compliant with the UDDI V3 specification. Aditional error reporting is necessary for security processing and some of the special purpose calls. These are defined as needed in this document.

A.6.1.3 Other Decisions

String values within UDDI SHALL NOT be case sensitive.

UDDI V3 records SHALL use keys of the format "uddi:<uuid-key>" (ref Appendix 1, Section A1.13).

A.6.2 Use of the UDDI Data Model

The metadata describing each service is important, because it is going to be used to locate relevant service descriptions, and if the information about each available service is not good enough, locating these descriptions will become hard. The metadata in question will be taxonomies used to classify the service descriptions, domain-specific attributes that may be of interest and service interfaces, message encodings

and transport protocols to use. An advantage on publishing information about service interfaces, message encodings and transport protocols in the service registry, is that it makes it easier to find compatible services and it facilitates run-time discovery of services. The technical note mentioned in Section A.6.7.1 provides the means to do this.

Another thing that is important is where to store the metadata concerning the services in the UDDI data model. It will be tModels that is used to represent taxonomies, domain-specific attributes, service interfaces, message encodings and transport protocols. References to tModels can be put on every entity within the UDDI data model (either within identifierBags, categoryBags or tModelInstanceDetails constructs). Domain-specific attributes or taxonomies used to describe a service will be placed as references to the corresponding tModels within the `categoryBag-element` of the `businessService`-element. Domain-specific attributes and taxonomies used to describe the organisation/unit/company/asset that publishes the service are placed as references to the corresponding tModels within the `categoryBag`-element in the `businessEntity`-element. Service interfaces, message encoding and protocols used to implement a web service will be placed as specified in the OASIS technical note [[2]].

A.6.2.1 The UDDI Data Model

The UDDI data model consists of four core entities, as seen in Figure A.9.

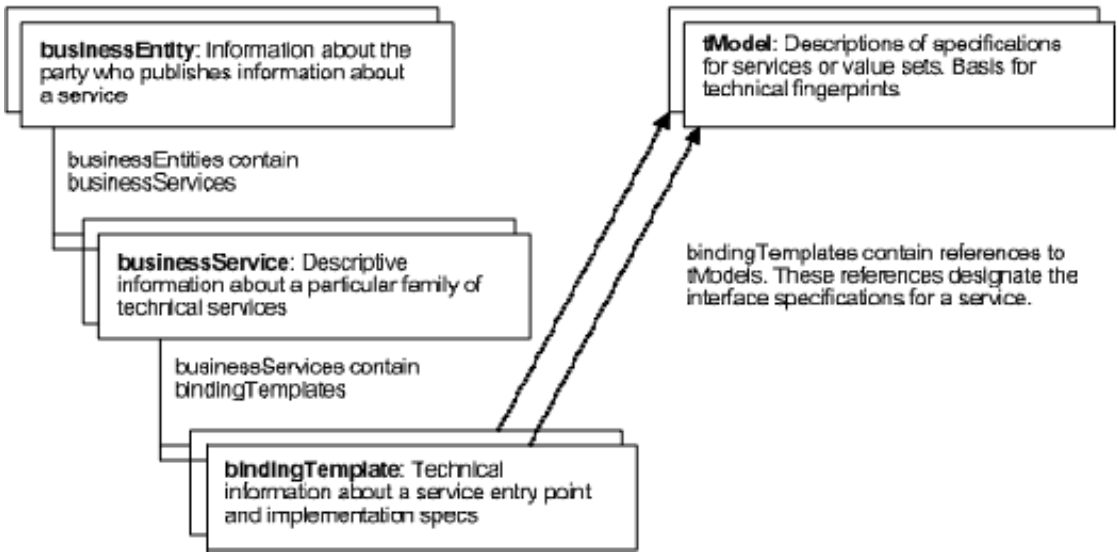

Figure A.9: UDDI Data Model.

The `businessEntity` contains descriptive information about the publisher of a service, for instance a company or an organisation. The `businessService` contains descriptive information about a logical service. Technical information about the businessService is found in the contained `bindingTemplate` entities. Each `bindingTemplate` entity describes an instance of a web service offered at a particular network address. It could for instance be that a logical service is implemented by several web services, each with its own message encoding and transport protocols so as to accommodate for a variety of clients, or it could be that a logical service consists of several web services that together make up the functionality offered by the logical service. The `tModel` entity is used to represent unique concepts or constructs in UDDI. This entity is used to describe web services in ways that are meaningful enough to be useful during searches, and to make these descriptions complete enough that people and programs can discover how to interact with web services they do not know much about.

The agreed set of metadata to be used and where it is placed within the UDDI data model will give directions on how to use the UDDI inquiry API to satisfy queries.

All information regarding security, i.e. signatures and labels, in this chapter is for the benefit of the implementers of the service registry. The publisher SHALL not use the dsig:Signature element or put a security label on any element, when publishing into the service registry. See Section A.6.6 for more information.

A.6.2.2 tModels

The tModel is an important entity within the UDDI data model. The structure of an UDDI `<tModel>` element is shown in Figure A.10.

```
<xsd:element name="tModel" type="uddi:tModel" final="restriction"/>
<xsd:complexType name="tModel" final="restriction">
  <xsd:sequence>
    <xsd:element ref="uddi:name"/>
    <xsd:element ref="uddi:description" minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element ref="uddi:overviewDoc" minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element ref="uddi:identifierBag" minOccurs="0"/>
    <xsd:element ref="uddi:categoryBag" minOccurs="0"/>
    <xsd:element ref="dsig:Signature" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="tModelKey" type="uddi:tModelKey" use="optional"/>
  <xsd:attribute name="deleted" type="uddi:deleted" use="optional" default="false"/>
</xsd:complexType>
```

Figure A.10: tModel-Structure in UDDI V3.

Several predefined `<tModel>` elements are given in Annex B.

A.6.3 What Metadata to Publish about Business Entities

The information about the organisation/asset/unit that publishes the service will be put into the businessEntity-structure in the UDDI registry. This structure is shown below in Figure A.11. Also shown are the sub structure contacts.

```
<xsd:element name="businessEntity" type="uddi:businessEntity" final="restriction"/>
<xsd:complexType name="businessEntity" final="restriction">
  <xsd:sequence>
    <xsd:element ref="uddi:discoveryURLs" minOccurs="0"/>
    <xsd:element ref="uddi:name" maxOccurs="unbounded"/>
    <xsd:element ref="uddi:description" minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element ref="uddi:contacts" minOccurs="0"/>
    <xsd:element ref="uddi:businessServices" minOccurs="0"/>
    <xsd:element ref="uddi:identifierBag" minOccurs="0"/>
    <xsd:element ref="uddi:categoryBag" minOccurs="0"/>
    <xsd:element ref="dsig:Signature" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="businessKey" type="uddi:businessKey" use="optional"/>
</xsd:complexType>

<xsd:element name="contacts" type="uddi:contacts" final="restriction"/>
<xsd:complexType name="contacts" final="restriction">
  <xsd:sequence>
    <xsd:element ref="uddi:contact" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:element name="contact" type="uddi:contact" final="restriction"/>
<xsd:complexType name="contact" final="restriction">
  <xsd:sequence>
    <xsd:element ref="uddi:description" minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element ref="uddi:personName" maxOccurs="unbounded"/>
    <xsd:element ref="uddi:phone" minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element ref="uddi:email" minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element ref="uddi:address" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="useType" type="uddi:useType" use="optional" default=""/>
</xsd:complexType>
```

Figure A.11: businessEntity-structure in UDDI V3.

Internally in the UDDI, uuid-keys with the prefix “uddi:” SHALL be used.

In the following the information regarding nations, assets and COIs (Community of Interest) will be described.

A.6.3.1 Nations

The `<businessEntity>` element which describes a nation is populated as follows:

- The `/businessEntity/discoveryURLs` element is not used
- The `/businessEntity/name` element MUST contain the name of the nation.
- The `/businessEntity/description` element contains a short description of the nation. This is OPTIONAL to provide.
- The `/businessEntity/contacts` element contains information about contact persons. At least one contact MUST be specified, specifying more than one contact is OPTIONAL.
- The `/businessEntity/businessServices` element contains services offered by the nation. It MAY be that a nation offers services indirectly via its assets and not directly from the nation businessEntity.
- The `/businessEntity/identifierBag` element contains an identification string, refer to Appendix 1, Section A1.1 for naming rules.
- The `/businessEntity/categoryBag` element contains keyedReferences referencing (using predefined RTG-027 tModels):
    - The *Entity Type tModel* and the value of the key is “`Nation`”.
    - The *General Keywords Category System tModel* where the value of the keyName is “`SecurityLabel`”and the value of the key is the appropriate label.

- The `/businessEntity/dsig:Signature` element contains the signature of the businessEntity description.
- The `/businessEntity/@businessKey` attribute is set to the key attributed to this `businessEntity` element.

In Figure A.12 the information that will be published regarding a Nation is illustrated.

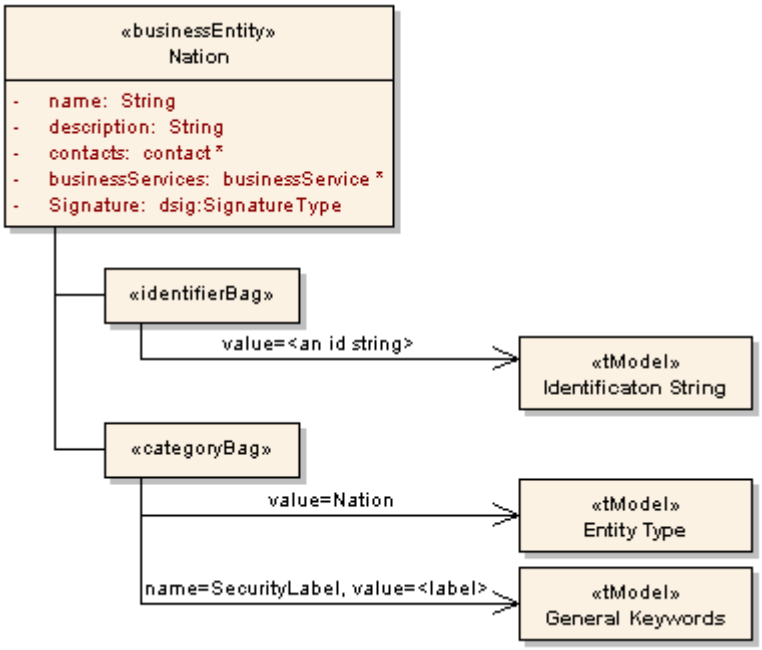

Figure A.12: Nation.

A nation contains assets; this relationship is captured using publisherAssertions as shown in Section A.6.3.4.

A.6.3.2 Assets

The `<businessEntity>` element which describes an asset is populated as follows:

- The `/businessEntity/discoveryURLs` element is not used
- The `/businessEntity/name` element MUST contain the name of the asset.
- The `/businessEntity/description` element contains a short description of the asset. This is REQUIRED to provide.
- The `/businessEntity/contacts` element contains information about contact persons. At least one contact MUST be specified, specifying more than one contact is OPTIONAL.
- The `/businessEntity/businessServices` element contains services offered by the asset.
- The `/businessEntity/identifierBag` element contains an identification string, refer to Appendix 1, Section A1.1 for naming rules.
- The `/businessEntity/categoryBag` element contains keyedReferences referencing (using predefined RTG-027 tModels):
    - The *General Keywords Category System tModel* where the value of the keyName is "`SecurityLabel`"and the value of the key is the appropriate label.
    - The *Entity Type tModel* and the value of the key is "`Asset`".
    - The *Asset Categorization tModel* and the value of the key is the type of asset this is.
- The `/businessEntity/dsig:Signature` element contains the signature of the businessEntity description.

- The `/businessEntity/@businessKey` attribute is set to the key attributed to this `businessEntity` element.

In Figure A.13 the information that will be published regarding an asset is illustrated.

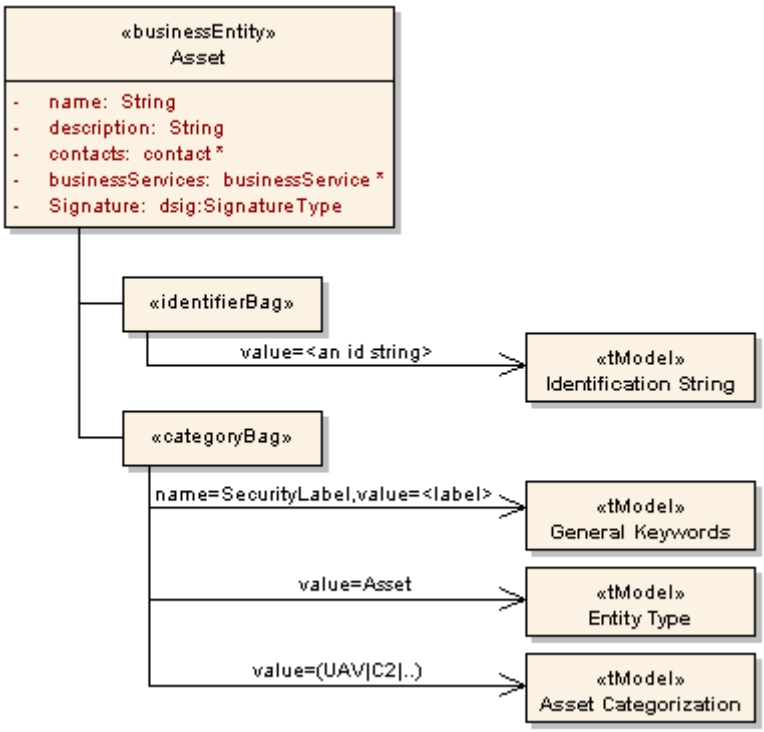

Figure A.13: Asset.

An asset belongs to a nation; this relationship is captured using publisherAssertions as shown in Section A.6.3.4. An asset is also related to a COI as shown in Section A.6.3.5.

A.6.3.3 COI

The `<businessEntity>` element which describes a COI is populated as follows:

- The `/businessEntity/discoveryURLs` element is not used
- The `/businessEntity/name` element MUST contain the name of the COI.
- The `/businessEntity/description` element contains a short description of the COI. This is REQUIRED to provide.
- The `/businessEntity/contacts` element is not used.
- The `/businessEntity/businessServices` element is not used.
- The `/businessEntity/identifierBag` element contains an identification string, refer to Appendix 1, Section A1.1 for naming rules.
- The `/businessEntity/categoryBag` element contains keyedReferences referencing (using predefined RTG-027 tModels):
    - The *General Keywords Category System tModel* where the value of the keyName is "`SecurityLabel`"and the value of the key is the appropriate label.
    - The *Entity Type tModel* and the value of the key is "`COI`".
- The `/businessEntity/dsig:Signature` element contains the signature of the businessEntity description.

- The `/businessEntity/@businessKey` attribute is set to the key attributed to this `businessEntity` element.

In Figure A.14 the information that will be published regarding a COI is illustrated.

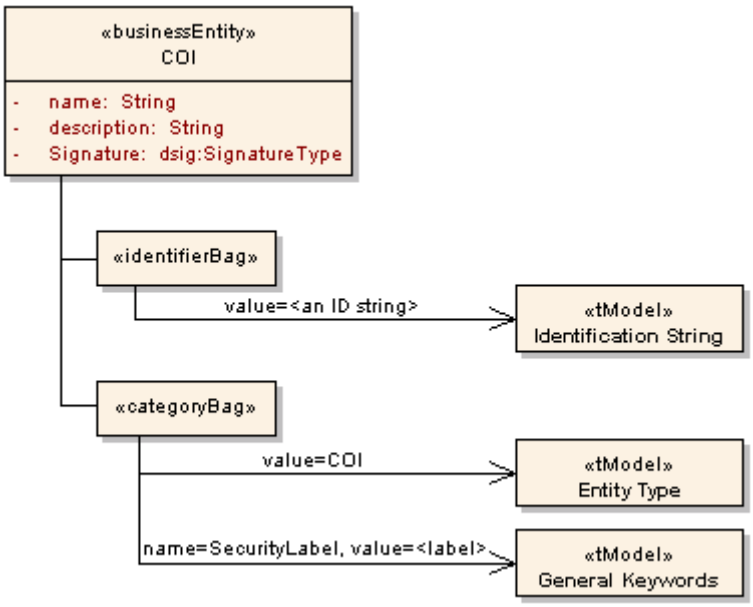

Figure A.14: COI.

A COI is related to an asset as shown in Section A.6.3.5.

A.6.3.4 Relationships between Nation and Assets

The information about the relationships between a nation and an asset will be put into a publisherAssertion-structure in the UDDI registry. This structure is shown below in Figure A.15.

```
<xsd:element name="publisherAssertion" type="uddi:publisherAssertion"
             final="restriction"/>
<xsd:complexType name="publisherAssertion" final="restriction">
  <xsd:sequence>
    <xsd:element ref="uddi:fromKey"/>
    <xsd:element ref="uddi:toKey"/>
    <xsd:element ref="uddi:keyedReference"/>
    <xsd:element ref="dsig:Signature" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:element name="fromKey" type="uddi:businessKey" final="restriction"/>
<xsd:element name="toKey" type="uddi:businessKey" final="restriction"/>

<xsd:element name="keyedReference" type="uddi:keyedReference" final="restriction"/>
<xsd:complexType name="keyedReference" final="restriction">
  <xsd:attribute name="tModelKey" type="uddi:tModelKey" use="required"/>
  <xsd:attribute name="keyName" type="uddi:keyName" use="optional" default=""/>
  <xsd:attribute name="keyValue" type="uddi:keyValue" use="required"/>
</xsd:complexType>
```

Figure A.15: publisherAssertion-structure in UDDI V3.

The `<publisherAssertion>` element is filled as follow:

- The `/publisherAssertion/fromKey` element contains the businessKey to the nation.
- The `/publisherAssertion/toKey` element contains the businessKey to the Asset.
- The `/publisherAssertion/keyedReference` element contains a reference to the *uddi-org:relationships tModel* and the value of the key is "`parent-child`".

In Figure A.16 the information that will be published regarding the relation between Nations and Assets are illustrated. Nations contains assets, and Assets belong to a Nation.

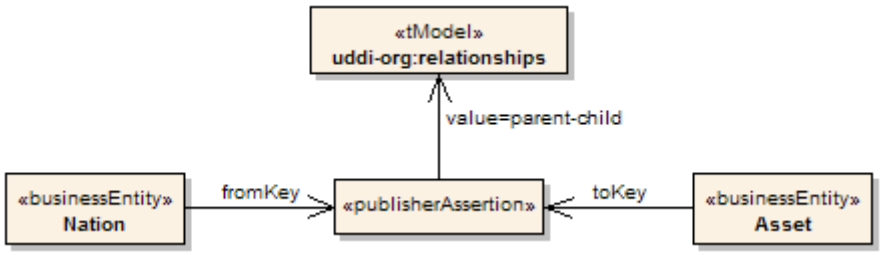

Figure A.16: Relation between Nations and Assets.

A.6.3.5 Relationships between Assets and COIs

This relationship is modeled in the same way as between Assets and Nations. A more sophisticated model – which is not realized in UDDI – is described in Section A.4.6.

A.6.3.6 Relationships between Different Assets

Since this relationship is not required for demonstration purposes it SHOULD NOT be used in the CWID 2006 demonstrator. If this functionality is wanted, this can be specified as an extension at a later stage.

A.6.4 What Metadata to Publish about Each Service

The metadata that describes each service is important, because it is going to be used to locate relevant service descriptions, and if the information about each available service is not good enough, locating these descriptions will become hard.

The metadata to publish about each service will be:

1. The name of the service
2. A short description of the service
3. Information about who published the service
4. The service type according to a service taxonomy
5. The coverage area of the service
6. The physical location of a service
7. What kind of information the service is providing
8. The topics the service is publishing to.
9. Various timestamps
    a. Published
    b. Valid until
10. Security metadata (see security in UDDI)
    a. Label
    b. Signature

11. Service interface descriptions
    a. see section about publishing WSDL in UDDI below.
12. The DistinguishedName (LDAP)

These metadata will be referred to as information items in the sub sections below.

A.6.4.1 Metadata about Services in the Registry

The information about services will be put into the businessService-structure (Figure A.17), the bindingTemplate-structure (Figure A.19) and the tModel-structure (Figure A.10) in the UDDI registry.

Figure A.17 shows the structure of the businessService-element in the UDDI registry.

```
<xsd:element name="businessService" type="uddi:businessService" final="restriction"/>
<xsd:complexType name="businessService" final="restriction">
  <xsd:sequence>
    <xsd:element ref="uddi:name" minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element ref="uddi:description" minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element ref="uddi:bindingTemplates" minOccurs="0"/>
    <xsd:element ref="uddi:categoryBag" minOccurs="0"/>
    <xsd:element ref="dsig:Signature" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="serviceKey" type="uddi:serviceKey" use="optional"/>
  <xsd:attribute name="businessKey" type="uddi:businessKey" use="optional"/>
</xsd:complexType>
```

Figure A.17: businessService-structure in UDDI V3.

The `<businessService>` element is filled as follows:

- The `/businessService/name` element contains information item 1.
- The `/businessService/description` element contains information item 2.
- The `/businessService/bindingTemplates` element contains bindingTemplates for this service.
- The `/businessService/categoryBag` element contains keyedReferences referencing (Information items 4-10, and partially information item 11):
    - The *Service Taxonomy tModel* and the value of the key is the type of service this service is.
    - The *General Keywords Category System tModel* where the value of the keyName attribute is "`SecurityLabel`" and the value of the key is the appropriate label.
    - The *Position tModel* and the value of the key is a URL-address specifying the location where one can retrieve an XML document containing the current position to the service.
    - The *Published tModel* and the value of the key is the date and time when the service was published into the service registry. The valid format for this is: YYYY-MM-DDThh:mm:ss±hh:mm.
    - The *Valid Until tModel* and the value of the key is the date and time when the service no longer will be available in the service registry The valid format for this is: YYYY-MM-DDThh:mm:ss±hh:mm.
    - The *WSDL Entity Type tModel* and the value of the key is "`service`".
    - The *XML Namespace tModel* and the value of the key is the namespace of the WSDL document containing the service. The value contained in the `/definitions/@targetNamespace` attribute in the WSDL document describing the service.
    - The *XML Local Name tModel* and the value of the key is the value of the attribute `/definitions/service/@name` in the WSDL document describing the service.
    - various topic tModels capturing the topics this service publishes to.

    - various Information Provided tModels capturing the kind of information this service provides.
- The `/businessService/categoryBag` element also contains keyedReferenceGroups referencing (Information item 5):
    - The *CoverageArea tModel* which specifies that the contained keyedReferences makes up a coverage area in the form of a rectangle. The contained keyReferences references:
        - The *Longitude tModel* where the value of the keyName attribute is "`upperLeft`" and the value of the keyValue attribute is the longitude of the upper left corner of the rectangle given in decimal degrees.
        - The *Latitude tModel* where the value of the keyName attribute is "`upperLeft`" and the value of the keyValue attribute is the latitude of the upper left corner of the rectangle given in decimal degrees.
        - The *Longitude tModel* where the value of the keyName attribute is "`lowerLeft`" and the value of the keyValue attribute is the longitude of the lower left corner of the rectangle given in decimal degrees.
        - The *Latitude tModel* where the value of the keyName attribute is "`lowerLeft`" and the value of the keyValue attribute is the latitude of the lower left corner of the rectangle given in decimal degrees.
- The `/businessService/dsig:Signature` element contains the signature of the service description.
- The `/businessService/@serviceKey` attribute is set to the key attributed to this `<businessService>` element.
- The `businessService/@businessKey` attribute is set to the businessKey of the containing businessEntity. This businessEntity will be a nation or an asset.

Figure A.18 illustrates how the information about each service will be published into the businessService-structure in the UDDI registry.

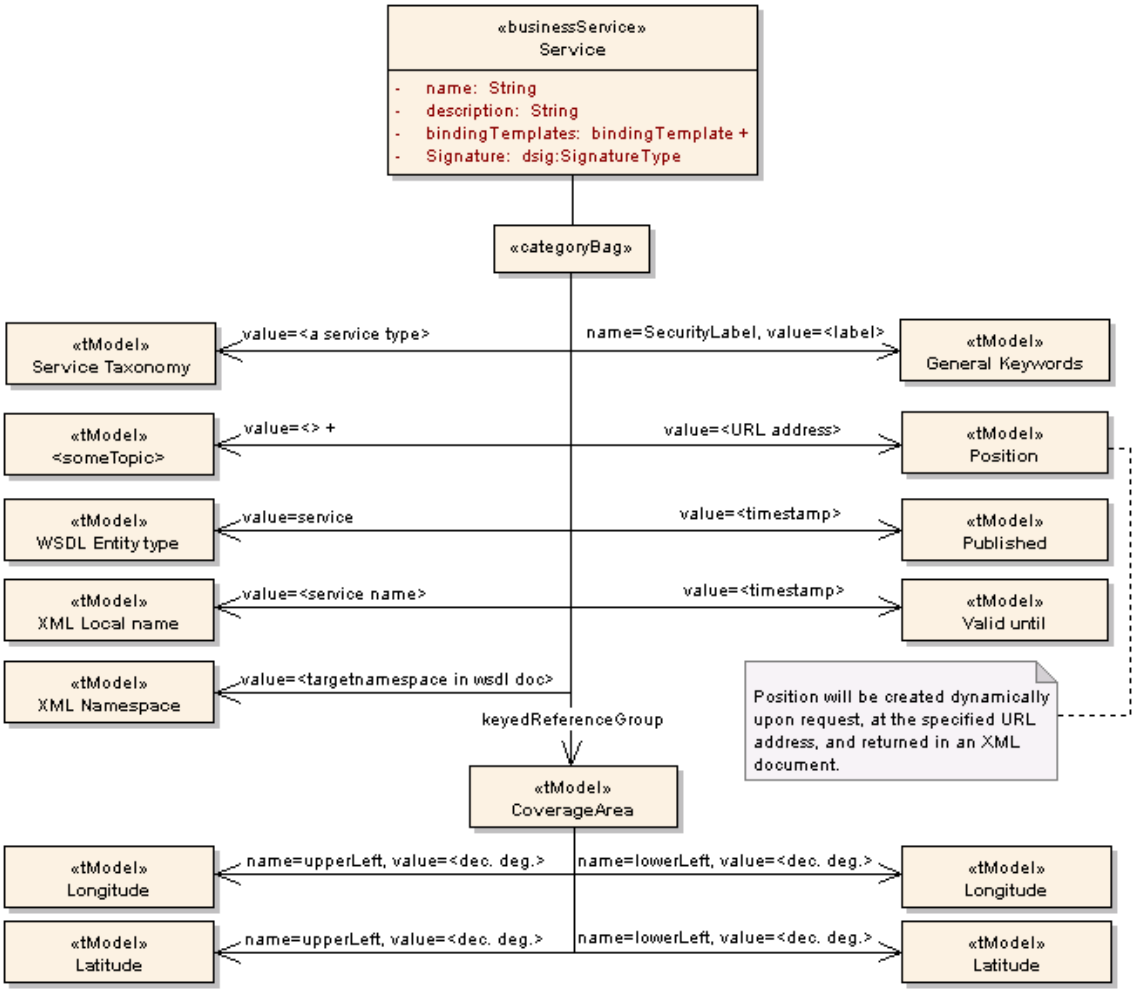

Figure A.18: Service information.

The General Keywords Category System tModel can be used to represent the metadata, or custom tModels can be created to represent a namespace/category. We see examples of both strategies in Figure A.18. The difference between the strategies is that during a search in the UDDI registry both the keyName- and the keyValue- attribute are relevant when searching for something described with the General Keywords tModel, whereas only the keyValue attribute is relevant in a search when using a custom tModel.

A.6.4.1.1 The bindingTemplate element

Figure A.19 shows the structure of the bindingTemplate-element in the UDDI registry. Also shown is the sub structure tModelInstanceDetails and its sub structures.

```
<xsd:element name="bindingTemplate" type="uddi:bindingTemplate" final="restriction"/>
<xsd:complexType name="bindingTemplate" final="restriction">
  <xsd:sequence>
    <xsd:element ref="uddi:description" minOccurs="0" maxOccurs="unbounded"/>
    <xsd:choice>
      <xsd:element ref="uddi:accessPoint"/>
      <xsd:element ref="uddi:hostingRedirector"/>
    </xsd:choice>
    <xsd:element ref="uddi:tModelInstanceDetails" minOccurs="0"/>
    <xsd:element ref="uddi:categoryBag" minOccurs="0"/>
    <xsd:element ref="dsig:Signature" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="bindingKey" type="uddi:bindingKey" use="optional"/>
  <xsd:attribute name="serviceKey" type="uddi:serviceKey" use="optional"/>
</xsd:complexType>


<xsd:element name="tModelInstanceDetails" type="uddi:tModelInstanceDetails"
             final="restriction"/>
<xsd:complexType name="tModelInstanceDetails" final="restriction">
  <xsd:sequence>
    <xsd:element ref="uddi:tModelInstanceInfo" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>


<xsd:element name="tModelInstanceInfo" type="uddi:tModelInstanceInfo"
             final="restriction"/>
<xsd:complexType name="tModelInstanceInfo" final="restriction">
  <xsd:sequence>
    <xsd:element ref="uddi:description" minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element ref="uddi:instanceDetails" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="tModelKey" type="uddi:tModelKey" use="required"/>
</xsd:complexType>


<xsd:element name="instanceDetails" type="uddi:instanceDetails" final="restriction"/>
<xsd:complexType name="instanceDetails" final="restriction">
  <xsd:sequence>
    <xsd:element ref="uddi:description" minOccurs="0" maxOccurs="unbounded"/>
    <xsd:choice>
      <xsd:sequence>
        <xsd:element ref="uddi:overviewDoc" maxOccurs="unbounded"/>
        <xsd:element ref="uddi:instanceParms" minOccurs="0"/>
      </xsd:sequence>
      <xsd:element ref="uddi:instanceParms"/>
    </xsd:choice>
  </xsd:sequence>
</xsd:complexType>
```

Figure A.19: bindingTemplate-structure in UDDI V3.

The `<bindingTemplate>` element is populated as follows (some of this information is related to information item 11):

- The `bindingTemplate/description` element is not used.
- The `bindingTemplate/accesspoint` element has the value of the attribute `definitions/service/port/soap:address/@location` in the WSDL document describing the service. The `bindingTemplate/accesspoint/@useType` attribute has the value "`http`".
- The `bindingTemplate/hostingRedirector` is not used
- The `bindingTemplate/tModelInstanceDetails` element is populated with two `tModelInstanceInfo` elements (it can contain more `tModelInstanceInfo`-elements if it implements more wsdl-bindings and wsdl-portTypes):
    - The first tModelInstanceInfo element:
        - The `tModelInstanceInfo/description` element is not used.
        - The `tModelInstanceInfo/instanceDetails/description` is not used.

        - The `tModelInstanceInfo/instanceDetails/overviewDoc` is not used.
        - the `tModelInstanceInfo/instanceDetalis/instanceParms` element has the value of the attribute `definitions/service/port/@name` in the WSDL document to the service
        - The `tModelInstanceInfo/@tModelKey` attribute refers to the tModel that implements the binding for this port.
    - The second tModelInstanceInfo element
        - The `tModelInstanceInfo/description` element is not used.
        - The `tModelInstanceInfo/instanceDetails` is not used.
        - The `tModelInstanceInfo/@tModelKey` attribute refers to the tModel that implements the portType for this port's binding.
- The `bindingTemplate/categoryBag` contains keyedReferences referencing:
    - The *General Keywords Category System tModel* where the value of the keyName attribute is "`SecurityLabel`" and the value of the key is the appropriate label.
    - The *Distinguished Name tModel* where the value of the keyName attribute is "DistinguishedName" and the value of the key is the LDAP distinguished name associated with this service.
- The `bindingTemplate/dsig:Signature` contains the signature of the binding information.
- The `bindingTemplate/@sbindingKey` attribute is set to the key attributed to this `<bindingTemplate>` element.
- The `bindingTemplate/@serviceKey` attribute is set to the serviceKey of the containing businessService.

Figure A.20 illustrates how the information about each service will be published into the bindingTemplate-structure in the UDDI registry.

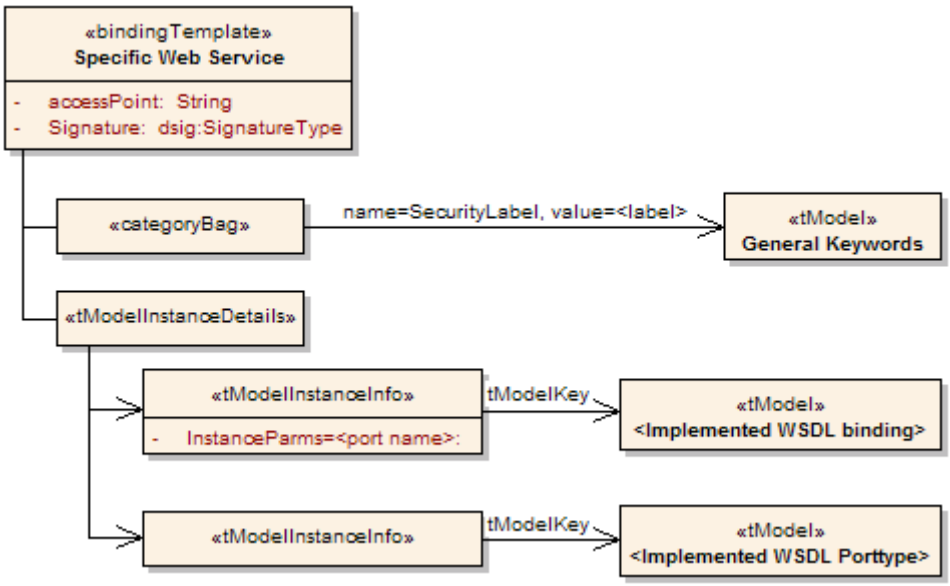

Figure A.20: Service Binding Information.

A.6.4.2 Publishing WSDL Information in UDDI

OASIS has published a Technical note that describes how to publish WSDL information in UDDI [[2]] to make it searchable. To be compliant with the service registry this specification must be followed. That

means that the tModels described in Appendix 2 in this technical note must be published and present in the UDDI registry, to facilitate the mapping of WSDL entities into the UDDI registry.

The tModels from Annex B in [[2]] are:

1. WSDL Entity Type tModel
2. XML Namespace tModel
3. XML Local Name tModel
4. WSDL portType reference tModel
5. SOAP Protocol tModel
6. HTTP Protocol tModel
7. Protocol Categorization
8. Transport Categorization
9. WSDL Address tModel

Additional tModels must be present if for instance other protocols are to be used (XMPP instead of HTTP). If HTTP is going to be used as the transport protocol, the HTTP Transport tModel must be published in the UDDI registry. This tModel is described in chapter 11.3.3 in [[1]].

The actual mapping of a WSDL document to UDDI using these tModels is described in more detail in the technical note from OASIS [[2]]. Figure A.21 and Figure A.22 illustrates this mapping by respectively showing the structure of a WSDL document and the mapping from WSDL to UDDI.

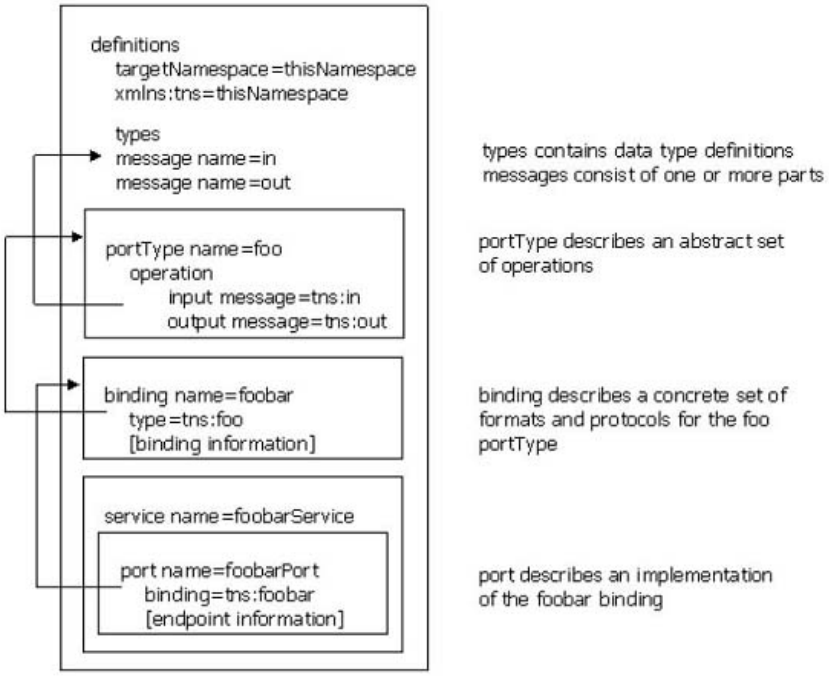

Figure A.21: Components in a WSDL Document.

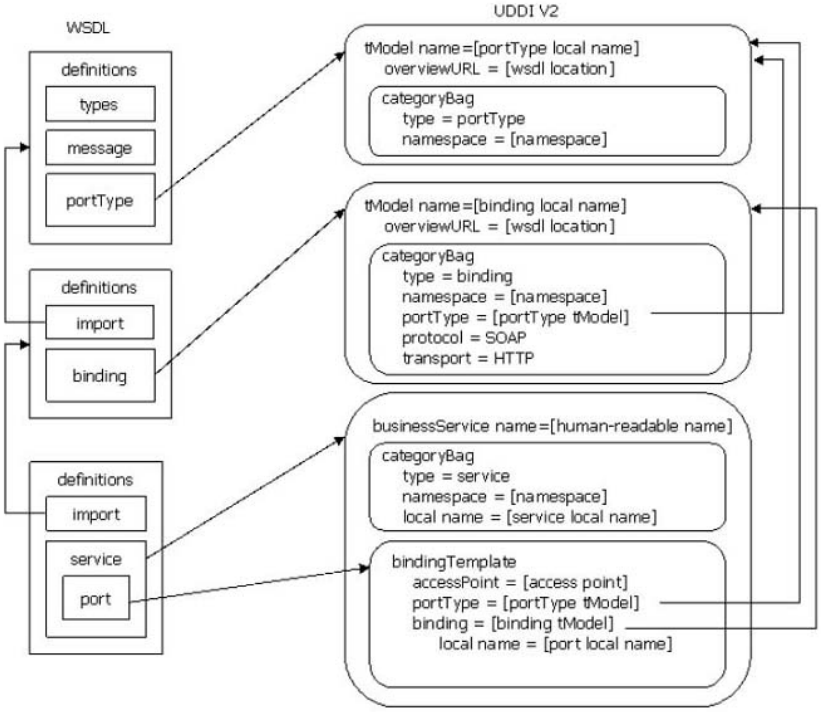

Figure A.22: Mapping a WSDL Document into UDDI Structures.

As shown in the two figures above the information that must be extracted from the WSDL document is:

- the name and targetNamespace attributes of the portType.
- the name and targetNamespace attributes of the binding.
- information about transports protocols.
- the name attribute and targetNamespace of the service.
- the name attribute of the port.
- the location attribute from the soap:address tag.

From this information; two tModels, a businessService and a bindingTemplate are created to map the WSDL document:

- a *tModel* that maps the portType section.
- a *tModel* that maps the binding section.
- a *businessService* that maps the service section.
- a *bindingTemplate* that maps the port section.

A.6.4.2.1 The portType tModel

The portType tModel represents the interface of the service and information in the portType section is mapped to this tModel:

- The `/tModel/name` element contains the value of the `/definitions/portType/@name` attribute in the WSDL document.
- The `/tModel/description` element is not used.
- The `/tModel/overviewDoc/description` element is not used.
- The `/tModel/overviewDoc/overviewURL` element contains the address to the corresponding WSDL document.
- The `/tModel/identifierBag` element is not used.

- The `/tModel/categoryBag` element contains keyedReferences referencing:
  - the *WSDL Entity Type tModel* and the value of the key is `portType`.
  - the *XML Namespace tModel* and the value of the key is the namespace of the WSDL document containing the portType.
  - The *General Keywords Category System tModel* where the value of the keyName attribute is "`SecurityLabel`" and the value of the key is the appropriate label (*Not a part of the OASIS technical note, but an addition by the RTG-027/IST-061 group*).
- The `/tModel/dsig:Signature` contains the signature of the tModel information.
- The `<tModel>/@<tModelKey>` attribute is set to the key attributed to this `<tModel>` element.
- The `<tModel>/@<deleted>` attribute is not used.

A.6.4.2.2 The Binding tModel

The binding tModel captures the transports and encodings information from the binding section of the WSDL Document:

- The `/tModel/name` element contains the value of the `/definitions/binding/@name` attribute in the WSDL document.
- The `/tModel/description` element is not used.
- The `/tModel/overviewDoc/description` element is not used.
- The `/tModel/overviewDoc/overviewURL` links to the WSDL Document.
- The `/tModel/identifierBag` element is not used.
- The `/tModel/categoryBag element` contains keyedReferences referencing:
  - the *WSDL Entity Type tModel* and the value of the key is "`binding`"
  - the *XML Namespace tModel* and the value of the key is the namespace of the WSDL document containing the binding.
  - The *WSDL portType reference tModel* and the value of the key is the tModelKey of the portType this binding implements.
  - the *UDDI Type Category System* and the value of the key is "`wsdlSpec`".
  - various protocol tModels capturing what kind of protocols that are used.
  - The *General Keywords Category System tModel* where the value of the keyName attribute is "`SecurityLabel`" and the value of the key is the appropriate label (*Not a part of the OASIS technical note, but an addition by the RTG-027/IST-061 group*).
- The `/tModel/dsig:Signature` contains the signature of the tModel information.
- The `<tModel>/@<tModelKey>` attribute is set to the key attributed to this `<tModel>` element.
- The `<tModel>/@<deleted>` attribute is not used.

A.6.4.2.3 The businessService and bindingTemplate

The information shown here in the businessService and the bindingTemplate elements do only reflect the information mapped from the WSDL document describing the service. For a complete description of what should be present in a businessService and a bindingTemplate element see Section A.6.4.1.

The businessService maps the service section of the WSDL document:

- The `categoryBag` of the businessService contains keyedReferences referencing:
  - the *WSDL Entity Type tModel* and the value of the key is "`service`".
  - the *XML Namespace tModel* and the value of the key is the namespace of the WSDL document containing the service.
  - the *XML Local Name tModel* and the value of the key is the value of the attribute `definitions/service/@name` in the WSDL document.

The bindingTemplate maps the port of the WSDL document:

- The `accessPoint` element has the value of the attribute `definitions/service/port/soap:address/@location` in the WSDL document.
- Two tModelInstanceInfo elements is created:
  - The first tModelInstanceInfo element
    - the `@tModelKey` attribute refers to the tModel that implements the binding for this port.
    - the `instanceDetalis/instanceParms` element has the value of the attribute `definitions/service/port/@name` in the WSDL document
  - The second tModelInstanceInfo element
    - the `@tModelKey` attribute refers to the tModel that implements the portType for this port's binding.

A.6.4.2.4 An Example

To make things clearer an example is provided that show how this mapping is done in practice. For detailed explanation of the mapping, see [[2]]. The WSDL document used as the basis for the example is one for a fictitious weather service for the area around Lakselv (see Figure A.23).

```
<?xml version="1.0" encoding="utf-8"?>
<definitions name="Weather"
             targetNamespace="http://test.com/weather/"
             xmlns:tns="http://test.com/weather/"
             xmlns:xsd1="http://test.com/weather/schema/"
             xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
             xmlns="http://schemas.xmlsoap.org/wsdl/">
  <types>
    <xs:schema targetNamespace="http://test.com/weather/schema/"
               xmlns:xs="http://www.w3.org/2001/XMLSchema">
      <xs:element name="weatherRequest" type="xs:string"/>
      <xs:element name="weatherInformation" type="xs:string"/>
    </xs:schema>
  </types>

  <message name="GetWeatherRequest">
    <part name="body" element="xsd1:weatherRequest"/>
  </message>
  <message name="WeatherInformation">
    <part name="body" element="xsd1:weatherInformation"/>
  </message>

  <portType name="WeatherPortType">
    <operation name="GetWeatherInformation">
      <input message="tns:GetWeatherRequest"/>
      <output message="tns:WeatherInformation"/>
    </operation>
  </portType>

  <binding name="WeatherSoapBinding" type="tns:WeatherPortType">
    <soap:binding style="document"
                  transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="GetWeatherInformation">
      <soap:operation soapAction="http://test.com/GetWeatherInformation"/>
      <input><soap:body use="literal"/></input>
      <output><soap:body use="literal"/></output>
    </operation>
  </binding>

  <service name="WeatherService">
    <port name="WeatherPort" binding="tns:WeatherSoapBinding">
      <soap:address location="http://accesspoint.of.service"/>
    </port>
  </service>

</definitions>
```

Maps to a tModel

Maps to a tModel

Maps to a business-Service and a binding-Template.

Figure A.23: A sample WSDL document

There are four parts of the WSDL document above that are going to be mapped over to UDDI structures. The elements in question are the portType, the binding, the service and the port elements. As shown in Figure A.22 and Figure A.23 they are respectively mapped to a tModel, a tModel, a businessService and a bindingTemplate.

Below in Figure A.24, Figure A.25 and Figure A.26 this mapping is done. The tModels and the businessService contains references to the tModels listed above in Section A.6.4.2 as keyedReferences in their categoryBag. At the right side of each figure it is indicated which tModel that is being referenced in the categoryBag.

```
<tModel tModelKey="uddi:e8cf1163-8234-4b35-865f-94a7322e40c3">
  <name>WeatherPortType</name>
  <overviewDoc>
    <overviewURL>http://location.of.wsdl.document</overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference tModelKey="uddi:uddi.org:xml:namespace"
                    keyName="portType namespace"
                    keyValue="http://test.com/weather/"/>
    <keyedReference tModelKey="uddi:uddi.org:wsdl:types"
                    keyName="WSDL type"
                    keyValue="portType"/>
  </categoryBag>
</tModel>
```

XML Name-space tModel

WSDL Entity Type tModel

Figure A.24: tModel Representing the portType.

```
<tModel tModelKey="uddi:49662926-f4a5-4ba5-b8d0-32ab388dadda">
  <name>WeatherSoapBinding</name>
  <overviewDoc>
    <overviewURL>http://location.of.wsdl.document</overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference tModelKey="uddi:uddi.org:xml:namespace"
                    keyName="binding namespace"
                    keyValue="http://test.com/weather/"/>
    <keyedReference tModelKey="uddi:uddi.org:wsdl:types"
                    keyName="WSDL type"
                    keyValue="binding"/>
    <keyedReference tModelKey="uddi:uddi.org:wsdl:porttypereference"
                    keyName="portType reference"
                    keyValue="uddi:e8cf1163-8234-4b35-865f-94a7322e40c3"/>
    <keyedReference tModelKey="uddi:uddi.org:wsdl:categorization:protocol"
                    keyName="SOAP protocol"
                    keyValue="uddi:uddi.org:protocol:soap"/>
    <keyedReference tModelKey="uddi:uddi.org:wsdl:categorization:transport"
                    keyName="HTTP transport"
                    keyValue="uddi:uddi.org:transport:http"/>
    <keyedReference tModelKey="uddi:uddi.org:categorization:types"
                    keyName="uddi-org:types"
                    keyValue="wsdlSpec"/>
  </categoryBag>
</tModel>
```

XML Namespace tModel

WSDL Entity Type tModel

WSDL Portype reference tModel

SOAP Protocol tModel

HTTP Transport tModel

UDDI Types Category System

Figure A.25: tModel Representing the Binding.

Mapping of port-element in the WSDL

```
<businessService serviceKey="uddi:102b114a-52e0-4af4-a292-02700da543d4"
                 businessKey="uddi:1e65ea29-4e0f-4807-8098-d352d7b10368">
  <name>Weather service in Lakselv</name>
  <bindingTemplates>
    <bindingTemplate bindingKey="uddi:f793c521-0daf-434c-8700-0e32da232e74"
                     serviceKey="uddi:102b114a-52e0-4af4-a292-02700da543d4">
      <accessPoint useType="http">http://accesspoint.of.service</accessPoint>
      <tModelInstanceDetails>
        <tModelInstanceInfo tModelKey="uddi:49662926-f4a5-4ba5-b8d0-32ab388dadda">
          <description xml:lang="en">wsdl:binding that this wsdl:port implements.
                                     The instanceParms specifies the port local name.
          </description>
          <instanceDetails>
            <instanceParms>WeatherPort</instanceParms>
          </instanceDetails>
        </tModelInstanceInfo>
        <tModelInstanceInfo tModelKey="uddi:e8cf1163-8234-4b35-865f-94a7322e40c3">
          <description xml:lang="en">wsdl:portType that this wsdl:port implements.
          </description>
        </tModelInstanceInfo>
      </tModelInstanceDetails>
    </bindingTemplate>
  </bindingTemplates>
  <categoryBag>
    <keyedReference tModelKey="uddi:uddi.org:wsdl:types"
                    keyName="WSDL type"
                    keyValue="service"/>
    <keyedReference tModelKey="uddi:uddi.org:xml:namespace"
                    keyName="service namespace"
                    keyValue="http://test.com/weather/"/>
    <keyedReference tModelKey="uddi:uddi.org:xml:localname"
                    keyName="service local name"
                    keyValue="WeatherService"/>
  </categoryBag>
</businessService>
```

Reference to the tModel for the implemented wsdl binding.

Reference to the tModel for the implemented porttype

WSDL Entity Type tModel

XML Name-space tModel

XML Local Name tModel

Figure A.26: businessService and bindingTemplate Representing the Service and Port Elements.

For each of the services published in the UDDI registry this information about their WSDL description has to be registered in the UDDI registry.

A.6.5 Modelling of Topics in UDDI

In the same manner the UDDI registry provides links to the WSDL schemas of the services, the UDDI registry can provide links to the XML schemas of the topics produced for all the COIs.

Thus, it becomes possible for an application to retrieve (to discover) the XML schema corresponding to a topic and to consider advanced computing, e.g. building of XSL style sheets to map the topic flow generated/received for a COI to internal representations.

tModels will be used to model topics and topic spaces, with some canonical tModels (see Appendix 1) to categorize them. References to topic-tModels can be put in the categoryBag of the businessService-elements in the UDDI data model, to indicate that that service provides information according to specific topics.

In Figure A.27 an example is shown about what information to register about topics and topic spaces.

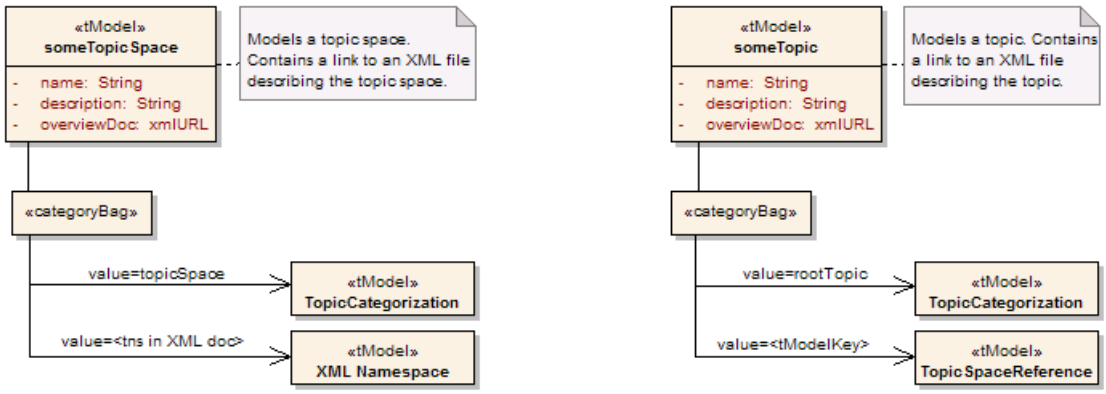

Figure A.27: Modelling of Topics in UDDI.

The *someTopicSpace*-tModel represents a topic space. Each topic space is given a name, a description, and a link to an XML file that describes the topic space completely. In addition it will be classified according to a topicCategorization scheme with the value "topicSpace", and it will be given an XML namespace that scopes all contained topics.

The *someTopic*-tModel represents a topic. Each topic is given a name, a description and a link to an XML file that describes the topic completely. In addition it will be classified according to a topicCategorization scheme with the value "rootTopic", and it will be given a reference to the topic space it belongs to.

A.6.5.1 How to Publish this Information into the Service Registry

The structure of a tModel is given in Figure A.10. The predefined tModels used to categorize a topic and a topic space is given in Annex B.

A.6.5.1.1 Topic Space

The `<tModel>` element which describes a topicSpace must be populated as follow:

- The `/tModel/name` element is set to the string "*%s1*" with %s1 set to the name of the topic space.
- The `/tModel/description` element is set to the string "*Topic space for the %s1 namespace*" with %s1 set to the namespace of the topic space.
- The `/tModel/overviewDoc/description` element is set to the string "XML Schema for the topic space: %s1", where %s1 is set to the name of the topic space.
- The `/tModel/overviewDoc/overviewURL` element is set to the path which contains the XML schema corresponding to the topic.
- The `/tModel/identifierBag` element is empty.
- The `/tModel/categoryBag` element contains keyedReferences referencing:
  - the *Topic Categorization tModel* and the value of the key is "`topicSpace`"
  - the *XML Namespace tModel* and the value of the key is the namespace of topic space.
  - The *General Keywords Category System tModel* where the value of the keyName attribute is "`SecurityLabel`" and the value of the key is the appropriate label.
- The /`tModel/dsig:Signature` contains the signature of the tModel information.
- The `<tModel>/@<tModelKey>` attribute is set to the key attributed to this `<tModel>` element.
- The `<tModel>/@<deleted>` attribute is not used.

An example of this structure is shown in Figure A.28, where a tModel referencing the XML document of the topic space for the ISR XML namespace is illustrated.

```
<tModel tModelKey="uddi:111111-6C7F-11DA-B1A0-FECD280E8AB2">
  <name>ISR Topic space</name>
  <description xml:lang="en">Topic space for the ISR namespace</description>
  <overviewDoc>
    <description xml:lang="en">
      XML Document for the topic space: ISR Topic space
    </description>
    <overviewURL>http://url.to.xml.document.for.the.topic.space</overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference tModelKey="uddi:A41915D0-4C6B-11DA-95D0-BD45C7FAF23E"
                    keyName="TopicCategorization"
                    keyValue="TopicSpace" />
    <keyedReference tModelKey="uddi:uddi.org:xml:namespace"
                    keyName="topic space namespace"
                    keyValue="ISR" />
    <keyedReference tModelKey="uddi:uddi.org:categorization:general_keywords"
                    keyName="SecurityLabel"
                    keyValue="NATO Secret" />
  </categoryBag>
  <dsig:Signature>---some signature---</dsig:Signature>
</tModel>
```

Figure A.28: Description of a Topic Space using a <tModel> Element.

A.6.5.1.2 Topic

The `<tModel>` element which describes a topic must be populated as follows:

1. The `/tModel/name` element is set to the string "*%s1:%s2*" with %s1 set to the namespace of the topic and %s2 set to the name of the topic.
2. The `/tModel/description` element is set to the string "*XML schema of the topic: %s1:%s2*" with %s1 set to the namespace of the topic and %s2 set to the name of the topic
3. The `tModel/overviewDoc/description` element is set to the same string as the `/tModel/description` element.
4. The `/tModel/overviewDoc/overviewURL` element is set to the path which contains the XML schema corresponding to the topic.
5. The `/tModel/identifierBag` element is empty.
6. The `/tModel/categoryBag` element contains keyedReferences referencing:
   - the *Topic Categorization tModel* and the value of the key is "`rootTopic`"
   - the *Topic Space reference tModel* and the value of the key is the tModelKey of the topic space that this topic belongs to.
   - The *General Keywords Category System tModel* where the value of the keyName attribute is "`SecurityLabel`" and the value of the key is the appropriate label.

- The `/tModel/dsig:Signature` contains the signature of the tModel information.
- The `<tModel>/@<tModelKey>` attribute is set to the key attributed to this `<tModel>` element.
- The `<tModel>/@<deleted>` attribute is not used.

And example of this structure is shown in Figure A.29, where a tModel referencing the XML schema of the Track topic for the ISR XML namespace is illustrated.

```xml
<tModel tModelKey="uddi:667E71A0-6C7F-11DA-B1A0-FECD280E8AB2">
  <name>ISR:Track</name>
  <description xml:lang="en">XML Schema of the topic: ISR:Track</description>
  <overviewDoc>
    <description xml:lang="en">XML Schema of the topic: ISR: Track</description>
    <overviewURL>http://url.of.the.xsd.document.describing.the.Track.topic
    </overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference tModelKey="uddi:A41915D0-4C6B-11DA-95D0-BD45C7FAF23E"
                    keyName="TopicCategorization"
                    keyValue="RootTopic" />
    <keyedReference tModelKey="uddi:A63DCA90-4C6B-11DA-8A90-FEE8FA5DB743"
                    keyName="TopicSpaceReference"
                    keyValue="uuid:11111111-6C7F-11DA-B1A0-FECD280E8AB2" />
    <keyedReference tModelKey="uddi:uddi.org:categorization:general_keywords"
                    keyName="SecurityLabel"
                    keyValue="NATO Secret" />
  </categoryBag>
  <dsig:Signature>---some signature---</dsig:Signature>
</tModel>
```

Figure A.29: Description of the Topic itself using a <tModel> Element.

A.6.6 Security in UDDI

The information stored in the service registry regarding security will be a label and a signature. This information is not something a publisher will have to provide, this will solely be the responsibility of the abstraction layer to do. The responsibility of the publisher regarding security will be to provide a label in the SOAP-header, and to sign the SOAP message that is sent to the service registry. It is important that the SOAP-label reflects the real security level of the UDDI information.

The label information will be put by the abstraction layer in the `categoryBag` on the businessEntity-, the businessService-, the bindingTemplate- and the tModel-elements as a `keyedReference` that references the General Keyword Category System tModel in the `tModelKey` attribute. The `keyName`-attribute will contain the value "`SecurityLabel`" and the `keyValue` will contain the corresponding value.

The signature information will be stored in the dsig:Signature element in the businessEntity-, the businessService-, the bindingTemplate- and the tModel-elements.

For more information on the security in UDDI please refer to Section A.7.7.

A sample of the different entities with a security label specified is shown below:

```
<businessEntity
      businessKey="uddi:74DC9A70-1620-11DA-A75E-D5F9C7FC0CA9">
  <name>Lakselv weather organization</name>
  <description>Provides weather information from Lakselv.</description>
  <contacts>
    <contact useType="Administrator">
      <personName>Anders Langmyr</personName>
      <phone>+4763807796</phone>
      <email>Anders.Langmyr@ffi.no</email>
    </contact>
  </contacts>
  <categoryBag>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:general_keywords"
        keyName="SecurityLabel"
        keyValue="The value of the label"/>
  </categoryBag>
<dsig:Signature>----Some signature---</dsig:Signature>
</businessEntity>
```

Figure A.30: businessEntity.

```
<businessService
      businessKey="uddi:74DC9A70-1620-11DA-A75E-D5F9C7FC0CA9"
      serviceKey="uddi:F5BFDFF0-1623-11DA-A75E-B9D6F00D57AC">
  <name>Lakselv weather service</name>
  <description>Weather updates from the north.</description>
  <categoryBag>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:general_keywords"
        keyName="SecurityLabel"
        keyValue="The value of the label"/>
  </categoryBag>
  <dsig:Signature>----Some signature---</dsig:Signature>
</businessService>
```

Figure A.31: businessService.

```
<bindingTemplate bindingKey="uddi:f793c521-0daf-434c-8700-0e32da232e74"
                 serviceKey="uddi:F5BFDFF0-1623-11DA-A75E-B9D6F00D57AC">
  <accessPoint URLType="http">http://accesspoint.of.service</accessPoint>
  <tModelInstanceDetails>
    <tModelInstanceInfo tModelKey="uddi:e8cf1163-8234-4b35-865f-94a7322e40c3" />
  </tModelInstanceDetails>
  <categoryBag>
    <keyedReference tModelKey="uddi:uddi.org:categorization:general_keywords"
                    keyName="SecurityLabel"
                    keyValue="The value of the label"/>
  </categoryBag>
  <dsig:Signature>----Some signature---</dsig:Signature>
</bindingTemplate>
```

Figure A.32: bindingTemplate.

```
<tModel
      tModelKey="uddi:e8cf1163-8234-4b35-865f-94a7322e40c3">
  <name>uddi-org:weather:types</name>
  <description xml:lang="en">Weather types</description>
  <overviewDoc>
    <description xml:lang="en">Previous statistics</description>
    <overviewURL>Link to somewhere</overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:general_keywords"
        keyName="SecurityLabel"
        keyValue="The value of the label"/>
  </categoryBag>
 <dsig:Signature>----Some signature---</dsig:Signature>
</tModel>
```

Figure A.33: tModel.

A.6.7 Publishing Services into the Registry

The UDDI V3 Publication API will be used to publish information regarding nations, assets, COIs, relationships between nations and assets, and relationships between assets and COIs into the service registry. Before using this API a valid UDDI V3 authToken must be retrieved, see Section A.7.6.2 for more details and on security processing on the Publishing API in general.

Relating to the publishing of services into the service registry, the UDDI V3 Publication API call save_services SHALL NOT be used. To make it easier for the publisher to publish a service, the service registry will accept an XML document containing all information about the different services to publish so that the publisher only have to make one publishing call. This is illustrated in Figure A.34. Publishing of services in the service registry should be performed by using the publishServices call defined in Section A.6.7.1.

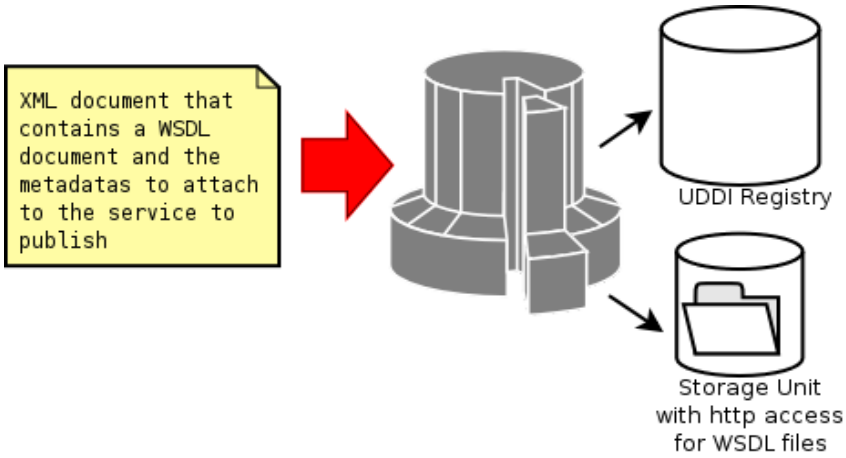

Figure A.34: Publication of a Service.

In addition, publisher-assigned keys SHALL NOT be used, the UDDI registry is responsible for key generation.

A.6.7.1 Save Services with the publishServices Call

The publishServices call is a special purpose extension to the UDDI Publishing API Set. This call MUST be used to save services in the service registry. This call adds or updates one or more businessServices elements.

The wsdl documents are known in advance and a reference to the predefined wsdl document to be used must be given. The information in the WSDL document is used to create the businessService, bindingTemplate and tModels that are necessary to publish the service. This is done following the procedure described in Section A.6.4.2. After that the metadata in the second part of the XML document is added to the businessService-element. When this is done the service can be published. Note that the tModels has to be published first.

After that the metadata in the publishServices call is added to the businessService-element. When this is done the service can be published. Note that the tModels has to be published first.

A businessService in a UDDI registry MUST always be attached to a businessEntity, so a businessIdentitifier (e.g. a UDDI:businessKey, or a custom made) must be given in order to affiliate the service to a businessEntity. This businessIdentifier is given as a metadata-element in the metadataBag. The businessEntity in question may be a nation or an asset, it cannot be a COI.

The abstraction layer will transform this call into a UDDI save_service call and forward this to the UDDI registry.

The syntax of the publishService is explained in short below. For further details on the generation of such messages please refer to the XML schema in Appendix 7. Details on the metadata can be found in Section A.6.4.

A.6.7.1.1 Syntax

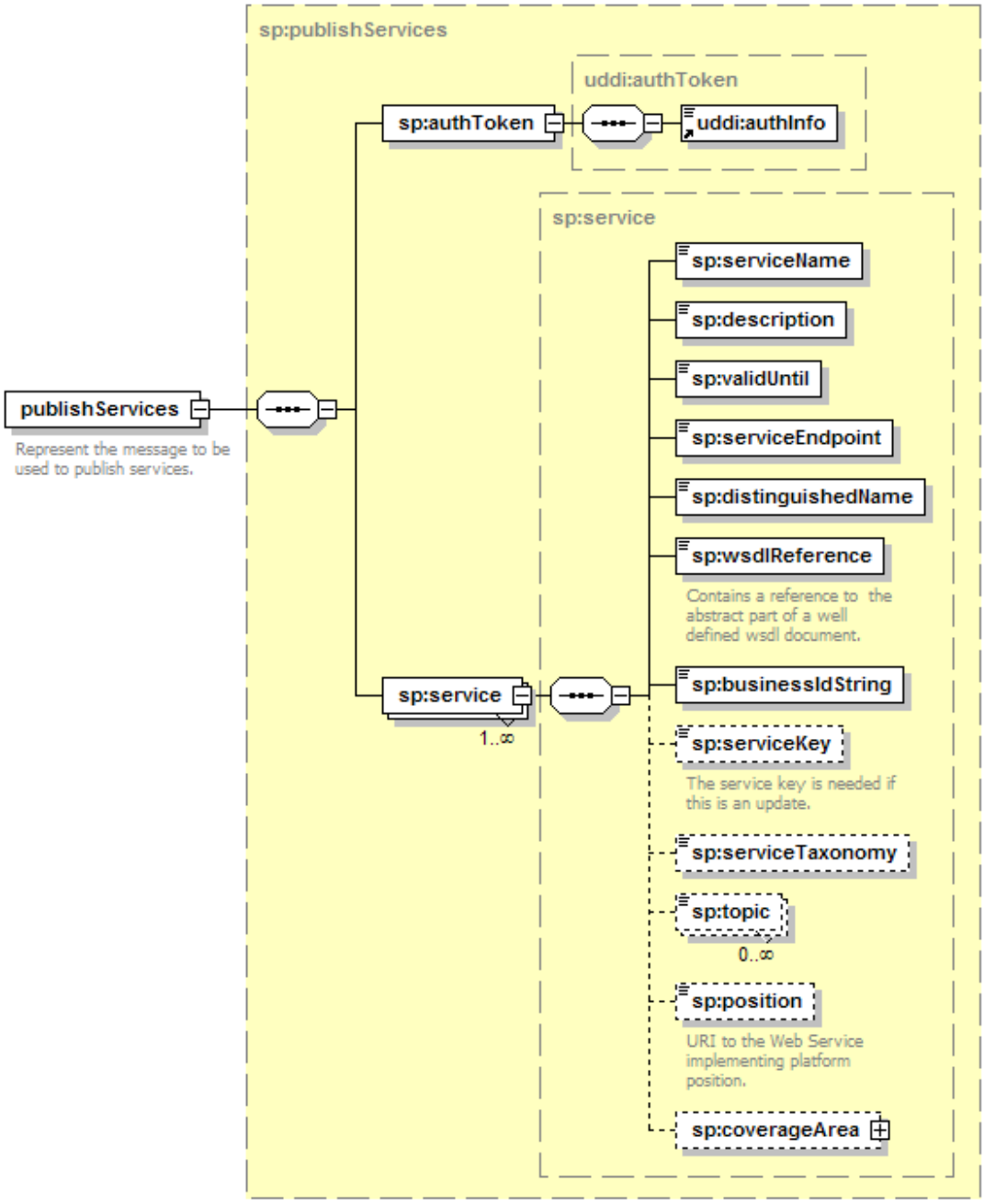

Figure A.35: Syntax of the publishServices Message.

A.6.7.1.2 Arguments

- **authInfo:** Required element that contain the following elements:
  - **uddi:authInfo:** Required element. Contains an authentication token obtained by using the get_authToken call of the UDDI Security Policy API Set.
- **Service:** Required and repeating element containing the metadata to be used to publish a service. If multiple services are present all will be have the same security label, please refere to Section A.7.6.2. A service element contain the following metada:
  - **serviceName:** Required element that provides the name of the service, textual information. This is mapped to the name variable of the UDDI businessService structure.
  - **description:** Required element that provides a textual description of the service. This is mapped to the description element of the UDDI businessService structure.
  - **validUntil:** Required element that defines the period of when this service is valid.
  - **serviceEndpoint:** Required element. This is the endpoint where the service may be contacted represented as an URI.
  - **distinguishedName:** Required element. Contains the unique distinquished name of the certificate connected with the service in LDAP. This information is used by a potential service client to retrieve the correct public key to encrypt requests to the service.
  - **wsdlReference:** Required element. Contain a string reference to the well known wsdl document to be used as a basis for the service publishing.
  - **businessIdString:** Required element. Contain a string reference to the UDDI business entity that the service will be linked to. This may either be a Nation, COI or Asset. This string must be identical to the identification string registered in the identifierBag of the business entity in question. It is used by the Abstraction Layer to retrieve the correct business key.
  - **serviceKey:** Optional element. This element MUST be present if an already existing service is beeing updated. If it is a new service this MUST be ommited since the UDDI registry is responsible for key generation.
  - **serviceTaxonomy:** Optional element. Used to classify a service.
  - **topic:** Optional and repeating element. Used to describe the topics the a notification service may publish on. If the service in question is a notificatin service it SHOULD have at least have one topic. Else if the service in question is of a request/response type of web service it MUST NOT have any topics associated.
  - **position:** Optional element. This is the endpoint of a service implementing a well known wsdl for retrieving the posistion of the service. The position service MUST implement the wsdl file presented in Appendix 8.
  - **coverageArea:** Optional element. The content of this element make up a geographic coverage area of a service in form of a rectangle. The details of this element are represented in the figure below. It should be noted that the lowerRight element is identical to the upperLeft element which is fully expanded in the figure.

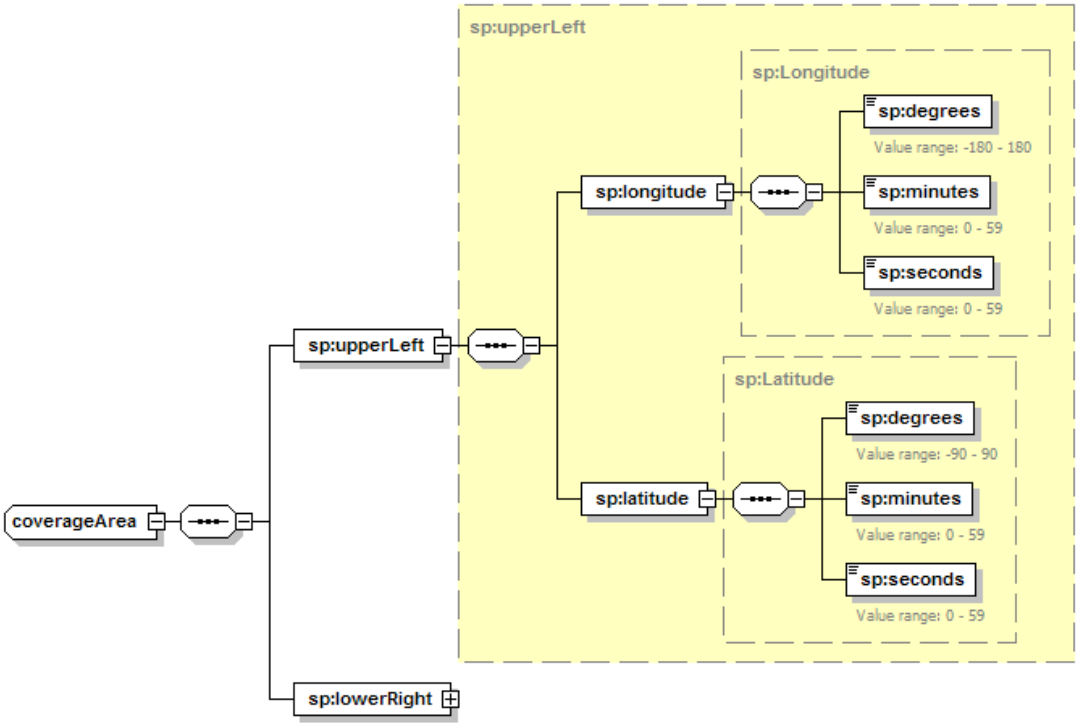

Figure A.36: Syntax of the coverageArea Element.

A.6.7.1.3 Returns and Error Reporting

Purpose made SOAP faults will be produced if security processing or validation of the publishServices message fails.

If the request is validated to be correct and forwarded to the UDDI registry, both returns and value reporting will be compliant with the returns of the UDDI V3 specification.

A.6.7.2 Resetting the Service Registry with resetRegistry Call

In some cases the ability too reset the content of the registry may be of importance. These cases include testing purposes and other. By reset it is in this case understood deleting all service records of a nation. It must be emphasized that functionality must only be used for demonstration management.

This functionality is realized using the special purpose resetRegistry call defined as an extension to the Publication API Set of UDDI V.3. The syntax of the publishService is explained in short below. For further details please refer to the XML schema in Appendix 7.

A.6.7.2.1 Syntax

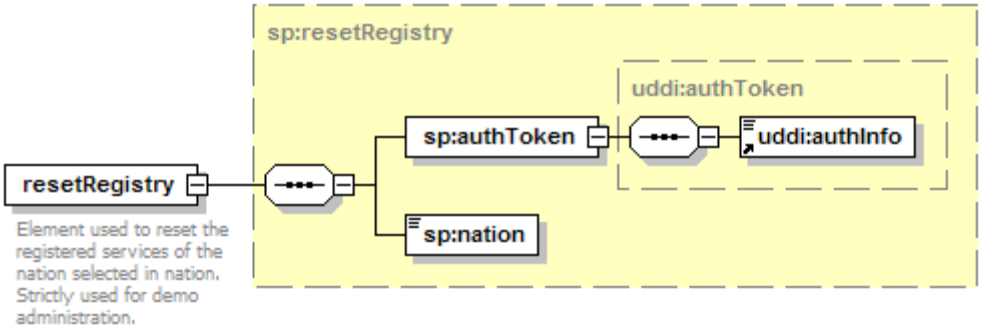

Figure A.37: Syntax of the resetRegistry Element.

A.6.7.2.2 Arguments

- **Nation:** mandatory argument containing the two letter nation code of the nation of which services should be deleted.

A.6.7.2.3 Returns and Error Reporting

Standard SOAP fault messages will be used.

A.6.8 What Extra Search Functionality is Required

There are some queries that cannot be satisfied by the UDDI registry. For instance lets say that services are registered in the registry with geographic coordinates specifying their location. If one wants to do a geographic search to find all services within a specified geographic area, the UDDI registry does not provide any means to do this.

The solution to be used in the service registry is to add an abstraction layer on top of UDDI, which will implement the extra search functionality. It is important to be aware of this and find out what extra search capabilities is required, if there is any.

A.6.9 Service Termination Policies

In dynamic environments the availability of services may not be stable. It is important that the UDDI registry reflect the services that are actually available on the network. There are two separate ways a service can leave the network. The first being graceful termination where the service is able to clean up after itself. The second case is ungraceful termination, where the service is terminated abruptly without being able to clean up after itself.

The UDDI registry has only functionality to handle the first case by providing delete methods for every entity in the UDDI data model. The second case is not supported; as UDDI does not have any built-in functionality to enforce for instance an expiration date or a valid until property. If an ungraceful termination happens UDDI will be left in an inconsistent state, containing service description for a terminated service. This means that external actions must be taken to ensure the consistency of the registry content.

The solution used in the service registry will be to require service descriptions to contain a domain-specific attribute, called "`valid until`" as mentioned in Section A.6.4. This attribute specifies the

validity interval for the service. The service providers will have to update the service description regularly to extend the validity interval, if the validity interval is exceeded the service is assumed dead and removed from the service registry. This functionality will be implemented by the service termination component of the abstraction layer.

A.6.10 Searching the Service Registry

The UDDI V3 inquiry API will be used to query the service registry. If extended search capabilities are incorporated in the service registry, additional findQualifiers must be defined, that can be used to indicate what type of extended search functionality is requested.

For a description of the security considerations when using the Inquiry API please refer to Section A.7.6.1.

A.6.11 Data Structures in UDDI – Exemplified

Since it has been identified a need to be more specific about terms like ”publisher”, ”provider” (of services) and ”user” with respect to the planned CWID demo implementation, this chapter describes a set of demo-relevant UDDI contents as an example. Hopefully, this may also give some answers to the question ”*What’s a service and what’s an asset?*”.

The example is taken from the Norwegian demo: Consider an Order of Battle with three main units, each deployed in their own ”machine” (Picture Compilation Node, PCN): Headquarter (HQ), Frigate and UAV. Each PCN is able to deliver two pub/sub services, let’s call them MTI Tracks (COI=ISR) and COP (COI=C2).

This will result in the following data structures in UDDI:

UDDI-record	**Name**	**Type / Topic**
BusinessEntity	NO	Nation
	ISR	COI
	C2	COI
	HQ	Asset
	Frigate	Asset
	UAV	Asset
PublisherAssertion	*- from NO to each of the assets HQ, Frigate or UAV -*	
PublisherAssertion	*- from both COI to all three Assets -*	
BusinessService	HQ.MTI	Topic=ISR
	HQ.COP	Topic=C2
	Frigate.MTI	Topic=ISR
	Frigate.COP	Topic=C2
	UAV.MTI	Topic=ISR
	UAV.COP	Topic=C2
BindingTemplates	*- one for each BusinessService -*	

These structures may also be visualized as a hierarchy, see Figure A.38.

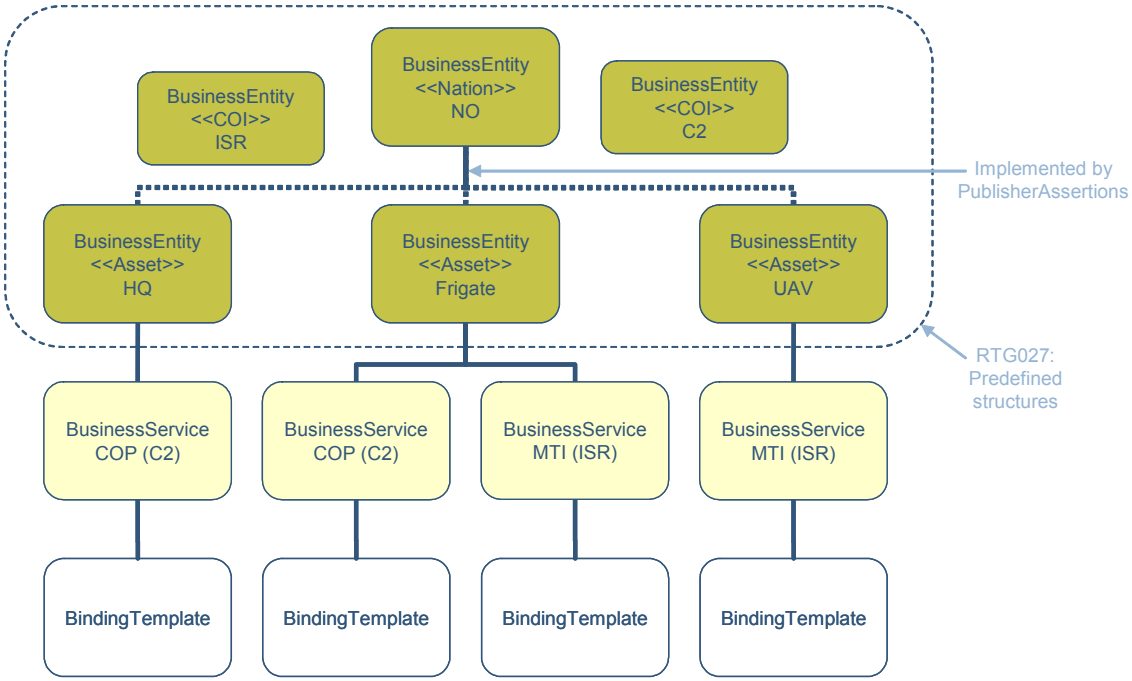

Figure A.38: Example of Norwegian Hierarchy.

The PublisherAssertions connecting COI and Assets, will not be used in the demo. In the future they may be used to describe allowed (and real) production and consumption of sets of topics.

Figure A.39 describes how the asset ”HQ” is connected to the COI ”C2”, referring to the set of topics ”NO-C2-topics” (containing topics T3 and T4) as Allowed Production. It is assumed that this is to be interpreted as allowing HQ to publish services on topics T3 and T4 (but not on T1/T2). Similarly, the other assets should be connected to their COI’s and sets of topics.

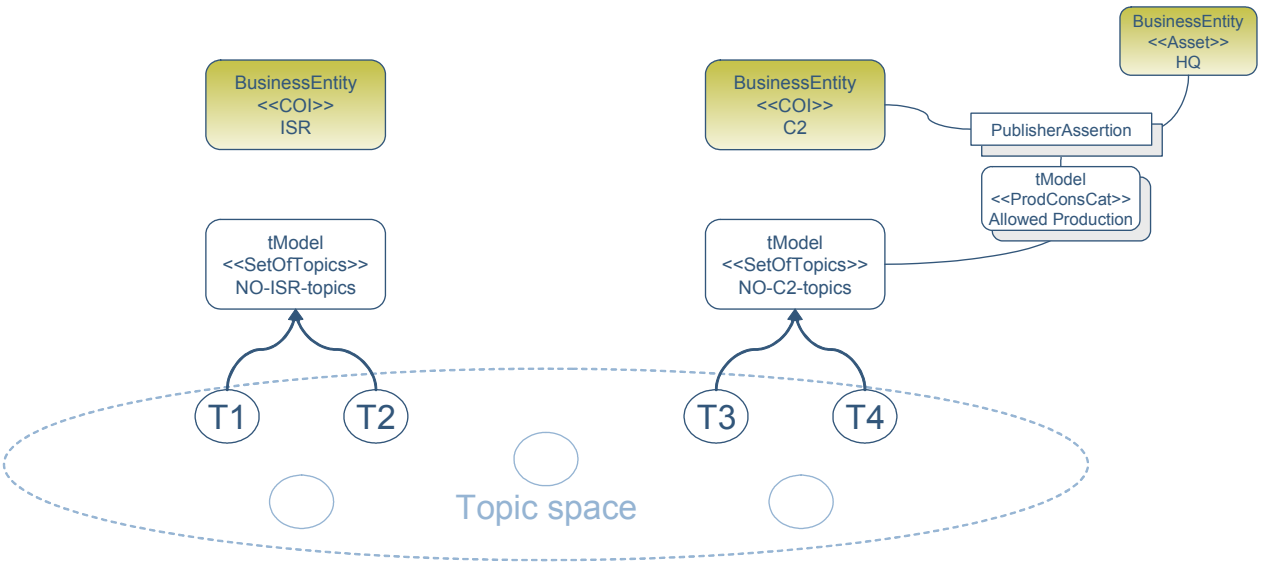

Figure A.39: Assets, COIs and Topics.

Note that the linking of COIs and Assets shown here, will not be implemented for the CWID demo.

One of the main functions for RTG027 is the service publishing function. This will be done by sending an XML file to the service registry, giving all the necessary details for the service to be saved in UDDI.

Service publishing input will include an unique reference to the BusinessEntity that the service is to be placed under.

Metadata about BusinessEntities is kept at a minimum level of complexity. To clarify the basic elements, contents of a Nation, a COI and an Asset is exemplified below.

Element name	**Nation**	**Asset**	**COI**
name	**NO**	**HQ**	**ISR**
description	**Norway as part of RTG027**	**Norwegian Headquarter**	**Tactical level track info**
contacts	TBD	TBD	
businessServices	<zero or more services>	<zero or more services>	
signature	TBD (security)	TBD (security)	TBD (security)
identifierBag	(>tModel) **nation:no/no**	(>tModel) **asset:no/HQ**	(>tModel) **coi:no/isr**
categoryBag	(>tModel) **Nation**	(>tModel) **Asset**	(>tModel) **COI**
	(>tModel) <security label>	(>tModel) <security label>	(>tModel) <security label>
		(>tModel) Categorization	

A.6.12 References

[1] OASIS, UDDI version 3.0.2, UDDI Spec Technical Committee Draft, http://uddi.org/pubs/uddi_v3.htm, 2004.

[2] OASIS, Using WSDL in a UDDI Registry, Version 2.0.2, Technical Note, http://www.oasis-open.org/committees/uddi-spec/doc/tn/uddi-spec-tc-tn-wsdl-v202-20040631.htm, 2004.

[3] OASIS, UDDI version 2.03 Data Structure Reference, UDDI Committee Specification, http://uddi.org/pubs/DataStructure_v2.htm, 2002.

[4] OASIS, UDDI version 2.04 API Specification, UDDI Committee Specification, http://uddi.org/pubs/ProgrammersAPI_v2.htm, 2002.

[5] jUDDI, open source java implementation of UDDI V2 by Apache, http://ws.apache.org/juddi/, 2005.

[6] UDDI solutions, http://www.uddi.org/solutions.html.

A.7 SECURITY SPECIFICATION

This section describes the end-to-end security services and related specifications/standards to be used in Secure Web Services Demonstrator of the NATO RTO/IST-061.

The security solutions described are based on the use of existing civil or military standards where possible, supported by solutions developed especially for this demonstrator.

A.7.1 Security Architecture

This Security architecture describes a set of national border LANs interconnected through CWID WAN and using *XML Security Domain Guards*. *XML Security Domain Guards* are used for access control of the information entering and leaving the national domains. Each national LAN contains a *Web Service Provider* and a *Directory*. The services, that a nation wants to share with its allies is replicated to the Web Service Provider of the LAN. In addition, one of the nations (or the NATO organization) will provide a Main *Web Service Registry* for looking up services published by the nations. Other nations may have a Local *Web Service Registry,* which may be synchronized with the Main service registry. The Directory systems are used for replication of X.509 Certificates and Certificate Revocation Lists (CRLs). The *System Protection Component*s (SPC) will provide the end-to-end security processing of the Web Services components.

A.7.2 Security Functionality to be Demonstrated

The security services described in this document are provided in order to protect the services of the Web Services Providers and the Web Services Registry (and the traffic exchanged between the Consumers, Providers and Registry) from unauthorized manipulation, disclosure and access. This section describes three security "packages", which will be implemented in the demonstrator.

A.7.2.1 Security Package 1: XML Security Filtering Between Domains

The aim of this security package is to show how XML security labels may be securely bound to the SOAP messages (using digital signatures) in order to allow for release control of the information passing between security domains. XML Security Domain Guards will inspect the SOAP messages being passed between the domains and control the signature and the security label attached in order to make release control decision.

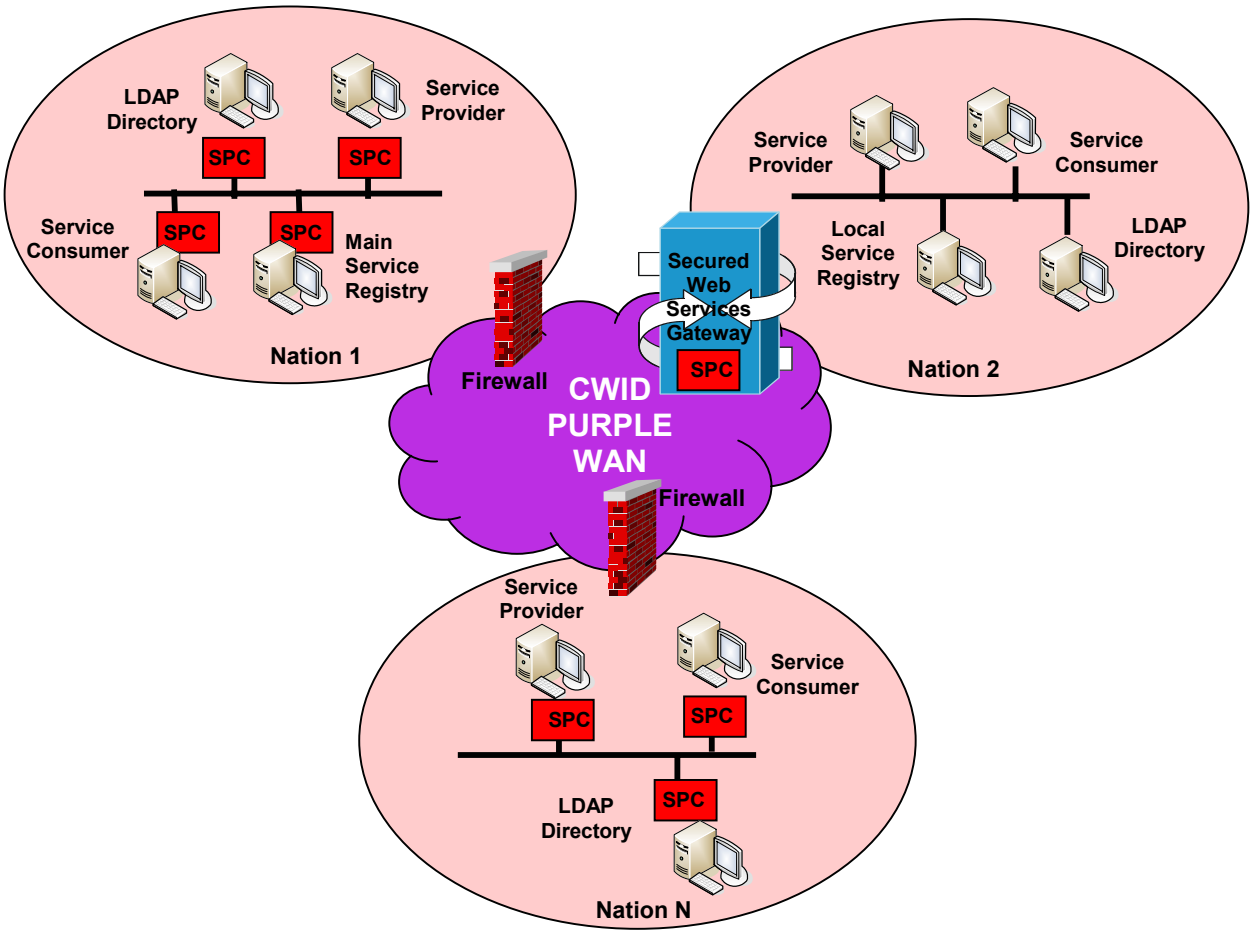

Figure A.40: The Figure Shows the Security Model of the Distributed Demonstrator. The securityfunctionality (SPC) may be placed in the end systems or in an Secured Web services Gateway.

A.7.2.2 Security Package 2: Differentiated Access Control of Information Objects

Access control is a means of enforcing the authorization of users to access the information. The aim of this demonstration package is to show how security labels bond to the information objects and user privileges may be used to allow for differentiated access control of the Service Registry and the Service Provider. XML security labels may be securely bound to the information objects of the Web Service Provider and the Web Service Registry by use of digital signatures. User privileges will indicate what type of information the user is allowed to access. In this demonstrator we will for simplicity include the user privileges in the users certificate in the form of a XML security label (this is not a very dynamic solution for an operational system). The access control will be enforced by comparing the users privileges included in the certificate with the information in the security label attached to the information.

A.7.2.3 Security Package 3: End-to-End Authentication, Integrity and Confidentiality

The aim of this security package is to show how the security services Integrity, Authentication and Confidentiality may be used to secure the SOAP messages in transit between the Service Consumers, Service Providers and the Service Registry.

A.7.2.3.1 Integrity

The Integrity service provides an indication to the recipients of the transmitted information whether the information has been modified, deleted or substituted without authorization. This security service will be achieved by hashing and digitally signing the XML information to be protected.

A.7.2.3.2 Authentication

The Authentication service gives assurance of the identity of some entity (a person, a process or a system). It is the means of gaining confidence that an originator of the information is who he claims to be. This security service will be achieved by hashing and digitally signing the XML information to be protected.

A.7.2.3.3 Confidentiality

The End-to-end Confidentiality service cryptographically protects the content of the SOAP message against disclosure. End-to-end Confidentiality can help to enforce need to know restrictions, or enables multiple different user communities to share the same network. The service is independent of the network and systems transporting the message. The content of the SOAP messages will be encrypted during transport, but the information will not be encrypted during storage in the Service Provider or Service Registry because of the requirement to have access to the metadata when invoking services or retrieving UDDI records.

A.7.3 SOAP Message Security

All SOAP messages SHALL be secured using the OASIS standard Web Services Security v1.0. [WS-Security 2004-OASIS 200401].

The specifications to be implemented are: Web Services Security: SOAP Message Security 1.0 (WS-Security 2004).

The “mustUnderstand” attribute for this element SHALL be set to “true” for the WSS header.

Available Freeware:
http://java.sun.com/webservices/downloads/webservicespack.html

A.7.3.1 XML Signature Details

See Section A.7.9 and Appendix 3 for algorithm and certificate specifications.

A.7.3.2 XML Encryption Details

The triple DES algorithm SHALL be used for symmetric encryption. Symmetric encryption keys SHALL be encrypted using the recipients public key and conveyed with the SOAP message as described in the xenc:EncryptedKey section of the [WS-Security 2004-OASIS 200401] specification.

A.7.3.3 Timestamp

Security Timestamps SHALL be used as defined by the [WS-Security 2004-OASIS 200401] specification.

A.7.3.4 SOAP XML Security Label

It SHALL be possible to attach a XML security label to the SOAP message. The InformationSecurityLabel (see Section A.7.4.1) SHALL always be present and will indicate the sensitivity

of the information attached. This Information Security Label will be used for release control of the SOAP messages by the XML Guards.

A.7.3.5 SOAP Addressing Heading Extension

In order to be able to perform the required security filtering of the outbound traffic in the XML Security Domain Guard, the SOAP header must be extended with an address field. This address filed SHALL be based on the [WS-Addressing-Core] specification:

```
?xml version="1.0" encoding="UTF-8"?>
<S11:Envelope                                xmlns:S11="http://www.w3.org/2003/05/soap-envelope"
xmlns:wsa="http://www.w3.org/2005/08/addressing">
        <s:Header>
                <wsa:To> http://www.client.org/Endpoint…. </wsa:To>
        </s:Header>
        <s:Body>
                …..
        </s:Body>
</s:Envelope>
```

This address and the InformationSecurityLabel of the SOAP header MAY be used by the XML Security Domain Guard for filtering of information leaving a security domain. See Section A.7.8.

A.7.3.6 Elements to be Signed

An example of a SOAP message with all the header fields is shown below. Note that some sub elements are not included.

```
                        <ds:Signature>
                                <ds:SignedInfo>
                                        ...
                                        <ds:Reference URI="#to">
                                        ...
                                        </ds:Reference>
                                        <ds:Reference URI="#from">
                                        ...
                                        </ds:Reference>
                                        <ds:Reference URI="#messageID">
                                        </ds:Reference>
                                        <ds:Reference URI="#timestamp">
                                        ...
                                        </ds:Reference>
                                        <ds:Reference URI="#informationSecurityLabel">
                                        ...
                                        </ds:Reference>
                                        <ds:Reference URI="#bodypart">
                                        ..
                                        </ds:Reference>
                                </ds:SignedInfo>
                                <ds:SignatureValue>Hp1z….
                                </ds:SignatureValue>
                                <ds:KeyInfo>
                                        <ds:X509Data>
                                                <ds:X509IssuerSerial>
                                                        <ds:X509IssuerName>CN=FFI Sub CA….
                                                        </X509IssuerName>
                                                        <ds:X509SerialNumber>1234
                                                        </ds:X509SerialNumber>
                                                </ds:X509IssuerSerial>
                                                <ds:X509SubjectName>CN=FFI Sender….
                                                </ds:X509SubjectName>
                                        </ds:X509Data>
                                </ds:KeyInfo>
                                ...
                        </ds:Signature>
<?xml version="1.0" encoding="UTF-8"?>
<S11:Envelope xmlns:S11="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:wsa="http://www.w3.org/2005/08/addressing”
xmlns:wsse=”http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd”
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
xmlns:ds=”http://www.w3.org/2000/09/xmldsig#” xmlns:xenc="http://www.w3.org/2001/04/xmlenc#”>
        <S11:Header>
                <wsa:To wsu:Id="to">...</wsa:To>
                <wsa:From wsu:Id="from">...</wsa:From>
                <wsa:MessageID wsu:Id="messageID">...</wsa:MessageID>
                <wsse:Security S11:mustUnderstand="true">
                        <wsu:Timestamp wsu:Id="timestamp">
                            ...
                        </wsu:Timestamp>
                        <slab:SecurityLabel xmlns:slab="…">
                                <slab:LabeledObjectGroup Id="informationSecurityLabel">
                                ...
                                </slab:LabeledObjectGroup>
                        </slab:SecurityLabel>
                        <ds:Signature>
                                <ds:SignedInfo>
                                        ...
                                        <ds:Reference URI="#to">
                                        ...
                                        </ds:Reference>
                                        <ds:Reference URI="#from">
                                        ...
                                        </ds:Reference>
                                        <ds:Reference URI="#messageID">
                                        <ds:Reference>
                                        <ds:Reference URI="#timestamp">
                                        ...
                                        </ds:Reference>
                                        <ds:Reference URI="#informationSecurityLabel">
                                        ...
                                        <ds:Reference>
                                        <ds:Reference URI="#bodypart">
                                        ..
                                        </ds:Reference>
                                </ds:SignedInfo>
                                <ds:SignatureValue>Hp1z….
                                </ds:SignatureValue>
                                <ds:KeyInfo>
                                        <ds:X509Data>
```

```
                                        <ds:X509IssuerSerial>
                                                <ds:X509IssuerName>CN=FFI Sub CA….
                                                </X509IssuerName>
                                                <ds:X509SerialNumber>1234
                                                </ds:X509SerialNumber>
                                        </ds:X509IssuerSerial>
                                        <ds:X509SubjectName>CN=FFI Sender…
                                        </ds:X509SubjectName>
                                </ds:X509Data>
                        </ds:KeyInfo>
                        ...
                </ds:Signature>
                <xenc:EncryptedKey>
                        ...
                        <xenc:ReferenceList>
                                <xenc:DataReference URI="#encryptedBodypart"/>
                        </xenc:ReferenceList>
                        <xenc:CipherData>
                                <xenc:CipherValue>... (encypted sym key)
                                </xenc:CipherValue>
                        </xenc:CipherData>
                        <ds:KeyInfo>
                                <ds:X509Data>
                                        <ds:X509IssuerSerial>
                                                <ds:X509IssuerName>CN=FFI Sub CA….
                                                </X509IssuerName>
                                                <ds:X509SerialNumber>5678
                                                </ds:X509SerialNumber>
                                        </ds:X509IssuerSerial>
                                        <ds:X509SubjectName>CN=FFI Receiver…
                                        </ds:X509SubjectName>
                                </ds:X509Data>
                        </ds:KeyInfo>
                </xenc:EncryptedKey>
        </wsse:Security>
    </S11:Header>
    <S11:Body wsu:Id="bodypart">
        <xenc:EncryptedData Id="encryptedBodypart">

                <xenc:CipherData>
                        <xenc:CipherValue>...(encrypted body)...</xenc:CipherValue>
                </xenc:CipherData>
        </xenc:EncryptedData>
    </S11:Body>
</S11:Envelope>
```

The following elements SHALL be signed

- wsa:To
- wsa:From
- wsa:MessageID
- wsu:Timestamp
- "informationSecurityLabel"
- “bodypart”, includes the “encryptedBodypart”

The KeyInfo in the EncryptedKey element SHALL be used to identify the encryption certificate, i.e. the public key, used by the sender to encrypt the symmetric key.

The KeyInfo in the Signature element SHALL be used to identify the senders signing certificate, i.e. the public key needed to verify the signature.

In order to reduce the message size, certificates SHALL never be included in the SOAP message. The certificates used SHALL be identified by the X509IssuerSerial and X509SubjectName elements.

A.7.4 XML Security Label Definition

Security Labels provides an indication of the security policy, sensitivity, compartments, and other handling caveats associated with the information. The Security labels can be used for purposes such as access control of information objects or a source for security guard functions to control the information being exchanged between domains.

The sensitivity information in a security label can be compared with a user's security privileges in order to determine if the user is allowed to access the services or advertisements of the services. Access controls are performed in each domain in accordance with the security policy in force.

A.7.4.1 The Information Security Label Syntax and Processing Specification

The InformationSecurityLabel describes the sensitivity of the encrypted content. It is anticipated that the SOAP header do not contain any classified information. This label is therefore also used to indicate the sensitivity of the whole SOAP message. This label is processed by the XML domain security guards.

The Detached XML Security Label syntax as defined in Section A4.2.1 and A4.2.4.1 of Appendix 4 SHALL be used, but the ID of the LabelledObjectGroup of the security label element SHALL be set to "informationSecurityLabel" as shown in the example below. The rational for this is to be able to distinguish between the security labels when they both are attached to the SOAP message. The Security Label attached to the UDDI records is of this type. The namespace http://nc3a.nato.int/2004/06/xmlslab# shall be used.

```
<slab:SecurityLabel xmlns:slab="http://nc3a.nato.int/2004/06/xmlslab#">
    <slab:LabeledObjectGroup Id="informationSecurityLabel">
      <slab:ConfidentialityLabel>
        <slab:SecurityPolicyIdentifier>
          RTO-IST-061-Mission
        </slab:SecurityPolicyIdentifier>
        <slab:SecurityClassification>
          SECRET
        </slab:SecurityClassification>
      </slab:ConfidentialityLabel>
    </slab:LabeledObjectGroup>
</slab:SecurityLabel>
```

A.7.4.2 XML Label Guidance

See Appendix 5, Section A5.1.

A.7.5 Security Privileges

In this demonstrator a simplified privilege management solution will be used. The security privileges of a user will be defined using a security label (as defined in Section A.7.4) The security privileges of a user, will be compared to the security label of the Notification, Service Response or the UDDI record according to the matching rules defined in Section A.7.5.4. The security privileges of a "user" will be stored in the certificate of the "user" as defined in Section A.7.5.2. This is not a very dynamic solution, but is used in the demonstrator for simplicity reasons. A "user" may e.g. be a person, role, user-groups, COI or any other entity that may logically be attached security privileges.

A.7.5.1 The Privilege Security Label Syntax and Processing Specification

The Detached XML Security Label syntax as defined in section A4.2.1 of Appendix 4 SHALL be used, the ID of the LabelledObjectGroup of the security label element SHALL be set to "privilegeSecurityLabel" as shown in the example below.

```
<slab:SecurityLabel xmlns:slab="http://nc3a.nato.int/2004/06/xmlslab#">
    <slab:LabeledObjectGroup Id="privilegeSecurityLabel">
      <slab:ConfidentialityLabel>
        <slab:SecurityPolicyIdentifier>
          RTO-IST-061-Mission
        </slab:SecurityPolicyIdentifier>
        <slab:SecurityClassification>
          SECRET
        </slab:SecurityClassification>
      </slab:ConfidentialityLabel>
    </slab:LabeledObjectGroup>
</slab:SecurityLabel>
```

A.7.5.2 Certificate Extension for the Privilege Security Label

The Privilege Security Label SHALL be added as a certificate extension:

privilegeSecurityLabel EXTENSION :: = {
SYNTAX PrivilegeSecurityLabel
IDENTIFIED BY id-privilegeSecurityLabel }

id-privilegeSecurityLabel OBJECT IDENTIFIER ::= 2 16 578 1 2 1 28 724 99

Where **privilegeSecurityLabel** is a XML Security Label as defined in Appendix 4.

Note that the object identifier is selected for this demonstrator only and is not registered.

Example:
…
SEQUENCE {
OBJECT IDENTIFIER 2.16.578.1.2.1.28.724.99
OCTET STRING
<?xml version="1.0"?><SecurityLabel>…

A.7.5.3 Privilege Label Semantics

- SecurityPolicyIdentifier: Identifies which security policy to be used for evaluating these privileges.
- SecurityClassification: Indicate the highest classification this user is allowed to access.
- PrivacyMark: Not Used.
- SecurityCategories: Indicates which nation, alliance, Role and/or Community of Interest (COI) the user belongs to.

See Appendix 5 for more details.

A.7.5.4 Security Label and Privilege Label Matching Rules

See Appendix 5, Section A5.2.

A.7.6 Securing the Web Service Registry

This section describes how to implement security in the Service Registry based on version 3 of the OASIS UDDI specification.

UDDI v.3 defines APIs for access to the data within the service registry. Two of these are the Inquiry API, which may be used for searching for records and the Publish API, which may be used for insertion and updates of records. In order to secure these interfaces and enforce differentiated access control on the stored records, we have introduced a System Protection Component (SPC) as part of the Security Abstraction Layer in front of the UDDI APIs. This security abstraction layer will perform the WSS related security processing of the SOAP messages (authentication, signature handling and encryption), in addition to performing differentiated access control on the UDDI records based on the security labels of the UDDI records and the privileges in the user certificates.

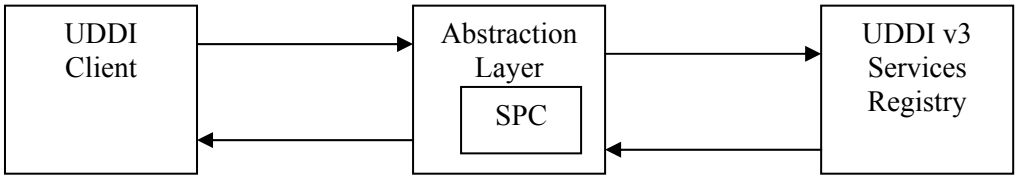

Figure A.41: Shows the Relation between the Client, the UDDI SPC, and the UDDI Registry. A Security Protection Component (SPC) is placed in front of the UDDI Inquiry and Publish APIs in order to perform the defined Security Processing.

A.7.6.1 Inquiry API

Access to the methods of the UDDI Inquiry API Set is restricted to users with a valid certificate, and that the SOAP message is correctly signed and encrypted. Access to the information in the result set is controlled comparing the security label of the UDDI records with the privileges in the user's certificate.

For initial testing: If execution is in security mode off, the Inquiry API is publicly accessible.

On inquiry, the security element of the SOAP header SHALL contain an Information Security Label (as defined in Section A.7.4.1). The InformationSecurityLabel will indicate the sensitivity of the information contained in the SOAP message.

The security label is defined in Section A.7.4 and SHALL be placed in the WSS Security element of the SOAP header (extension). The security label SHALL be bound to the content of the SOAP message by the digital signature in the security element of the SOAP header.

A.7.6.1.1 Security processing of the Security Protection Component

On reception of a SOAP message from the UDDI client containing any of the UDDI v3 Inquiry API "find" or "get" calls described in the standard, the following (or equivalent) processing SHALL be performed by the Security Protection Component.

1. Inspect the keyInfo element of the XML signature in the SOAP header in order to retrieve the X509IssuerSerial and X509SubjectName, which together can uniquely identify the user's signing certificate. Use this information to fetch the certificate with the User Privileges from the LDAP Directory.
2. Validate the signature to see if the request comes from a trusted UDDI client.
3. If the signature validation process is OK, store the User Privileges for control against the Security Labels of the returned records (if any) from the UDDI Inquiry function.
4. Inspect the keyInfo element of the XML encryption in the SOAP header in order to retrieve the signature X509IssuerSerial and X509SubjectName, which together can uniquely identify the certificate used to encrypt the symmetric key. Use this information to fetch the certificate.
5. Decrypt the SOAP message content and retrieve the Inquiry API function call.

6. Decompress the SOAP content.
7. Invoke the Inquiry API function using the standard UDDI Inquiry API.

When the Inquiry API function returns, traverse the list of records returned, validate the signature and inspect the Security Label for each one of them and build a list of records that matches the user's privileges.

8. Compress, Encrypt, label, sign and return the list of records to the UDDI client if not empty, else return any other valid response. The classification of the InformationLabel attached to the SOAP message SHALL be set to the highest classification of the included UDDI records.

A.7.6.2 Publish API: Security Processing of the Security Protection Component

In order to be allowed to publish to the registry, a publisher must be listed in the Access Control List of the registry. The publisher must first receive an authToken by invoking the *get_authToken* method in the Security Policy API Set. This *authToken* can be used to invoke a publish via the Publication API Set.

All the SOAP messages must be encrypted, labeled and signed correctly, if not the message will not be forwarded to the backend UDDI registry.

Several services may be published in the same publish message given that they have equivalent security labels. If the services have different security labels, they must be published using one publish message for each variation of the security label (ref. Section A.6.7.2). The security label from the SOAP message used in the publish request, will be used to mark the records put into the UDDI registry. The SOAP message used to send the publish response, will have the same security label as used for the publish request.

For initial testing: Even if execution of the demonstrator is in security mode off, a publisher must still acquire an authToken.

On reception of a SOAP message from the UDDI client containing any of the UDDI v3 Publish API described in the standard, the following security processing SHALL be performed by the Security Protection Component:

1. Inspect the keyInfo element of the XML signature in the SOAP header in order to retrieve the X509IssuerSerial and X509SubjectName, which together can uniquely identify the user signing certificate. Use this information to fetch the certificate from the LDAP Directory.
2. Validate the signature to see if the request comes from a trusted UDDI client and check if the signer is allowed to publish to the UDDI registry (comment: map the certificate ID against the Auth Info access control mechanism in UDDI v2).
3. If the previous tests are OK, identify the encryption certificate and decrypt and decompress the SOAP message content.
4. Validate the signature of the UDDI record in order not to put any invalid data into the UDDI registry.
5. Invoke the Publish API function using the standard UDDI Publish API.
6. Compress, Encrypt, label, sign and return the response from the UDDI registry to the UDDI client.

A.7.6.3 Binding an XML Security Label to the UDDI Records

The following processing SHALL be performed by the UDDI client before publishing the record to the UDDI registry using the Publish API:

1. An XML Security Label SHALL be bound to each record published to the UDDI registry using an XML Signature (see Section A.7.4 for details).
2. The SOAP message used for transfer SHALL then be encrypted and signed.

A.7.7 Securing the Web Service Provider

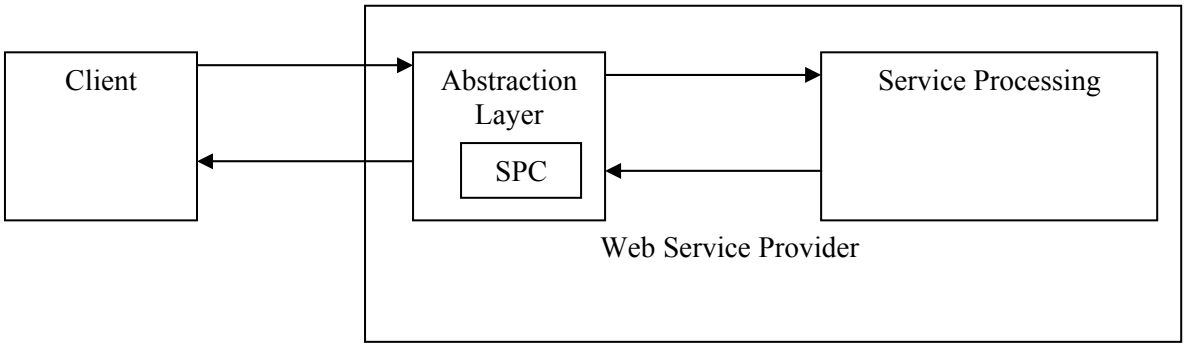

Figure A.42: Shows the Relation between the Client, the Web Service Provider SPC, and the Web Service Provider. A Security Protection Component is introduced in order to perform the defined Security Processing.

A.7.7.1 Securing WS Notification

The WS-BaseNotification standard makes a distinction between the roles NotificationConsumer and Subscriber, but as described in the definition of a *Subscriber* in Section A.4.5, in this demonstrator context **the Subscriber will also be the Notification Consumer**. Another important issue (ref. Section A.4.5) is that WS-BrokeredNotification is not used in this specification. These restrictions influence the following specification of the security functionality. Therefore, in the following we will use the term *Subscriber* to mean both a Subscriber and the NotificationConsumer.

A.7.7.1.1 Notification Producer: Security processing on receipt of a Subscription Request

On reception of a SOAP message from the Subscriber containing a wsnt:Subscribe element, the following (or equivalent) processing SHALL be performed by the NotificationProducer.

1. Inspect the keyInfo element of the XML signature in the SOAP header in order to retrieve the X509IssuerSerial and X509SubjectName, which together can uniquely identify the signing certificate. Use this information to fetch the certificate with the User Privileges from the LDAP Directory.
2. Validate the signature to see if the request comes from a trusted Subscriber.
3. If the signature validation process is OK, store the User Privileges for later control against the Security Label(s) of the Notifications.
4. Retrive the encryption certificate and decrypt and decompress the SOAP message content.
5. The NotificationProducer SHALL create a Subscription Resource for the Subscription. The User Privileges found in the certificate SHALL be included in this Subscription Resource for matching against the InformationSecurityLabel of the Notifications.

A.7.7.1.2 Notification Producer: Security Processing on Generation of Notifications

When the Notification Producer has a notification to distribute, the NotificationProducer SHALL match the InformationSecurityLabel in the SOAP message of the notification against the User Privileges registered for each subscription. For each of the subscriptions it identifies a match, it shall issue the Notification to the Subscriber associated with the subscription.

Each Notification sent SHALL be compressed, encrypted, labeled and signed by the Notification Producer. The classification of the InformationLabel attached to the SOAP message SHALL be set to the highest classification of the included information.

A.7.7.1.3 Notification Consumer

When a Subscriber sends a Subscribe message to a NotificationProducer, this SOAP message SHALL be compressed, encrypted, labeled and signed. As for the security of the UDDI Inquiry API (see Section A.7.6.1) the security element of the SOAP header SHALL contain an Information Security Label (as defined in Section A.7.4.1). The InformationSecurityLabel will indicate the sensitivity of the information contained in the SOAP message and will be used by the XML Security Domain Guards.

A.7.7.2 Securing Request/Response Interactions

Request and response interactions SHALL be secured using the same procedures and functionality as described for the notifications in Section A.7.7.1.

A.7.8 The XML Security Domain Guard

XML security labels may be securely bound to the SOAP messages (using digital signatures) in order to allow for release control of the information passing between security domains. The XML Security Domain Guards SHALL inspect the SOAP messages being passed between the domains and verify the signature and enforce release control based on the InformationSecurityLabel attached to the WSS security element of the SOAP header. This release control will be required to ensure that only information allowed to be released are leaving the national domain.

A.7.8.1 Outbound Traffic

The Security domain guard SHALL perform the following (or similar) processing based on the information in the SOAP header:

1. Inspect the keyInfo element of the XML signature in the SOAP header in order to retrieve the X509IssuerSerial and X509SubjectName, which together can uniquely identify the user signing certificate. Use this information to fetch the certificate from the LDAP Directory.
2. Validate the signature
3. Check the InformationSecurityLabel in order to determine if the SOAP message is to be released according to national policy.

A.7.9 PKI

The text in this section is based on the PKI specification of the military standard ACP 145 version 1.0. Profiles for certificates and CRLs are included in Appendix 3. If there are any mismatches between this text and the profiles for CWID 06, the profiles takes precedence.

A.7.9.1 Certificate Management

Each Nation has a PKI to support national Public Key (PK) enabled applications.

The National PKI should be designed to minimize the amount of information that would need to be published to other nations. It is therefore recommended that the PKI should only consist of a root (self-signing entity), a subordinate CA and the “user” (see Figure 7.4).

Trust is distributed amongst the systems via the exchange of the National PKI’s Root certificates (see below). The Root certificates are placed into a "trust file," which the systems uses during digital signature

verification (see Figure 7.4). Only the PKI Root CA certificate is needed in the trust file because the CA certificate and the certificate of the system who performed the signature will be available from the LDAP Directory.

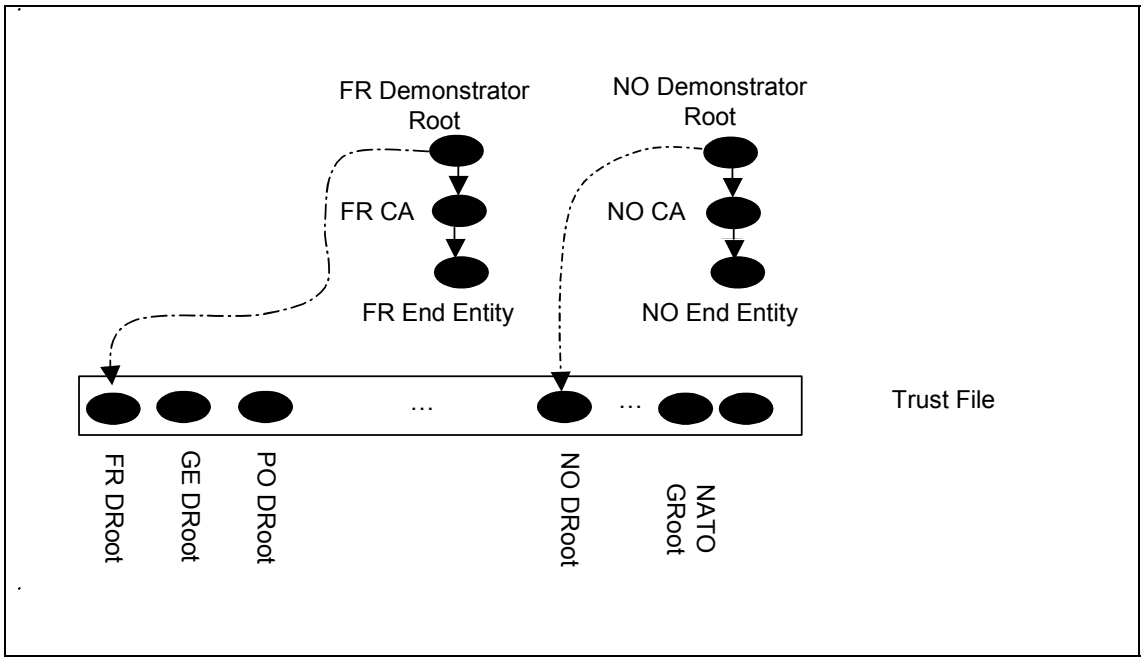

Figure A.43: Demonstrator PKI Architecture.

A.7.9.2 Certificate Generation

The specific requirements for the Root CA certificate, Intermediate CA's certificate(s), and "user's" certificate are included in Section A.7.9.6. The processing requirements for EEs can be found in Section A.7.9.6.

A.7.9.3 Certificate Distribution

A.7.9.3.1 Root Certificate Distribution

Distribution of the National PKI's Root Certificate is performed via LDAP or some other bilaterally agreed mechanism. The National PKI's Root Certificate contains the DER encoded public key certificate of the Nation's Root Certificate.

A.7.9.3.2 Intermediate CA Certificate(s) Distribution

Distribution of Intermediate CAs certificate is done via the pkiCA X.500 directory object class, as described in Section A.7.10.7. The CA's public key certificate is included in the cACertificate attribute. Note that the CA's directory entry and the directory name contained in the CA's certificate issuer field must match.

A.7.9.3.3 User Certificate Distribution

The "user's"' public key certificate is distributed in the directory "user" entry as described in Section A.7.9.3. The "user" in this context may e.g. be a person, organization, role, application, process, ….

A.7.9.4 Lifecycle

All subscribers of the PKI may, at one time or another, require new certificates. Root CAs, Intermediate CAs, and End Entities may need new certificates to continue subscribing to the PKI after the validity date in their certificate passes, to resubscribe to the PKI after their certificate has been compromised, and to update some information bound in to the certificate (e.g., subject name or key). Various terms (e.g., renewal, rekey, update, reissue) are sometimes used to address the various scenarios but for simplicity this specification will use the term renewal. Further, the procedures defined herein imply that the renewal process will result in a name being bound to new key (i.e., a new key will be generated for the public key certificate).

A.7.9.4.1 Root Certificate and Intermediate CA's Certificate Renewal

For simplicity, the validity dates of the Nation's Root Certificate and the CA Certificate SHALL be set to a date that will not cause them to expire during the lifetime of the demonstrator.

A.7.9.4.2 "User" Certificate Renewal

To support "user" certificate expiration (i.e., current date is past the validity date in the certificate), a new certificate needs to be issued to the "user" and distributed via the procedure defined in Section A.7.9.3. There is a need to generate a new "user" certificate prior to the end of the validity date because the new "user" certificate has to be distributed in the Directory prior to use.

To support "user" certificate compromise, a new "user" certificate needs to be issued and distributed via the procedures in Section A.7.9.3. Further revocation notifications must be generated and distributed via the procedures in Section A.7.9.5.

A.7.9.5 Revocation Notification

Revocation of a National PKI Root CA or any Intermediate CA certificate(s) that was used to issue a "user" certificate is a catastrophic event that requires immediate action be taken by each of the National PKI Points of Contact (POC). For the lifetime of the demonstrator, revocation of the National PKI Root CA or any Intermediate CA certificate(s) SHALL NOT be performed.

In the event of a "user" certificate revocation, all Nations with which the revoked "user" certificate has been shared must be contacted to indicate that the "user" certificate has been revoked. If the "user's" certificate is to be renewed, a new private key needs to be generated and the corresponding public key certificate needs to be distributed (see Section A.7.9.3). The CA that issued the old End Entity certificate must issue and distribute a CRL indicating the old End Entity certificate is revoked.

CRL Issuance (update) will generally happen when a certificate has been revoked. However, all CAs including Root CAs will be required to generate CRLs. If there are no revoked certificates the CAs are required to publish an empty CRL (i.e., a CRL with no revoked certificates). The directory entry for a particular CA contains the ***certificateRevocationList*** attribute in which all revoked certificates will be included for at least one period beyond the revoked certificate's validity period, as per paragraph 3.3 of RFC 3280. The directory entry for a particular CA also contains the ***authorityRevocationList*** attribute in which an exact copy of the contents of the ***certificateRevocationList*** attribute will be stored. Note that the CA's directory entry and the directory name contained in the CA's CRL ***issuer*** field are the same.
The specific requirements for a CA's CRL generation are included in Section A.7.9.7. The processing requirements for EEs can also be found in those paragraphs.

A.7.9.6 Public Key Certificates

This section specifies the Version 3 (V3) X.509 certificates as described in Recommendation X.509 (1997) and profiled in RFC 3280. The NATO certificate profiles are used and documented in Section A.7.9.6. Three "types" of certificates are needed to support the demonstrator architecture. Three "types" of certificates are described herein and are as depicted in Figure A.44.

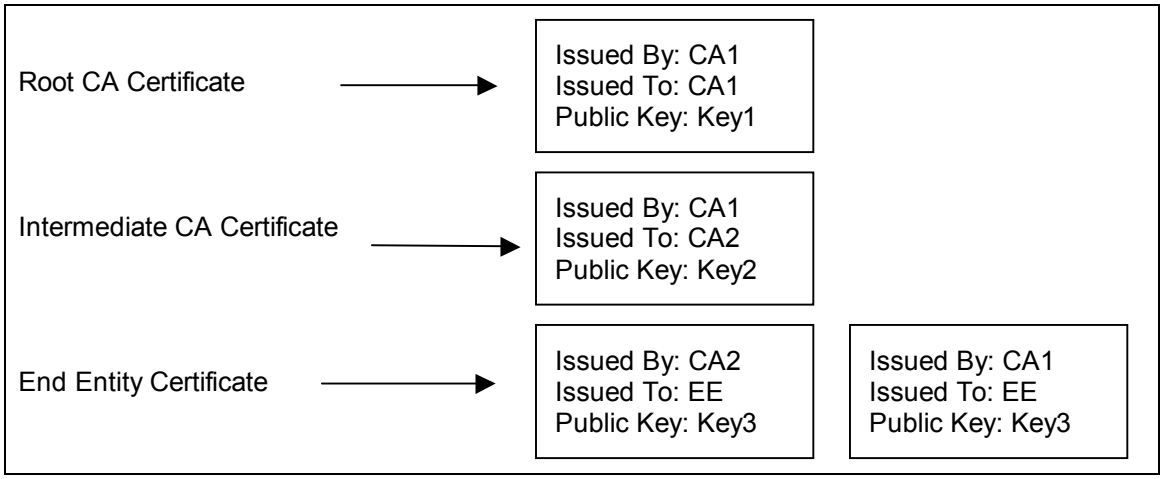

Figure A.44: Demonstrator Certificate Types

Root CA certificates: – These certificates act as the trust anchors. As described in Section A.7.9.3 they are exchanged between each Nation and are used by each Nation's systems during digital signature verification.

CA certificates: – These certificates are issued by the Root, and issue the "users'" certificates. They are exchanged, as described in Section A.7.9.3 and are used by each Nation's systems during digital signature verification.

"User" certificates: – These certificates are issued by a intermediate CA. They are used by the National systems to verify digitally signed information received. The private key corresponding to the public key in the certificate is used to generate the digital signatures on the information to be protected. These certificates SHALL also be used for encryption.

A.7.9.6.1 Public Key Certificate Profiles

The "profile" contained within this document is considered to be compliant to RFC 3280. There following are additional constraints for National End Entities:

- Processing version is restricted to Version 3 public key certificates. National systems SHALL only issue Version 3 certificates; therefore, it is unnecessary to support processing additional values for version and the corresponding differences between the certificates.

Processing the following fields and extensions is technically required:

- Processing the ***authorityKeyIdentifier*** and ***subjectKeyIdentifier*** extensions is required.
- Marking ***keyUsage*** as critical.

The following two paragraphs describe specific implementation details for public key certificate fields and extensions used to intercommunicate in this environment. CA implementation guidance is provided for generating each field and extension and EE implementation guidance is also provided for processing each

field and extension. The fields described in Section A.7.9.6.2 are included in all certificates, while inclusion of the extension in Section A.7.9.6.3 varies depending on the "type" of certificate. Annex A provides a detailed profile for the different certificates.

A.7.9.6.2 Public Key Certificate Fields

Public Key Certificate Fields are as follows:

a. All certificates used within this environment shall indicate Version 3 (v3) by including the integer value of two (2) in the version field. EEs shall support processing this field, but they need only recognize Version 3 certificates.
b. CAs shall include the certificate's ***serialNumber*** field in every certificate and it shall be a positive integer that is shorter than 20 octets and unique for a particular CA. This may be accomplished by the CAs maintaining and incrementing a counter to assign certificate serial numbers. Recipients shall support processing this field.
c. The **AlgorithmIdentifier** in the **signature** field shall include the OID as per Table 7-2 of the signature algorithm used to sign the certificate. Inclusion of the parameters is as per Table A.1. The values included in this **AlgorithmIdentifier** field must be identical to those in the **AlgorithmIdentifier** field in paragraph k, below. EEs shall support processing this field; however, the parameters themselves are not used during signature verification.
d. The issuer's **Name** in the **issuer** field shall contain the unique X.500 DN identifying the issuer. CAs and EEs are required to support the following standard naming attribute types: country, organization, organizational-unit, distinguished name qualifier, state or province name, common name, and serial number.
e. The certificate validity period field contains two dates: **notBefore** and **notAfter**. Both the dates shall be expressed as Greenwich Mean Time (GMT) (Zulu). Both **notBefore** and **notAfter** may be encoded as either Coordinated Universal Time (**UTCTime**) or **GeneralizedTime**.[2] Dates through the year 2049 shall be encoded as **UTCTime**, and dates in 2050 or later shall be encoded as **GeneralizedTime**. **UTCTime** shall include a two-digit year, two-digit month, two-digit day, two-digit hours, two-digit minutes, and two-digit seconds terminated by a "Z" (i.e., YYMMDDHHMMSSZ). **GeneralizedTime** shall include a four-digit year, two-digit month, two-digit day, two-digit hours, two-digit minutes, and two-digit seconds terminated by a "Z" (i.e., **YYYYMMDDHHMMSSZ**). The **UTCTime** year value shall be interpreted as follows:
   - If YY is equal to or greater than 50, the year shall be 19YY.
   - If YY is less than 50, the year shall be 20YY.

f. The ***Name*** in the ***subject*** field shall contain the X.500 DN of the subject. CAs and recipients are required to support the following standard naming attribute types: country, organization, organizational-unit, distinguished name qualifier, state or province name, common name, and serial number. Unique subject names are accomplished through careful assignment by CAs, as there are few systems.
g. The **subjectPublicKeyInfo** field shall contain the subject's public key. CAs and recipients are required to generate and processing, respectively, the sub-fields of **subjectPublicKeyInfo** as follows:

[2] The Distinguished Encoding Rules (DER) allow several methods for formatting UTCTime and GeneralizedTime. All implementations shall use the same format to minimize signature verification problems. To ensure that UTCTime and GeneralzedTime values are consistently formatted:

1. The "Z" format and shall always be employed; a time differential shall never be employed.

2. The seconds field (even when it is '00') shall be present.

3. GeneralizedTime values shall not include fractional seconds.

- **algorithmIdentifier** shall include the algorithm's parameters in self-signed CA certificates and may include the parameters in CA and End Entity certificates. If parameters are included in this field they shall be used during signature verification.
- **subjectPublicKey** shall include the subject's public key.

h. The **issuerUniqueIdentifier** field should not be generated within certificates used in the demonstrator.

i. The **subjectUniqueIdentifier** should not be generated within certificates used in the demonstrator.

j. The **extension** field shall be generated. See Section A.7.9.6.3 for a description of certificate extensions.

k. The issuer's signature shall be contained in the fields produced with the SIGNED Macro. The OID of the signature algorithm used to generate the signature shall also be included and must match the value present in paragraph c, above. Inclusion of the parameters is as per Table A.1. However, these parameters shall not be used to verify signatures.

A.7.9.6.3 Public Key Certificate Extensions

Public Key Certificate Extensions are as follows:

a. The ***authorityKeyIdentifier*** extension shall be included in "user" and CA certificates to indicate which of the issuer's keys was used to sign the subject's certificate. It is optional in self-signed CA certificates. The ***keyIdentifier*** choice hall be used as the identifier method. Various methods exist for creating the value; however, the value in field is not regenerated by EE applications therefore CAs may either derived the value from the key or a method that generates a unique value be used. EEs shall support processing this extension.

b. The ***subjectKeyIdentifier*** extension shall be included in all certificates. The ***keyIdentifier*** shall be generated as specified above. EEs shall support processing this extension.

c. The ***keyUsage*** extension shall be included in EE and CA certificates, and it shall be marked as a critical extension. This extension constrains the keying material in the certificate to a specific purpose. CA certificates shall have the ***keyCertSign, cRLSign***, ***digitalSignature***, and ***nonRepudiation*** bit set. In EE certificates, the ***digitalSignature***, ***nonRepudiation, keyEncipherment*** and ***dataEncipherment*** bits shall be set. There is no requirement on other bits. EEs shall support processing this extension.

d. The ***basicConstraints*** extension shall be included in all CA certificates and it shall be marked critical. Although processing of this extension in a "trusted" certificate is not required in the CMDE, there exists the possibility that some certificate chain processing developers may write their applications to require every certificate in a chain, including the "trusted" certificate (possibly a self-signed certificate), to assert that they are CAs in a basic constraints extension. For this reason, this extension shall be included in self-signed CA certificates. The ***cA*** BOOLEAN flag shall be set to true in the CA and self-signed CA certificates. The extension is optional in EE certificates; however, if a EE certificate has ***keyUsage*** as critical and ***keyCertSign*** not set, then the EE certificate should not be confused with a CA certificate (i.e., a certificate signed by an EE with a critical ***keyUsage*** extension and ***keyCertSign*** not set should be rejected as a CA certificate). EEs shall support processing this extension.

A.7.9.6.4 Public Key Certificate Checking

Section 6 of RFC 3280 specifies a procedure for performing certification path verification. An implementation shall be functionally equivalent to the external behaviour resulting from that procedure. The algorithm used by a particular implementation to derive the correct outputs from the given inputs is not standardized herein.

A.7.9.7 Certificate Revocation Lists

A.7.9.7.1 Certificate Revocation List Profiles

This section specifies the Version 2 (V2) X.509 Certificate Revocation List (CRL) as described in Recommendation X.509 (1997) and profiled in RFC 3280. One "type" of CRL is needed to support the demonstrator architecture. The "profile" contained within this document is considered to be compliant to RFC 3280 and is identical to the NATO CRL profile. The following are the additional constraints:

- All CAs are required to generate CRLs, as it is the agreed revocation mechanism.
- RFC 3280 requires the ability to process ***version*** 1 CRLs; however, testing of this support is unnecessary, as all CRLs issued by CA supporting the demonstrator architecture will be issuing ***version*** 2 CRLs.

The following two paragraphs describe specific implementation details for CRLs fields and extensions used to intercommunicate in this environment. CA implementation guidance is provided for generating each field and extension and EE implementation guidance is also provided for processing each field and extension. Annex B provides a detailed CRL profile.

A.7.9.7.2 Certificate Revocation List Fields

CRL fields are as follows:

a. CRLs created for use within this environment are based on the 1997 version of the X.509 standard. ***version*** shall have a value of 2.
b. The ***AlgorithmIdentifier*** in issuer's ***signature*** field shall include the OID as per Table 7-2 of the signature algorithm used to sign the CRL. Inclusion of the parameters is as per Table-7.1.
c. The issuer's ***Name*** in the ***issuer*** field shall contain the unique X.500 DN identifying the issuer of the CRL. CAs and EEs are required to support the following standard naming attribute types: country, organization, organizational-unit, distinguished name qualifier, state or province name, common name, and serial number.
d. CAs and EEs shall support the ***thisUpdate*** field. The CRL ***thisUpdate*** field contains one date that shall be expressed as Greenwich Mean Time (GMT) (Zulu). ***thisUpdate*** may be encoded as either Coordinated Universal Time (***UTCTime***) or ***GeneralizedTime***.[3] Dates through the year 2049 shall be encoded as ***UTCTime***, and dates in 2050 or later shall be encoded as ***GeneralizedTime***. ***UTCTime*** shall include a two-digit year, two-digit month, two-digit day, two-digit hours, two-digit minutes, and two-digit seconds terminated by a "Z" (i.e., YYMMDDHHMMSSZ). ***GeneralizedTime*** shall include a four-digit year, two-digit month, two-digit day, two-digit hours, two-digit minutes, and two-digit seconds terminated by a "Z" (i.e., **YYYYMMDDHHMMSSZ**). The ***UTCTime*** year value shall be interpreted as follows:

[3] The Distinguished Encoding Rules (DER) allow several methods for formatting UTCTime and GeneralizedTime. All implementations shall use the same format to minimize signature verification problems. To ensure that UTCTime and GeneralzedTime values are consistently formatted:

1. The "Z" format and shall always be employed; a time differential shall never be employed.

2. The seconds field (even when it is '00') shall be present.

3. GeneralizedTime values shall not include fractional seconds.

- If YY is equal to or greater than 50, the year shall be 19YY.
- If YY is less than 50, the year shall be 20YY.

e. CAs and EEs shall support the ***nextUpdate*** field. The CRL ***nextUpdate*** field contains one date that shall be expressed as Greenwich Mean Time (GMT) (Zulu). ***nextUpdate*** may be encoded as either Coordinated Universal Time (***UTCTime***) or ***GeneralizedTime***.[4] Dates through the year 2049 shall be encoded as ***UTCTime***, and dates in 2050 or later shall be encoded as ***GeneralizedTime***. ***UTCTime*** shall include a two-digit year, two-digit month, two-digit day, two-digit hours, two-digit minutes, and two-digit seconds terminated by a "Z" (i.e., YYMMDDHHMMSSZ). ***GeneralizedTime*** shall include a four-digit year, two-digit month, two-digit day, two-digit hours, two-digit minutes, and two-digit seconds terminated by a "Z" (i.e., YYYYMMDDHHMMSSZ). The ***UTCTime*** year value shall be interpreted as follows:

- If YY is equal to or greater than 50, the year shall be 19YY.
- If YY is less than 50, the year shall be 20YY.

f. CAs and EEs shall support the ***revokedCertificates*** field. The ***revokedCertificates*** field shall contain a sequence of revoked certificates, when the CA has revoked certificates. If the CA has no revoked certificate, but must generate a CRL, then the ***revokedCertificates*** will be omitted. The structure of this list shall be a sequence of certificate serial numbers and a ***revocationDate***. The revocation date expressed as Greenwich Mean Time (GMT) (Zulu), field may be encoded as either Coordinated Universal Time (***UTCTime***) or ***GeneralizedTime***.[4] Dates through the year 2049 shall be encoded as ***UTCTime***, and dates in 2050 or later shall be encoded as ***GeneralizedTime***. ***UTCTime*** shall include a two-digit year, two-digit month, two-digit day, two-digit hours, two-digit minutes, and two-digit seconds terminated by a "Z" (i.e., YYMMDDHHMMSSZ). ***GeneralizedTime*** shall include a four-digit year, two-digit month, two-digit day, two-digit hours, two-digit minutes, and two-digit seconds terminated by a "Z" (i.e., YYYYMMDDHHMMSSZ). The ***UTCTime*** year value shall be interpreted as follows:

- If YY is equal to or greater than 50, the year shall be 19YY.
- If YY is less than 50, the year shall be 20YY.

g. The *crlExtensions* shall be generated as specified in Appendix 3.

h. The *issuer's signature* shall be contained in the fields produced with the SIGNED Macro. The OID of the signature algorithm used to generate the signature shall also be included. Inclusion of the parameters is as per Table-7.1. However, these parameters shall not be used to verify signatures.

A.7.9.7.3 Certificate Revocation List Entry Extensions

Certificate Revocation List Entry Extensions are:

a. The ***reasonCode*** extension is optional for both processing and generating in a CRL entry. The following reasons for revocation shall be indicated if this extension is supported: ***unspecified***, ***keyCompromise***, ***cACompromise***, ***affiliationChanged***, ***superseded***, and ***cessationOfOperation***. Each entry on a ***distributionPoint*** may indicate whether the certificate was revoked for ***keyCompromise*** or ***cAComprmise***.

b. The ***instructionCode*** extension is optional for both processing and generating in a CRL entry. Placing certificates on hold will not be recognized across international boundaries. Implementations checking CRLs are only concerned with whether a certificate is on the CRL or not.

c. The ***invalidityDate*** extension is optional for both processing and generating in a CRL entry. The invalidity date will not be recognized across international boundaries. Implementations checking the CRL are only concerned with whether a certificate is on the CRL not when the user may have considered the certificate invalid.

d. The ***certificateIssuer*** extension is optional for processing and generating in a CRL entry. It shall be included in distribution point entries and it shall be marked as critical. The name form shall be ***directoryName*** and it shall be equal to the ***issuer*** field of the revoked certificate. This entry extension is optional in full CRLs (i.e., an authority does need to generate this extension in a CRL if it revokes all of its own certificates and no other authority's certificates.)

A.7.9.7.4 Certificate Revocation List Extensions

CRL Extensions are as follows:

a. The **authorityKeyIdentifier** extension shall be included in CRLs to indicate which of the issuer's keys was used to sign the CRL. The **keyIdentifier** field shall be used to identify the key. The identifier shall be constructed as described in paragraph 534 a, which describes the certificate's **authorityKeyIdentifier** extension. All EEs shall support processing this extension.

A.7.9.7.5 Certificate Revocation List Checking

Section 6 of RFC 3280 specifies a procedure for performing certification path verification, which includes verification of CRLs. An implementation shall be functionally equivalent to the external behaviour resulting from that procedure. The algorithm used by a particular implementation to derive the correct outputs from the given inputs is not specified herein.

A.7.9.8 Cryptography

This section specifies algorithm information, which must be consistent and interoperable between the National systems.

At the application layer, currently only two algorithms are required to enable the agreed security services: a hash algorithm and a digital signature algorithm. Specific information for the agreed hashing, Secure Hash Algorithm (SHA-1), and digital signature algorithms, The RSA Algorithm[4], is included herein. Implementations are to support generating and verifying RSA.

A.7.9.9 Hash

The SHA –1 hashing algorithm SHALL be used for the demonstrator.

[4] Note that there are two versions of the RSA Algorithm commonly referred to as the X9 version and PKCS v1.5. The version included herein is the PKCS v1.5 certificate.

Table A.1: Hashing Algorithm

Specification	FIPS PUB 180-1, Secure Hash Standard 17 April 1995 NIST
OID	iso(1) identified-organization(3) oiw(14) secsig(3) algorithm(2) SHA-1(26) 1.3.14.3.2.26 PARAMETER NULL NIST OSI Implementers Workshop, Security Special Interest Group, Stable Implementation Agreements: Part 12 - OS Security, June 1995 NULL parameters shall never be present when an id-sha1 Algorithm Identifier is encoded. (future) X9.57, Appendix 3
Block size	512-bits, 64-bytes
Hash value size	160-bits, 20-bytes
Padding	512-bits, 64-bytes A message has length l < 2^64. Before it is input to the SHA-1, the message is padded on the right as follows: a. "1" is appended. Example: if the original message is "01010000", this is padded to "010100001". b. "0"s are appended. The number of "0"s will depend on the original length of the message. The last 64-bits of the last 512-bit block are reserved for the length l of the original message. c. Obtain the 2-word representation of l, the number of bits in the original message. If l < 2^32 then the first word is all zeroes. Append these two words to the padded message. The padded message will contain 16 * n words for some n > 0. The padded message is regarded as a sequence of n blocks M(1) , M(2), ... , M(n), where each M(i) contains 16 words and M(1) contains the first characters (or bits)of the message.

A.7.9.9.1 Digital Signature

RSA with SHA-1 (PKCS #1 version).

Table A.2: Certificate

Specification	PKCS #1 RSA Encryption Standard ,Version 1.5, 1 November 1993 and FIPS PUB 180-1, Secure Hash Standard, 17 April 1995, NIST
OID	iso(1) member-body(2) US(840) rsadsi(113549) pkcs(1) pkcs-1(1) sha1WithRSAEncryption(5) 1.2.840.113589.1.1.5 PARAMETER NULL PKCS #1 RSA Encryption Standard, Version 1.5, 1 November 1993 and FIPS PUB 180-1, Secure Hash Standard, 17 April 1995, NIST
Hash Encapsulation	A block type *BT*, a padding string *PS*, and the data *D* shall be formatted into an octet string *EB*, the encryption block. EB = 00 || BT || PS || 00 || D . The block type BT shall be a single octet -- value 01. The padding string PS shall consist of k-3-||D|| octets -- they shall have value FF PKCS #1 RSA Encryption Standard, Version 1.5, 1 November 1993
ASN.1 encoding	AN RSA signature (an INTEGER) is conveyed in a BIT STRING in the obvious way: the most significant bit of the INTEGER becomes the most significant bit of the BIT STRING, and the least significant bit of the INTEGER becomes the least significant bit of the BIT STRING. ISO/TC68/SC2/WG8 1997-08-18 ISO/WD-15782 Banking - Certificate Management

Table A.3: Algorithm

OID	iso(1) member-body(2) US(840) rsadsi(113549) pkcs(1) pkcs-1(1) rsaEncryption(1) 1.2.840.113549.1.1.1 PARAMETER NULL PKCS #1 RSA Encryption Standard, Version 1.5, 1 November 1993
Subject Public Key	RSAPublicKey ::= SEQUENCE { modulus INTEGER, -- called n publicExponent INTEGER } -- called e PKCS #1 RSA Encryption Standard, Version 1.5, 1 November 1993
Subject Public Key Length	The key shall be 1024 bits in length.

A.7.9.9.2 Compromised Key Lists

Compromised Key Lists will not be used between nations.

A.7.9.10 Available Freeware

OpenSSL: http://www.openssl.org/ (contains i.a. Certificate Authority (CA) software)

A.7.10 Directory

A.7.10.1 Main Principles

The main principles for the design of the directory are the following:

- The directory is implemented using the LDAP technology;
- The standard used for LDAP is LDAP V3;
- Each nation deploys its own LDAP server;
- Each nation is free of the choice of its LDAP COTS;
- Each nation populates its LDAP server with its own data (certificates, etc) under its branch;
- Each nation has a copy of the branches of the other nations (master to master replication with the responsibility of one main branch attributed to only one LDAP server). This copy is done using a replication software component able to forward directory information (LDIF format) cross the SOAP/SPC guards. This replication component uses the NATO Data Publishing Network deployed between the nations.

A.7.10.2 LDAP DIT

The Directory Information Tree (DIT) are organised as follow:

- The DIT is prefixed with a list of domain component (dc) names as described in the RFC 2247. The top-level dc is NATO and the second level is IST061
- A country (C) is defined for each nation, e.g. c=fr or c=nl. These organisations names are children of the dc prefix. Thus, for instance, the path to the Poland part of the DIT is c=pl,dc=IST061, dc=NATO;
- The part of the DIT attached to a nation is relevant to this nation.

A.7.10.3 LDAP Servers Deployment

All the nations deploy an LDAP server. An LDAP server deployed by a nation contains:

- Its own data stored under its branch (certificates, CRL, etc),
- A copy of the data extracted from the other LDAP servers of the other nations. These copies are stored under the branch of these nations.

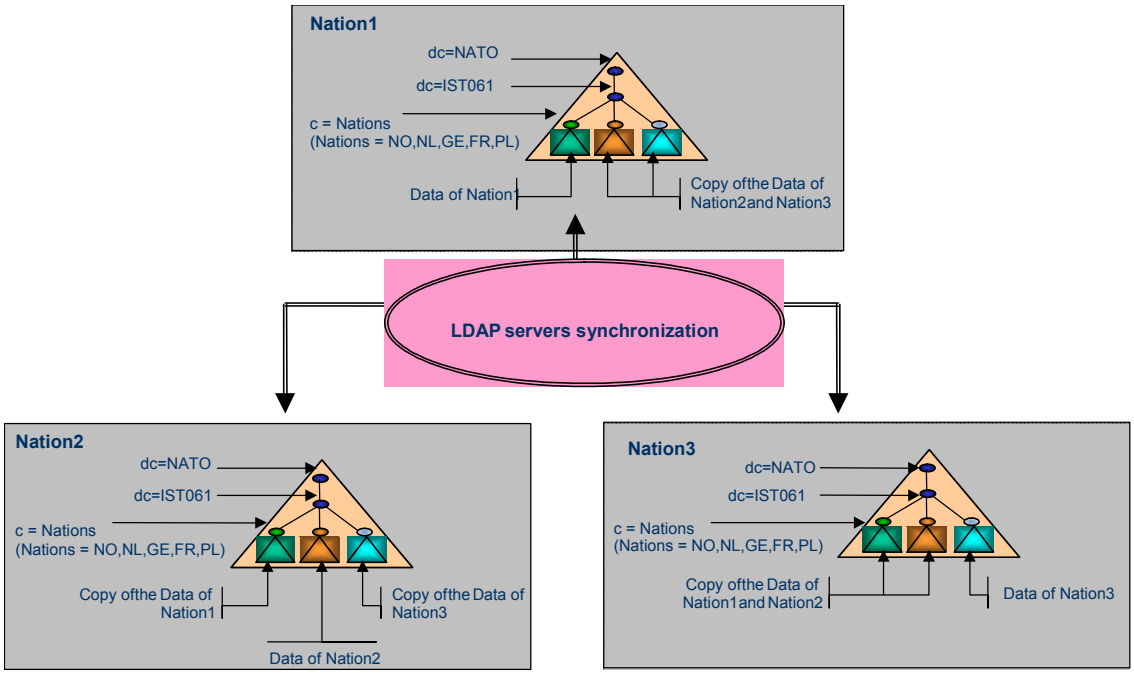

Figure A.45: The Figure Illustrates this Principle with the Deployment of 3 Nations.

Remark: This manner to deploy the LDAP infrastructure with no central management and point of failure is closed to the view explained into the NNEC documents (NNEC is based on the federation of systems paradigm).

A.7.10.4 Replication

A.7.10.4.1 Design Aspects to Take into Account

The design aspects to take into account for the replication between the LDAP servers are the following :

- There is no LDAP replication standard. The replication mechanisms supplied by the available COTS are based on proprietary solutions, i.e. LDAP messages generated by the slurpd daemon for OpenLDAP, SMTP messages for Active directory. Thus, it becomes very hard to synchronise LDAP servers when they are not based on the same COTS (i.e. from an OpenLdap server for a nation to an Active Directory for another nation).
- One possible solution is the use of a meta directory COTS which provides advanced capabilities like transformation of DIT or LDAP entries. This solution has the following problems :
  - These COTS are very expensive;
  - The security solutions deployed in the demonstrator are based on the use of SOAP/XML technologies (SOAP guard/SPC) while these COTS use LDAP requests.
  - Some LDAP servers are difficult to synchronise, e.g. OpenLdap, because these COTS do not provide a log accessible with LDAP requests.

A.7.10.4.2 Synchronisation Design

The design for the synchronisation of the LDAP servers is the following:

- A synchronisation component is deployed in front end of the LDAP server deployed by each nation;
- Periodically, this synchronisation component extracts all the content of the branch of the nation under an LDIF Form as described in the RFC 2849;

- The interoperable Publish/Subscribe services is used to send/receive this extracted content to the other nations. The configuration of the publish/subscribe networking for this synchronisation is as follow :
  - A technical Topic named LDAP_SYNCHRONISATION is supplied. The payload attached to this technical topic is the LDIF content.
  - A technical COI named SYNCHRONISATION_BETWEEN_NATIONS is supplied.
- On the reception of a notification message containing an instance of the LDAP_SYNCHRONISATION topic generated by a nation, the synchronisation component populates the corresponding branch in its LDAP directory with the LDIF payload.
- Please see Appendix 9 for the XML schema to the LDAP synchronization component.

Remarks :
- Using the publish/subscribe services networked with the Secured Web Services infrastructure of the demonstrator, it becomes possible to easily drop a synchronisation message in a secure SOAP message.
- This design illustrates the fact that it is possible to integrate legacy applications with advanced mechanisms (NEC like applications).
- In the demonstrator, there is no need to deploy complex synchronisation mechanisms based on the transmission of deltas because the number of entries in the LDAP servers is small.

A.7.10.4.3 Synchronisation Period

The synchronisation period must be easily configurable to adapt it to the deployment, i.e. number of nations, number of entries in the LDAP servers, efficiency to import data in the LDAP servers.

A nation may decide to send the content of its branch before to wait to the next period, i.e. to reflect immediately to the other nations that a certificate becomes revoked. (i.e. a manually total update.)

A.7.10.4.4 Bootstrapping/Handshake

The use of the secured publish/subscribe services to synchronise the LDAP servers of the nations imposes to know the mandatory bootstrapping certificates. These certificates cannot be known using the synchronisation component. So, it is mandatory for a nation to give the certificate attributed to the synchronisation component to the other nations using the mechanism used to give the PKI root certificate. Bootstrapping may be performed by exchanging the initial LDIF files using e-mail or similar.

A.7.10.5 LDAP Demonstrator Profile

If OpenLDAP is used, version 2.2.28 (released September 2005) SHALL be used [OpenLDAP].

A.7.10.6 Directory Schema

If OpenLDAP is used, the OpenLDAP core schema SHALL be used.

Remark: Since we permit use of different directory systems, e.g. AD, we may encounter some schema related issues. However, since we do not have any strict schema requirements, we should be able to solve these by modifying the schemas “on demand”.

A.7.10.7 PKI Management Support

A PKI will be established for the national systems. The directory service will support the use of PKI(s) between nations by publishing the information necessary for path validation of certificates and the

distribution of Certification Authority's certificate revocation list (CRL). Certificate Authority entry distinguished names shall align with the name stored in the issuer field of the CA certificate.

Root CA entry – This entry must be based on a structural object class, which can incorporate the **pkiCA** auxiliary object class. e.g. organizationalRole. This entry is used to store the CRL and ARL.

Intermediate CA entry - This entry must be based on a structural object class, which can incorporate the **pkiCA** auxiliary object class. This entry is used to store the certificate and CRL.

A.7.10.7.1 Systems support

End Entity PKI support – This entry must be based on a structural object class, which can incorporate the **pkiUser** auxiliary object class. e.g. organizationalPerson. This entry is used to store the End Entity certificate.

A.7.10.7.2 PKI Object Classes

pkiCA
The pkiCA object class, defined in ITU-T Rec. X.509 | ISO/IEC 9594-8 DAM 1, SHALL be used in defining directory entries for Certification Authorities. The table below shows the composition of the pkiCA object class.

pkiCA Object Class

Attribute	m/o
X.509: authorityRevocationList	o
X.509: cACertificate	o
X.509: certificateRevocationList o	o
X.509: crossCertificatePair	-

pkiUser
The pkiUser object class SHALL be used in defining directory entries for objects that include user certificates, as defined in ITU-T Rec. X.509 | ISO/IEC 9594-8 DAM 1. The table below shows the composition of the pkiUser object class.

pkiUser Object Class

Attribute	m/o
X.509: userCertificate	o

A.7.10.8 Available Freeware

- OpenLDAP: http://www.openldap.org/
- Version 2.2.28 Win32 binaries:
  http://bergmans.us/list/openldap-windows/2005/09/msg00003.html
- Admin guide for OpenLDAP:
  http://www.openldap.org/doc/admin23/
- JXplorer 3.1 LDAP Browser/client:
  http://www.jxplorer.org

A.7.11 Demonstrator Security Policy Identifier

The policy identifier of the security labels SHALL be set to "RTO-IST-061-Mission".

A.7.12 References

[1] [WS-Security 2004-OASIS 200401] Web Services Security: SOAP Message Security 1.0 (WS-Security 2004).

[2] [WS-Addresing-Core] http://www.w3.org/TR/ws-addr-core.

[3] [OpenLDAP] http://www.openldap.org/, Version 2.2.28 Win32 binaries: http://bergmans.us/list/openldap-windows/2005/09/msg00003.html.

[4] [RFC 2247]: Using Domains in LDAP/X.500 Distinguished Names: ftp://ftp.rfc-editor.org/in-notes/rfc2247.txt.

[5] [RFC 2849]: The LDAP Data Interchange Format (LDIF) - Technical Specification: ftp://ftp.rfc-editor.org/in-notes/rfc2849.txt.

A.8 COMPRESSION TECHNIQUES

This chapter contains the specification of data compression applied to SOAP messages for the CWID'2006 demonstrator. Please refer to previous versions of this chapter for an introduction and overview of related issues, existing tools and techniques. Appendix 10 contains performance analysis of a few most promising compression methods.

A.8.1 Introduction

The research and implementation works, performed by the RTG-027 Working Group, are trying to address issues related to employing the SOA architecture and related technologies (i.e., SOAP/Web Services) in limited-bandwidth networks. To address the problem of communication overhead, caused by using XML for communication, the Group has decided to implement and demonstrate data compression techniques, together with an analysis of its influence on the overall demonstrator's performance.

Note that enabling data compression does not necessarily improve the overall communication performance (that is, the request–response delay). Compression trades smaller message sizes (and thus better communication performance) for increased processing overhead at clients and servers. Thus, generally, compression improves performance in bandwidth-limited environments while it may impair performance in high-speed networks.

A.8.2 Compression of a SOAP Message

For every SOAP message, compression is performed for these elements of the SOAP envelope, which are selected for encryption. That is, both compression and encryption on the sender side (as well as decryption and decompression on the receiver side) should be considered as a single, complex step of data processing, with binary compressed data being an immediate input to encryption (or, with binary decrypted data being an immediate input to decompression on the receiver side).

The following remarks should be made:

1. Enabling compression is optional; if enabled, it may select one of the compression methods specified in Section A.8.3.
2. Compression is configured at the demonstrator's startup, through a configuration file, and does not change during the demonstration.
3. It is assumed that the compression method is known in advance and no additional headers or markers, indicating the compression type, are added to the SOAP message being processed[5].
4. It is assumed that compression, if enabled, will always be executed even for small messages[6]. Speaking in other words, there is no on-the-fly decision whether data should be compressed or left plain.

A.8.2.1 Motivation

Compression of selected SOAP Envelope elements is similar to encryption, as defined in the specification [OASIS-WSS]; except that for compression, no additional secret data are involved. In both cases, a SOAP message's fragment (and XML element) is replaced with binary data, possibly of a different length (due to compression or padding for encryption). Then, the binary data must be encoded – usually, using Base64 – to keep the SOAP envelope valid. This, unfortunately, causes the message size to grow (the encoding replaces every 3 bytes with 4 new ones) and the gain from compression may be diminished or even lost.

[5] This choice has been made for simplicity but of course it remains open to discussion. Additionally, in most cases, it is possible to dynamically detect the compression method by just looking at a few first bytes of the message.

[6] See above. It was verified during compression performance tests that both selected compression methods are able to further shrink even quite small messages (120 bytes).

Due to simplicity, we do not define any extension mechanisms. Additionally, performing two separate steps (e.g., compress, encode, then encrypt, and encode again) does not seem to be the desired way. Instead, for each selected XML element, the encryption should be performed just after the compression, i.e., it should take the binary output from the compression step. Such an approach should improve the encryption (and decryption) performance due to smaller number of bytes to encrypt.

It would also be possible to compress (or, encode) the whole SOAP message after it has been prepared for supplying to the transport layer but the Group decided not to implement this functionality in the demonstrator.

A.8.3 Compression Methods

Two data compression algorithms have been selected for the demonstrator:

1. GZip – the standard, general-purpose compression algorithm;
2. XMill – XML-specific data compressor.

It has been verified through extensive tests that both algorithms are able to significantly reduce sizes of XML documents, even for small ones (120-200 bytes). In terms of data compression and decompression times, GZip is definitely faster. Please refer to Appendix X for performance data.

A.8.4 Configuration

The following remarks should be made:

1. Compression configuration is read from an appropriate demonstrator's configuration file.
2. Compression method MAY be selected, and additional options MAY be specified. If the compression method is not specified, compression is disabled.
3. The compression method is specified using an `XmlZipper` XML element in the configuration file.
4. Optional options are specified using a number of additional elements: `XmlZipperOptionCount`, which contains the number of options, and `XmlZipperOption`*N*, with *N* = 0..<number of options – 1>, containing subsequent options[7]. If the `XmlZipperOptionCount` element is missing, no options are assumed (this is equivalent to `XmlZipperOptionCount` = 0).
5. An exemplary configuration file with compression disabled should look like this:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <appSettings>
    <add key="XmlZipper" value=""/>
    <add key="XmlZipperOptionCount" value="0"/>
  </appSettings>
</configuration>
```

6. An exemplary configuration file with GZip-based compression enabled should look like this:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <appSettings>
    <add key="XmlZipper" value="gzip"/>
    <add key="XmlZipperOptionCount" value="0"/>
  </appSettings>
</configuration>
```

[7] This looks quite primitive but allows to avoid parsing and splitting options, necessary when a single option string would be used.

7. An exemplary configuration file with XMill-based compression enabled should look like this:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <appSettings>
    <add key="XmlZipper" value="xmill"/>
    <add key="XmlZipperOptionCount" value="3"/>
    <add key="XmlZipperOption0" value="-P"/>   <!— ppmdi compression     -->
    <add key="XmlZipperOption1" value="-9"/>   <!- 9th compression level -->
    <add key="XmlZipperOption2" value="-on"/> <!- no output formatting  -->
  </appSettings>
</configuration>
```

A.8.5 References

[1] [OASIS-WSS] Web Services Security: SOAP Message Security 1.0, OASIS Standard 200401, March 2004. Available: http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf.

[2] [SharpZip] http://www.icsharpcode.net/OpenSource/SharpZipLib/.

[3] A popular .NET open-source compression library containing GZip, BZip and Zip compressors.

[4] [XMill] http://sourceforge.net/projects/xmill.

[5] XMill source code. Note that it contains a plethora of memory bugs, so the original source had to be debugged and corrected.

A.9 OTHER ISSUES

A.9.1 Time Zone

To avoid problems with time zones it is RECOMMENDED that UTC (GMT/ZULU) is used.

Appendix 1: tModels

This specification introduces a number of canonical tModels that are used to represent information about services, assets, nations and COIs in the service registry. These tModels are defined here.

A1.1 IDENTIFICATION STRING TMODEL

A tModel used to identify businessEntities such as Nations, Assets and COIs. The tModel structure is shown below.

```
<tModel tModelKey="uddi:5B595F40-6BB4-11DA-9F40-981D933BF1E5">
  <name>IdentificationString</name>
  <description xml:lang="en">Identification string identifier system</description>
  <overviewDoc>
    <overviewURL>http://link.to.spec</overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="unchecked"/>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="identifier"/>
  </categoryBag>
</tModel>
```

The keyValue attribute must be unique within the UDDI registry to ensure that publication can be performed correctly. To ensure uniqueness, each nation SHALL generate keyValues that start with the two letter nation code followed by colon. Each nation is responsible for ensuring this uniqueness within their national domain when publishing businessEntities. Examples of a keyValues:

- No:asset:frigate:nansen

When a reference to this tModel is used, the keyName attribute MUST have the value "`IdentificationString`".

A1.2 SERVICE TAXONOMY TMODEL

A tModel used to classify services. The tModel structure is shown below.

```
<tModel tModelKey="uddi:102CBCF0-4C6B-11DA-BCF0-9D94332B2CEE">
  <name>ServiceTaxonomy</name>
  <description xml:lang="en">Service category system</description>
  <overviewDoc>
    <overviewURL>http://link.to.spec</overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="unchecked"/>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="categorization"/>
  </categoryBag>
</tModel>
```

While this is an unchecked category system, the allowed values to be used with this category system are:

KeyValue	Description
Sensor	Classifies the service as a Sensor

When a reference to this tModel is used, the keyName attribute MUST have the value "`ServiceType`".

A1.3 COVERAGEAREA TMODEL

This tModel is used to group two references to the Longitude tModel and two references to the Latitude tModel together to indicate that these tModels constitute the upper left and lower right corner in a rectangle. Which Longitude tModel that belongs to the upper left or lower right point is indicated by the value in the keyName attribute in the reference to the corresponding tModel. The same applies for the Latitude tModel. The tModel structure is shown below.

```
<tModel tModelKey="uddi:72CDEA00-4C6B-11DA-AA00-CDECD53229D7">
  <name>CoverageArea</name>
  <description xml:lang="en">Coverage area category system</description>
  <overviewDoc>
    <overviewURL>http://link.to.spec</overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="unchecked"/>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue=" categorizationGroup"/>
  </categoryBag>
</tModel>
```

A1.4 LONGITUDE TMODEL

This tModel is used to indicate a longitude. The tModel structure is shown below.

```
<tModel tModelKey="uddi:A1DC1E20-4C6B-11DA-9E20-A10FD9209C35">
  <name>Longitude</name>
  <description xml:lang="en">The longitude of a geodetic point (wgs84)</description>
  <overviewDoc>
    <overviewURL>http://link.to.spec</overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="unchecked"/>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue=" categorization"/>
  </categoryBag>
</tModel>
```

Valid values for this category system are decimal degrees in the range -180 degrees to 180 degrees.

The keyName attribute MUST either have the value "`upperLeft`" or "`lowerRight`", indicating which corner of the coverage area rectangle this longitude is part of.

A1.5 LATITUDE TMODEL

This tModel is used to indicate a latitude. The tModel structure is shown below.

```
<tModel tModelKey="uddi:9C707BC0-4C6B-11DA-BBC0-B4DACA2160E4">
  <name>Latitude</name>
  <description xml:lang="en"> The latitude of a geodetic point (wgs84)</description>
  <overviewDoc>
    <overviewURL>http://link.to.spec</overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="unchecked"/>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue=" categorization"/>
  </categoryBag>
</tModel>
```

Valid values for this category system are decimal degrees in the range -90 degrees to 90 degrees.

The keyName attribute MUST either have the value "`upperLeft`" or "`lowerRight`", indicating which corner of the coverage area rectangle this longitude is part of.

A1.6 POSITION TMODEL

A tModel used to specify the position of the platform the service belongs to. The tModel structure is shown below.

```
<tModel tModelKey="uddi:7F663FB0-4C6B-11DA-BFB0-9985F7CAF2E1">
  <name>Position</name>
  <description     xml:lang="en">Represents     the     dynamic     position     of     the
service.</description>
  <overviewDoc>
    <overviewURL>http://link.to.spec</overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="unchecked"/>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="categorization"/>
  </categoryBag>
</tModel>
```

The valid values for this category system are URL addresses. These addresses points to a location where the position of the sensor can be generated dynamically on the fly.

When a reference to this tModel is used, the keyName attribute MUST have the value "`Position`".

A1.7 PUBLISHED TMODEL

This tModel is used to specify the date and time when the service was published into the service registry. The tModel structure is shown below.

```
<tModel tModelKey="uddi:81A1B0C0-4C6B-11DA-B0C0-9311F1331935">
  <name>Published</name>
  <description xml:lang="en">Represents the publishing date to the
service.</description>
  <overviewDoc>
    <overviewURL>http://link.to.spec</overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="unchecked"/>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="categorization"/>
  </categoryBag>
</tModel>
```

The valid values for this category system are timestamps on the format:
YYYY-MM-DDThh:mm:ss±hh:mm
Where YYYY=year, MM=month, DD=day, hh=hour, mm=minute and ss=second.

When a reference to this tModel is used, the keyName attribute MUST have the value "`Published`".

A1.8 VALID UNTIL TMODEL

This tModel is used to specify the date and time the information about the service in the service registry cease to be valid. The tModel structure is shown below.

```
<tModel tModelKey="uddi:8DA2CF80-4C6B-11DA-8F80-AB91377B4FFC">
  <name>ValidUntil</name>
  <description xml:lang="en">Represents the end of the validity period for the
service.</description>
  <overviewDoc>
    <overviewURL>http://link.to.spec</overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="unchecked"/>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="categorization"/>
  </categoryBag>
</tModel>
```

The valid values for this category system are timestamps on the format:
YYYY-MM-DDThh:mm:ss±hh:mm
Where YYYY=year, MM=month, DD=day, hh=hour, mm=minute and ss=second.

When a reference to this tModel is used, the keyName attribute MUST have the value "`ValidUntil`".

A1.9 ENTITY TYPE TMODEL

This tModel is used to indicate whether a businessEnity represents a Nation, an Asset or a COI. The structure of the tModel is shown below.

```
<tModel tModelKey="uddi:92123790-4C6B-11DA-B790-A4F45DE39EFE">
  <name>EntityType</name>
  <description xml:lang="en">Entity Type category system.</description>
  <overviewDoc>
    <overviewURL>http://link.to.spec</overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="unchecked"/>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="categorization"/>
  </categoryBag>
</tModel>
```

While this is an unchecked category system, there are only three values that should be used with this category system:

KeyValue	Description
Nation	Indicates that the businessEntity referencing this tModel is a nation.
Asset	Indicates that the businessEntity referencing this tModel is an asset.
COI	Indicates that the businessEntity referencing this tModel is a COI.

In all of the above cases, the keyName attribute MUST have the value "`EntityType`".

A1.10 ASSET CATEGORIZATION TMODEL

This tModel is used in the categoryBag of a businessEnity that represents an Asset to indicate what kind of asset this is. The structure of the tModel is shown below.

```
<tModel tModelKey="uddi:962D8C30-4C6B-11DA-8C30-E0B6053F4D2C">
  <name>AssetCategorization</name>
  <description xml:lang="en">Asset category system.</description>
  <overviewDoc>
    <overviewURL>http://link.to.spec</overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="unchecked"/>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="categorization"/>
  </categoryBag>
</tModel>
```

When a reference to this tModel is used, the keyName attribute MUST have the value "`AssetCategorization`".

A1.11 TOPICCATEGORIZATION

A tModel used to classify tModels as a topic space, a root topic or a sub topic. The tModel structure is shown below.

```
<tModel tModelKey="uddi:A41915D0-4C6B-11DA-95D0-BD45C7FAF23E">
  <name>TopicCategorization</name>
  <description xml:lang="en">Topic category system</description>
  <overviewDoc>
    <overviewURL>http://link.to.spec</overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="unchecked"/>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="categorization"/>
  </categoryBag>
</tModel>
```

While this is an unchecked category system, there are only three values that should be used with this category system:

KeyValue	Description
TopicSpace	Indicates that the tModel represents a topic space.
RootTopic	Indicates that the tModel represents a root topic.
SubTopic	Indicates that the tModel represents a sub topic.

In all of the above cases, the keyName attribute MUST have the value "`TopicCategorization`".

A1.12 TOPICSPACEREFERENCE

UDDI does not provide a built-in mechanism to describe a relationship between two tModels. The topicSpaceReference tModel provides a mechanism to indicate that a topic-tModel has a relationship (belonging to) to a certain topicSpace tModel. The structure of the tModel is shown below.

```
<tModel tModelKey="uddi:A63DCA90-4C6B-11DA-8A90-FEE8FA5DB743">
  <name>TopicSpaceReference</name>
  <description xml:lang="en">A category system used to reference a topicSpace tModel.
  </description>
  <overviewDoc>
    <overviewURL>http://link.to.spec</overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="unchecked"/>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="categorization"/>
  </categoryBag>
</tModel>
```

Valid values for this category system are tModelKeys. The content of the keyValue attribute in a keyedReference that refers to this tModel is the tModelKey of the topicSpace tModel being referenced.

When a reference to this tModel is used, the keyName attribute MUST have the value "`TopicSpaceReference`".

A1.13 DISTINGUISHED NAME

This tModel is used to specify the distinguished name associated with a security certificate. The distinguished name can be used to retrieve the certificate from LDAP. This tModel is used in the categoryBag of bindingTemplates.

```
<tModel tModelKey="uddi:767CC130-4C6B-11DA-8130-C5E1F03D1906">
  <name>DistinguishedName</name>
  <description xml:lang="en">Distinguished Name Category system</description>
  <overviewDoc>
    <overviewURL>http://link.to.spec</overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="unchecked"/>
    <keyedReference
        tModelKey="uddi:uddi.org:categorization:types"
        keyName="uddi-org:types"
        keyValue="categorization"/>
  </categoryBag>
</tModel>
```

When a reference to this tModel is used, the keyName attribute MUST have the value “DistinguishedName”.

Appendix 2: Features in UDDI V3

There are many similarities between UDDI V2 and UDDI V3, but there is some extra features available in UDDI V3, which would make the implementation of the service registry easier. These features include:

- Support for digital signatures
- A subscription API
- Support for multi-registry environment
- Better search API (can do single-step instead of multi-step queries as one have to in version 2)
- Possible to categorize bindingTemplate-entities

The support for digital signatures assures the integrity and authenticity of the data stored in the UDDI registry, both for the publisher and the inquirer of the data.

The subscription API makes it possible to track registry activity by subscribing to the entities of interest, and each time something changes about that entity one will get notified. For instance in the case a service goes down, and based on the service termination policies used, its entry in the UDDI registry may get deleted which will trigger a notification to those who has subscribed to that entity. This can again trigger a search for a replacement service.

The support for a multi-registry environment makes it possible to have a more broadly distributed environment. In addition to having a UDDI registry consisting of several nodes, which replicates data among themselves (UDDI V2 and V3), UDDI version 3 also allows data sharing among different registries.

The search API in UDDI version 3 has been improved, so that queries that previously required two or more sub-queries to be satisfied, now can be satisfied using a single query with nested sub-queries, reducing the number of round trips a client must make to a UDDI registry.

The possibility to categorize bindingTemplates allows metadata to be attributed directly to the technical details of a web service. This facilitates more specific searches on web services.

UDDI version 2 and 3 have different key rules. The table below gives an overview.

	UDDI version 2	UDDI version 3
businessKey	<uuid-key>	domain-key OR uddi:<uuid-key>
serviceKey		
bindingKey		
tModelKey	uuid:<uuid-key>	

Appendix 3: PKI Profiles

A3.1 SIGNATURE CERTIFICATES

A3.1.1 Signature Certificate Introduction

This appendix provides the Profile for the Signature Certificates (self-signed CA, CA, and end entities) for use in this environment. The structure for the Certificate is defined in the 1997 version of ITU-T X.509 | ISO/IEC 9594-8.

A3.1.2 Description of Tables

- The "Item" and "Notes" columns are provided for cross-referencing. The numbers in the "Item" column are the row numbers. The numbers in the "Notes" column indicate the table numbers followed by the "item" number enclosed in parentheses. These two columns are used together to point to sub-elements. The "Notes" column also refers to additional information supplied in the last row of the table.
- The "Protocol Elements" column refers to the name of ASN.1 fields taken from the X.500 recommendations.

- In each table, the "Base" column reflects the level of support required for conformance to the base standard[8]. The level of support refers to the support classification for the "Base" column. The "Base" column is broken into "Proc." (i.e., processing) and "Gen." (i.e., generation) columns. The "Proc." column reflects the level of support required by compliant certificate processing and using entities who process certificates. The "Gen." column reflects the level of support required in compliant signature certificates (i.e., the information that is included in the certificate). The types of signature certificates include: (i.e., self-signed CA (see A.2), CA (see A.3), and end-entities (see A.5). When the CA acts as an End Entity (e.g., when a CA verifies the signature on a message), then the "Proc." column applies.[9]

The "Support" column is provided for completion by the supplier of the implementation as follows:

Y the protocol element is fully supported (i.e., supporting the requirements of the m support classification)

N the protocol element is not fully supported, further qualified to indicate the action taken on receipt of such an element as follows:
ND - the element is discarded/ignored
NR - the PDU is rejected

– or blank the protocol element is not applicable

A3.1.3 Support Classifications

- Each of the protocol elements listed in section A.2[10], A.3, A.4, and A.5 are designated as having a support requirement of mandatory or optional. Where protocol elements are nested (i.e., the

[8] If the CCEB defined a certificate extension, field, or attribute not in the base standard (i.e., X.509), the classification for the "Base" column is –.

[9] "Proc." columns in the PAA, PCA, CA, External Domain, and EE certificate tables are identical.

[10] (U) Implementation of clause A.2 is dependent on National policy. Formatting the trusted public key as a self-signed certificate is dependent on National policy (see clause 2.4 of the CMI Authentication Framework).

elements contain sub-elements), the requirement to support the nested element is relevant only when the immediately containing (parent) element is supported.

- To specify the support level of the protocol elements, the following terminology is defined.

A3.1.4 Static Capability

The following classifications are used to specify static conformance (i.e., capability).

- **mandatory support (m):** Implementations claiming to create certificates shall be able to generate the protocol element. Implementations claiming to process certificates shall be able to receive the protocol elements and perform all associated procedures (i.e., implying the ability to handle both the syntax and the semantics of the element) as relevant.
- **optional (o)**: Implementations claiming to create certificates are not required to support generation of the protocol element. If support is claimed, the element shall be treated as if it were specified as mandatory support, and the sub-elements, if present, shall be supported as specified. Implementations claiming to perform processing of certificates shall ignore the protocol element and continue processing of the certificate.
- **conditional (c)**: Implementations shall support the protocol element under the conditions specified. If the conditions are met, the protocol element shall be treated as if it were specified as mandatory support. If these conditions are not met, the protocol element shall be treated as if it were specified as optional support (unless otherwise stated).
- **not applicable** (–) : This element is not applicable in the particular context in which this classification is used.

A3.1.5 Dynamic Capability

The following classifications are used to specify dynamic conformance (i.e., behaviour).

- **required (r):** The information for this protocol element must be populated upon certificate generation.

Table A3.1: Self-Signed CA Signature Certificate

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	signed	m	m	m	mr	
2	toBeSigned	m	m	m	mr	
3	version	m	m	m	mr	
4	serialNumber	m	m	m	mr	
5	signature	m	m	m	mr	See Table A3.2 Note 2
6	issuer	m	m	m	mr	See ACP 133
7	validity	m	m	m	mr	
8	notBefore	m	m	m	mr	See Table A3.46 (1)
9	notAfter	m	m	m	mr	See Table A3.46 (1)
10	subject	m	m	m	mr	See ACP 133
11	subjectPublicKeyInfo	m	m	m	mr	
12	algorithm	m	m	m	mr	See Table A3.2
13	subjectPublicKey	m	m	m	mr	
14	issuerUniqueIdentifier	o	o	-	-	
15	subjectUniqueIdentifier	o	o	-	-	
16	extensions	o	o	m	mr	See Table A3.3
17	algorithmIdentifier	m	m	m	mr	See Note 2
18	encrypted	m	m	m	mr	

Table A3.2: Algorithm Identifier

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	algorithm	m	m	m	mr	
2	parameters	m	m	m	mr	Note 3

Table A3.3: Extensions

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	extnID	m	m	m	mr	Note 4
2	critical	m	m	m	mr	d(false)
3	extnValue	m	m	m	mr	

Table A3.4: Standard Extensions

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	authorityKeyIdentifier	o	o	m	-	See Table A3.5
2	subjectKeyIdentifier	o	o	m	mr	
3	keyUsage	o	o	m	-	See Table A3.36
4	extKeyUsage	o	o	-	-	
5	privateKeyUsagePeriod	o	o	-	-	See Table A3.7
6	certificatePolicies	o	o	-	-	See Table A3.8
7	policyMappings	o	o	-	-	See Table A3.9
8	subjectAltName	o	o	-	-	See Table A3.46 (1), Note 5
9	issuerAltName	o	o	-	-	See Table A3.46 (1), Note 5
10	subjectDirectoryAttributes	o	o	-	-	
11	basicConstraints	o	o	m	mr	See Table A3.10 Note 6
12	nameConstraints	o	o	-	-	See Table A3.11
13	policyConstraints	o	o	-	-	See Table A3.13
14	cRLDistributionPoints	o	o	-	-	See Table A3.14 Note 5
15	authorityInfoAcess	o	o	-	-	See Table A3.15
16	inhibitAnyPolicy	o	o	-	-	
17	subjectInfoAccess	o	o	-	-	
18	freshestCRL	o	o	-	-	

Table A3.5: Authority Key Identifier

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	keyIdentifier	c3	c3	m	mr	Note 7
2	certIssuer	c4	c4	-	-	
3	certSerialNumber	c4	c4	-	-	

Table A3.6: Key Usage

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	digitalSignature	o	o	o	o	Note 8
2	nonRepudiation	o	o	o	o	Note 8
3	keyEncipherment	o	o	–	–	
4	dataEncipherment	o	o	–	–	
5	keyAgreement	o	o	–	–	
6	keyCertSign	o	o	m	mr	
7	cRLSign	o	o	m	m	Note 8
8	encipherOnly	o	o	–	–	
9	decipherOnly	o	o	–	–	

Table A3.7: Private Key Usage Period

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	notBefore	m	c5	-	-	
2	notAfter	m	c5	-	-	

Table A3.8: Certificate Policies

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	policyIdentifier	m	m	-	-	
2	policyQualifiers	o	o	-	-	
3	policyQualifierId	m	m	-	-	
4	qualifier	o	o	-	-	

Table A3.9: Policy Mappings

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	issuerDomainPolicy	m	m	-	-	
2	subjectDomainPolicy	m	m	-	-	

Table A3.10: Basic Constraints

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	cA	m	m	m	mr	d(false)
2	pathLenConstraint	m	o	m	-	

Table A3.11: Name Constraints

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	permittedSubtrees	m	o	-	-	See Table A3.12
2	excludedSubtrees	m	o	-	-	See Table A3.12

Table A3.12: General Subtrees

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	base	m	m	-	-	See Table A3.46 (5)
2	minimum	m	m	-	-	d(0)
3	maximum	m	o	-	-	

Table A3.13: Policy Constraints

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	requireExplicitPolicy	m	o	-	-	
2	inhibitPolicyMapping	m	o	-	-	

Table A3.14: CRL Distribution Points

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	distributionPoint	o	o	-	-	See Table A3.46 (17)
2	reasons	o	o	-	-	See Table A3.46 (20)
3	cRLIssuer	o	o	-	-	See Table A3.46 (4)

Table A3.15: Authority Information Access

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	accessMethod	o	o	-	-	Note 9
2	accessLocation	o	o	-	-	Note 9

Table A3.16: CA Signature Certificate

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	signed	m	m	m	mr	
2	toBeSigned	m	m	m	mr	
3	version	m	m	m	mr	
4	serialNumber	m	m	m	mr	
5	signature	m	m	m	mr	See Table A3.17, Note 10
6	issuer	m	m	m	mr	See ACP 133
7	validity	m	m	m	mr	
8	notBefore	m	m	m	mr	See Table A3.46 (1)
9	notAfter	m	m	m	mr	See Table A3.46 (1)
10	subject	m	m	m	mr	See ACP 133
11	subjectPublicKeyInfo	m	m	m	mr	
12	algorithm	m	m	m	mr	See Table A3.17
13	subjectPublicKey	m	m	m	mr	
14	issuerUniqueIdentifier	o	o	-	-	
15	subjectUniqueIdentifier	o	o	-	-	
16	extensions	o	o	m	mr	See Table A3.18
17	algorithmIdentifier	m	m	m	mr	See Table A3.17, Note 10
18	encrypted	m	m	m	mr	

Table A3.17: Algorithm Identifier

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	algorithm	m	m	m	mr	
2	parameters	m	m	m	m	Note 11

Table A3.18: Extensions

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	extnID	m	m	M	mr	Note 12
2	critical	m	m	M	mr	d(false)
3	extnValue	m	m	M	mr	

Table A3.19: Standard Extensions

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	authorityKeyIdentifier	o	o	M	mr	See Table A3.20
2	subjectKeyIdentifier	o	o	M	mr	
3	keyUsage	o	o	M	mr	See Table A3.21, Note 13
4	extKeyUsage	o	o	-	-	
5	privateKeyUsagePeriod	o	o	-	-	See Table A3.22
6	certificatePolicies	o	o	-	-	See Table A3.23
7	policyMappings	o	o	-	-	See Table A3.24
8	subjectAltName	o	o	-	-	See Table A3.46 (1), Note 14
9	issuerAltName	o	o	-	-	See Table A3.46 (1), Note 22
10	subjectDirectoryAttributes	o	o	-	-	See Table A3.25
11	basicConstraints	o	o	M	mr	See Table A3.25, Note 13
12	nameConstraints	o	o	-	-	See Table A3.26
13	policyConstraints	o	o	-	-	See Table A3.28
14	cRLDistributionPoints	o	o	-	-	See Table A3.29 Note 14
15	authorityInfoAcess	o	o	-	-	See Table A3.30
16	inhibitAnyPolicy	o	o	-	-	
17	subjectInfoAccess	o	o	-	-	
18	freshestCRL	o	o	-	-	See Table A3.29

Table A3.20: Authority Key Identifier

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	keyIdentifier	c9	c9	M	mr	Note 15
2	certIssuer	c10	c10	-	-	
3	certSerialNumber	c10	c10	-	-	

Table A3.21: Key Usage

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	digitalSignature	o	o	M	m	Note 16
2	nonRepudiation	o	o	M	m	Note 16
3	keyEncipherment	o	o	–	–	
4	dataEncipherment	o	o	–	–	
5	keyAgreement	o	o	–	–	
6	keyCertSign	o	o	M	mr	
7	cRLSign	o	o	M	m	Note 16
8	encipherOnly	o	o	–	–	
9	decipherOnly	o	o	–	–	

Table A3.22: Private Key Usage Period

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	notBefore	m	c11	-	-	
2	notAfter	m	c11	-	-	

Table A3.23: Certificate Policies

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	policyIdentifier	m	m	-	-	
2	policyQualifiers	o	o	-	-	
3	policyQualifierId	m	m	-	-	
4	qualifier	o	o	-	-	

Table A3.24: Policy Mappings

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	issuerDomainPolicy	m	m	-	-	
2	subjectDomainPolicy	m	m	-	-	

Table A3.25: Basic Constraints

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	cA	m	m	m	mr	d(false)
2	pathLenConstraint	m	o	m	-	

Table A3.26: Name Constraints

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	permittedSubtrees	m	o	-	-	See Table A3.27
2	excludedSubtrees	m	o	-	-	See Table A3.27

Table A3.27: General Subtrees

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	base	m	m	-	-	See Table A3.46 (5)
2	minimum	m	m	-	-	d(0)
3	maximum	m	o	-	-	

Table A3.28: Policy Constraints

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	requireExplicitPolicy	m	o	-	-	
2	inhibitPolicyMapping	m	o	-	-	

Table A3.29: CRL Distribution Points

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	distributionPoint	o	o	-	-	See Table A3.46 (17)
2	reasons	o	o	-	-	See Table A3.46 (20)
3	cRLIssuer	o	o	-	-	See Table A3.46 (4)

Table A3.30: Authority Information Access

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	accessMethod	o	o	-	-	Note 17
2	accessLocation	o	o	-	-	Note 17

Table A3.31: End Entity Signature Certificate

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	signed	m	m	mr	mr	
2	toBeSigned	m	m	mr	mr	
3	version	m	m	mr	mr	
4	serialNumber	m	m	mr	mr	
5	signature	m	m	mr	mr	See Table A3.32, Note 18
6	issuer	m	m	mr	mr	See ACP 133
7	validity	m	m	mr	mr	
8	notBefore	m	m	mr	mr	See Table A3.46 (1)
9	notAfter	m	m	mr	mr	See Table A3.46 (1)
10	subject	m	m	mr	mr	See ACP 133
11	subjectPublicKeyInfo	m	m	mr	mr	
12	algorithm	m	m	mr	mr	See Table A3.32
13	subjectPublicKey	m	m	mr	mr	
14	issuerUniqueIdentifier	o	o	-	-	
15	subjectUniqueIdentifier	o	o	-	-	
16	extensions	o	o	mr	mr	See Table A3.33
17	algorithmIdentifier	m	m	mr	mr	See Table A3.32, Note 18
18	encrypted	m	m	mr	mr	

Table A3.32: Algorithm Identifier

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	algorithm	m	m	m	mr	
2	parameters	m	m	m	m	Note 19

Table A3.33: Extensions

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	extnID	m	m	M	m	Note 20
2	critical	m	m	M	m	d(false)
3	extnValue	m	m	M	m	

Table A3.34: Standard Extensions

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	authorityKeyIdentifier	o	o	M	mr	See Table A3.35
2	subjectKeyIdentifier	o	o	M	mr	
3	keyUsage	o	o	M	mr	See Table A3.36, Note 21
4	extKeyUsage	o	o	-	-	
5	privateKeyUsagePeriod	o	o	-	-	See Table A3.37
6	certificatePolicies	o	o	-	-	See Table A3.38
7	policyMappings	o	o	-	–	See Table A3.39
8	subjectAltName	o	o	-	-	See Table A3.46 (1), Note 22
9	issuerAltName	o	o	-	-	See Table A3.46 (1), Note 2
10	subjectDirectoryAttributes	o	o	-	-	
11	basicConstraints	o	o	M	-	See Table A3.40
12	nameConstraints	o	o	-	–	See Table A3.41
13	policyConstraints	o	o	-	–	See Table A3.43
14	cRLDistributionPoints	o	o	-	-	See Table A3.44, Note 21
15	authorityInfoAcess	o	o	-	-	See Table A3.45
16	inhibitAnyPolicy	o	o	-	–	
17	subjectInfoAccess	o	o	-	-	
18	freshestCRL	o	o	-	-	See Table A3.44

Table A3.35: Authority Key Identifier

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	keyIdentifier	c14	c14	Mr	mr	Note 23
2	certIssuer	c15	c15	-	-	
3	certSerialNumber	c15	c15	-	-	

Table A3.36: Key Usage

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	digitalSignature	o	o	M	m	Note 24
2	nonRepudiation	o	o	M	m	Note 24
3	keyEncipherment	o	o	M	m	
4	dataEncipherment	o	o	M	m	
5	keyAgreement	o	o	–	–	
6	keyCertSign	o	o	-	–	
7	cRLSign	o	o	-	-	Note 24
8	encipherOnly	o	o	–	–	
9	decipherOnly	o	o	–	–	

Table A3.37: Private Key Usage Period

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	notBefore	m	c16	-	-	
2	notAfter	m	c16	-	-	

Table A3.38: Certificate Policies

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	policyIdentifier	m	m	-	-	
2	policyQualifiers	o	o	-	-	
3	policyQualifierId	m	m	-	-	
4	qualifier	o	o	-	-	

Table A3.39: Policy Mappings

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	issuerDomainPolicy	m	m	-	–	
2	subjectDomainPolicy	m	m	-	–	

Table A3.40: Basic Constraints

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	cA	m	m	M	mr	d(false)
2	pathLenConstraint	m	o	M	-	

Table A3.41: Name Constraints

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	permittedSubtrees	m	o	-	–	See Table A3.42
2	excludedSubtrees	m	o	-	–	See Table A3.42

Table A3.42: General Subtrees

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	base	m	m	-	–	See Table A3.46 (5)
2	minimum	m	m	-	–	d(0)
3	maximum	m	o	-	–	

Table A3.43: Policy Constraints

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	requireExplicitPolicy	m	o	-	–	
2	inhibitPolicyMapping	m	o	-	–	

Table A3.44: CRL Distribution Points

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	distributionPoint	o	o	-	m	See Table A3.55 (17)
2	reasons	o	o	-	-	See Table A3.55 (20)
3	cRLIssuer	o	o	-	-	See Table A3.55 (4)

Table A3.45: Authority Information Access

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	accessMethod	o	o	-	-	Note 25
2	accessLocation	o	o	-	-	Note 25

Table A3.46: Common Fields

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	Time					
2	UTCTime	m	m	m	c17	
3	GeneralizedTime	o	o	m	c18	
4	GeneralNames					
5	GeneralName	m	m	m	m	
6	otherName	o	o	-	-	
7	rfc822Name	o	o	m	c19	Note 26
8	dNSName	o	o	-	-	Note 26
9	x400Address	o	o	-	-	
10	directoryName	o	o	m	m	
11	ediPartyName	o	o	-	-	
12	nameAssigner	o	o	-	-	
13	partyName	o	o	-	m	
14	uniformResourceIdentifier	o	o	m	m	Note 26
15	iPAddress	o	o	-	-	Note 26
16	registeredID	o	o	-	-	
17	DistributionPointName					
18	fullName	m	m	m	m	See (4)
19	nameRelativeToCRLIssuer	m	m	m	m	
20	ReasonFlags					
21	unused	o	o	-	-	
22	keyCompromise	o	o	m	m	
23	cACompromise	o	o	m	m	
24	affiliationChange	o	o	m	-	
25	superseded	o	o	m	-	
26	cessationOfOperation	o	o	m	-	
27	certificateHold	o	o	m	-	

A3.2 CERTIFICATE REVOCATION LISTS

A3.2.1 CRL Introduction

- This appendix provides the Profile for the CRL for use in this environment. The structure for the CRL is defined in the 1997 version of ITU-T X.509 | ISO/IEC 9594-8.

A3.2.2 Description of Tables

- The “Item” and “Notes” columns are provided for cross-referencing. The numbers in the “Item” column are the row numbers. The numbers in the “Notes” column indicate the table numbers

followed by the "item" number enclosed in parentheses. These two columns are used together to point to sub-elements. The "Notes" column also refers to additional information supplied in the last row of the table.

- The "Protocol Elements" column refers to the name of ASN.1 fields taken from the X.500 recommendations.

- In each table, the "Base" column reflects the level of support required for conformance to the base standard. The level of support refers to the support classification for the "Base" column. The "Base" column is broken into "Proc." (i.e., processing) and "Gen." (i.e., generation) columns. The "Proc." column reflects the level of support required by compliant certificate processing and using entities who process CRLs. The "Gen." column reflects the level of support required in compliant CRLs (i.e., the information that is included in the CRL). When the CA acts as an End Entity (e.g., when a CA receives a message), then the "Proc." column applies.

- The "Support" column is provided for completion by the supplier of the implementation as follows:

Y	the protocol element is fully supported (i.e., supporting the requirements of the m support classification)
N	the protocol element is not fully supported, further qualified to indicate the action taken on receipt of such an element as follows:
	ND - the element is discarded/ignored
	NR - the PDU is rejected
– or blank	the protocol element is not applicable

A3.2.3 Support Classifications

- Each of the protocol elements listed in Section A1.2, A1.3, A1.4 and A1.5 are designated as having a support requirement of mandatory or optional. Where protocol elements are nested (i.e., the elements contain sub-elements), the requirement to support the nested element is relevant only when the immediately containing (parent) element is supported.
- To specify the support level of the protocol elements, the following terminology is defined.

A3.2.4 Static Capability

- The following classifications are used to specify static conformance (i.e., capability).

- **mandatory support (m)** : Implementations claiming to create certificates shall be able to generate the protocol element. Implementations claiming to process certificates shall be able to receive the protocol elements and perform all associated procedures (i.e., implying the ability to handle both the syntax and the semantics of the element) as relevant.

- **optional (o)** : Implementations claiming to create certificates are not required to support generation of the protocol element. If support is claimed, the element shall be treated as if it were specified as mandatory support, and the sub-elements, if present, shall be supported as specified. Implementations claiming to perform processing of certificates shall ignore the protocol element and continue processing of the certificate.

- **conditional (c)** : Implementations shall support the protocol element under the conditions specified. If the conditions are met, the protocol element shall be treated as if it were specified as mandatory support. If these conditions are not met, the protocol element shall be treated as if it were specified as optional support (unless otherwise stated).

- **not applicable** (–) : This element is not applicable in the particular context in which this classification is used.

A3.2.5 Dynamic Capability

The following classifications are used to specify dynamic conformance (i.e., behaviour).

- **required (r)** : The information for this protocol element must be populated upon certificate generation.

Table A3.47: CRL

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	signed	m	m	m	mr	
2	toBeSigned	m	m	m	mr	
3	version	o	o	m	mr	
4	signature	m	m	m	mr	See Table A3.48, Note 2
5	issuer	m	m	m	mr	See ACP 133
6	thisUpdate	m	m	m	mr	See Table A3.55 (1)
7	nextUpdate	o	o	m	mr	See Table A3.55 (1)
8	revokedCertificates	o	o	m	m	
9	userCertificates	m	m	m	mr	
10	revocationDate	m	m	m	mr	See Table A3.55 (1)
11	crlEntryExtensions	o	o	-	-	See Table A3.53
12	crlExtensions	o	o	m	mr	See Table A3.49
13	algorithmIdentifier	m	m	m	mr	See Table A3.48, Note 2
14	encrypted	m	m	m	mr	

Table A3.48: Algorithm Identifier

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	algorithm	m	m	m	mr	
2	parameters	m	m	m	m	

Table A3.49: Extensions

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	extnID	m	m	m	mr	Note 3
2	critical	m	m	m	mr	d(false)
3	extnValue	m	m	m	mr	

Table A3.50: CRL Extensions

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	authorityKeyIdentifier	o	o	m	mr	See Table A3.51
2	issuerAltName	o	o	-	-	See Table A3.55 (4)
3	cRLNumber	o	o	-	-	
4	issuingDistributionPoint	o	o	-	-	See Table A3.52, Note 4
5	deltaCRLIndicator	o	o	-	-	

Table A3.51: Authority Key Identifier

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	keyIdentifier	c3	c4	m	mr	
2	certIssuer	c4	c4	-	-	
3	certSerialNumber	c4	c4	-	-	

Table A3.52: Issuing Distribution Point

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	distributionPoint	o	o	-	-	See Table A3.55 (17)
2	onlyContainsUserCerts	o	o	-	-	d(false)
3	onlyContainsCACerts	o	o	-	-	d(false)
4	onlySomeReasons	o	o	-	-	See Table A3.55 (20)
5	indirectCRL	o	o	-	-	d(false)

Table A3.53: CRL Entry Extensions

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	reasonCode	o	o	-	-	See Table A3.54
2	instructionCode	o	o	-	-	
3	invalidityDate	o	o	-	-	
4	certificateIssuer	o	o	-	-	See Table A3.55 (4), Note 5

Table A3.54: Reason Code

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	unspecified	o	o	-	-	
2	keyCompromise	o	o	-	-	
3	cACompromise	o	o	-	-	
4	affiliationChanged	o	o	-	-	
5	superseded	o	o	-	-	
6	cessationOfOperation	o	o	-	-	
7	certificateHold	o	o	-	-	
8	removeFromCRL	o	o	-	-	

Table A3.55: Common Fields

Item	Protocol Element	Base		Profile		Notes
		Proc.	Gen.	Proc.	Gen.	
1	Time					
2	UTCTime	m	m	M	c5	
3	GeneralizedTime	o	o	M	c6	
4	GeneralNames					
5	GeneralName	m	m	M	m	
6	otherName	o	o	-	-	
7	rfc822Name	o	o	M	c7	Note 6
8	dNSName	o	o	-	-	Note 6
9	x400Address	o	o	-	-	
10	directoryName	o	o	M	m	
11	ediPartyName	o	o	-	-	
12	nameAssigner	o	o	-	-	
13	partyName	o	o	-	m	
14	uniformResourceIdentifier	o	o	M	m	Note 6
15	iPAddress	o	o	-	-	Note 6
16	registeredID	o	o	-	-	
17	DistributionPointName					
18	fullName	m	m	M	m	See (4)
19	nameRelativeToCRLIssuer	m	m	M	m	
20	ReasonFlags					
21	unused	o	o	-	-	
22	keyCompromise	o	o	M	m	
23	caCompromise	o	o	M	m	
24	affiliationChange	o	o	M	-	
25	superseded	o	o	M	-	
26	cessationOfOperation	o	o	M	-	
27	certificateHold	o	o	M	-	

Appendix 4: XML Security Label Syntax

This Annex is a draft specification from the NATO C3 Agency (NC3A) called "Alternative XML-Security Label Syntax and Processing (Draft Version 2005) ". This is not an official document from the NC3A and has not been provided for the purpose of this demonstrator. It has been used by the RTO/IST 061 because parts of the specification fits our needs for XML Security Labeling.

A4.1 INTRODUCTION

This document specifies the XML syntax and processing rules for creating and representing digital information security labels. More specifically, this specification defines an XML security label element type. Conformance requirements are specified by way of schema definitions and prose respectively. XML security labels represent the security classification or "sensitivity" of digital data. An XML security label can be applied to any *digital data (data object)*, including XML. Similar to XML Signatures [XML-Sig], *enveloped* or *enveloping* security labels apply to data within the same XML document as the XML security label; *detached* security labels may apply to XML data without modifying the structure of the data and to data which is external to the XML document containing the XML security label.
This document is based on the *XML-Security Label Syntax and Processing* by Andreas Thummel, NC3A and incorporates some lessons learned from a prototype implementation of that specification. The main lesson learned is that it could be beneficial to decouple the XML security label and the XML signature. Detached labels are still enabled by referencing both the security label and the labeled object with references inside an XML digital signature. This decoupling has the benefit of enabling standard XML digital signature validation to include the validation of the XML security labels. The main disadvantage of this construct is that it obscures the link between label and labeled object by hiding it in the references of a digital signature and that it in some complex scenarios of multiple labels applied to multiple objects requires multiple digital signatures.

A4.1.1 Versions, Namespaces and Identifiers

No provision is made for an explicit version number in this syntax. If a future version is needed, a different namespace will be generated. The XML namespace [XML-ns] URI that MUST be used by implementations of this specification is:

```
xmlns:slab="http://nc3a.nato.int/2004/06/xmlslab#"
```

This namespace is also used as the prefix for algorithm identifiers used by this specification. While applications MUST support XML and XML namespaces, the use of internal entities [XML] or the "slab" XML namespace prefix and defaulting/scoping conventions are OPTIONAL; these facilities are used to provide compact and readable examples.

This document uses the namespace prefix "dsig" to refer to the XML Signatures namespace:

```
xmlns:dsig="http://www.w3.org/2000/09/xmldsig#"
```

A4.2 SECURITY LABEL OVERVIEW AND EXAMPLES

This section provides an overview and examples of how XML security labels are applied within this specification.
In this section, an informal representation and examples are used to describe the structure of the XML security label syntax. This representation and examples may omit attributes, details and potential features that are fully explained later.
XML security labels are represented by the SecurityLabel element which has the following structure (where "?" denotes zero or one occurrence; "+" denotes one or more occurrences; "*" denotes zero or more occurrences; and "|" denotes choice):

```
<SecurityLabel>
  (<LabeledObjectGroup ID?>
    <ConfidentialityLabel>
       <SecurityPolicyIdentifier/>
       <SecurityClassification/>
       (<Privacymark/>)?
       (<SecurityCategory/>)*
    </ConfidentialityLabel>
   (<Object/>)*
  </LabeledObjectGroup>)+
</SecurityLabel>
```

XML security labels can be applied to *digital content (data objects)* in any data format. If a data object is XML the security label can be applied to a fraction (more specifically a subset of the XPath [XPath] node set) of the data object. Otherwise, the security label is applied to the entire data object which is then treated as a single binary/octet data stream. Since a Securitylabel element (and its Id attribute values/names) may co-exist or be combined with other elements (and their IDs) within a single XML document, care should be taken in choosing names such that there are no subsequent collisions that violate the ID uniqueness validity constraint [XML]. XML security labels SHOULD be applied to data objects as a parent or as a child of the data object it applies to. A label applies to all data objects that are children of the label. If the label has no children, it applies to its parent object (but not to the parent's children). A label that applies to its children is termed an enveloping label. A label that applies to its parent is termed an enveloped label. Labels MAY also be applied to objects through references in digital signatures. This is termed detached labels and is particularly useful when labeling data that is not represented in XML or when labeling XML data without changing the structure of the data.

To protect the integrity of security labels and the labeled data objects, applications are strongly encouraged to apply an XML digital signature that covers both the labels and the data objects. For an XML digital signature to cover both the labels and the associated data objects, the computation of the signed digest value must include the digests of both the labels and the data objects. If the security label is used as basis for security policy enforcing such as in an access control mechanism, the label and the data object it applies to MUST be signed with a digital signature. The security policy enforcing application MUST validate the digital signature before making policy decisions based on the label.

Digital signatures may be applied to labels and the labeled objects as enveloping, enveloped or detached signatures. Detached signatures MAY be used to apply labels to data objects by including a reference to that object and to the label in the references of the digital signature. Processing applications MUST include logic to process such linking of labels and objects through digital signature references. This construct of linking labels and objects through digital signature references is designed to allow verification of digital signatures on labels to follow the standard XML Digital Signature specification. The construct also allows the usage of XPath transformations to referenced objects to select parts of the reference to apply the label to.

This specification allows any object to be labeled with any label and does not include mechanisms to prevent labeling conflicts that might arise from poor labeling of objects. Applications are encouraged to avoid labeling conflicts and security enforcing applications that relies on the labels of objects MUST resolve labeling conflicts before making policy decisions. Label conflict resolution strategies may be different depending on the security policy of the application. Possible conflict resolution strategies are aborting with an error or applying the highest classification of the conflicting labels.

A4.2.1 Detached Example

The following example is a detached security label of the content of the NC3A Internet home page.

```
[r01] <SecurityLabel xmlns:dsig=http://www.w3.org/2000/09/xmldsig#>
[r02]   <LabeledObjectGroup Id="myLabeledObject">
[r03]     <ConfidentialityLabel>
[r04]       <SecurityPolicyIdentifier>
```

```
[r05]         NATO
[r06]       </SecurityPolicyIdentifier>
[r07]       <SecurityClassification>
[r08]         UNCLASSIFIED
[r09]       </SecurityClassification>
[r10]       <SecurityCategory type="Permissive">
[r11]         RELEASABLE FOR INTERNET TRANSMISSION
[r12]       </SecurityCategory>
[r13]     </ConfidentialityLabel>
[r14]   </LabeledObjectGroup>
[r15] </SecurityLabel>

[r16] <dsig:Signature>
[r17]   <dsig:SignedInfo>
[r18]     <dsig:Reference URI="#myLabeledObject">
[r19]       <Transforms>
[r20]         <Transform Algorithm=
[r21]          "http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
[r22]       </Transforms>
[r23]       <DigestMethod
[r24]        Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
[r25]       <DigestValue>
[r26]         3d726c554c90427dd535a0aa2dead8c7e77f8c2a
[r27]       </DigestValue>
[r28]     </dsig:Reference>
[r29]     <dsig:Reference URI="http://www.nc3a.nato.int/index.html">
[r30]       <DigestMethod
[r31]        Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
[r32]       <DigestValue>
[r33]         3d726c554c90427dd535a0aa2dead8c7e77f8c2a
[r34]       </DigestValue>
[r35]     </dsig:Reference>
[r36]   </dsig:SignedInfo>
[r37]   <dsig:SignatureValue>DUEm0...</dsig:SignatureValue>
[r38] </dsig:Signature>
```

`[r02-15]` The required `LabeledObjectGroup` element defines a `ConfidentialityLabel` element. The `LabeledObjectGroup` element is the data that is signed in the Signature element along with the referenced web page.

`[r03-13]` The required `ConfidentialityLabel` element contains the actual information about the security classification marking (e.g., NATO UNCLASSIFIED) of the data objects.

`[r04-06]` The required `SecurityPolicyIdentifier` specifies the security policy, e.g. NATO.

`[s07-09]` The required `SecurityClassification` element specifies the sensitivity of the data object, e.g. UNCLASSIFIED.

`[r10-12]` The optional `SecurityCategory` element can be `PERMISSIVE`, `RESTRICTIVE` or `INFORMATIVE` which is expressed by its required Type attribute.

`[r16-38]` The optional `dsig:Signature` signs the `LabeledObjectGroup` element and thus binds the information contained in `ConfidentialityLabel` to the data objects contained referenced by the `dsig:Reference` elements; it conforms to the XML Signature standard [XML-Sig].

A4.2.2 Enveloping Example

The following example shows an enveloping security label for a base64 encoded binary image in JPEG format:

```
[i01] <SecurityLabel>
[i02]   <LabeledObjectGroup Id=”mySecondLabeledObject“>
```

```
[i03]     <ConfidentialityLabel>
[i04]       <SecurityPolicyIdentifier>
[i05]         NATO
[i06]       </SecurityPolicyIdentifier>
[i07]           <SecurityClassification>
[i08]         UNCLASSIFIED
[i09]       </SecurityClassification>
[i10]       <SecurityCategory Type="PERMISSIVE">
[i11]         RELEASABLE FOR INTERNET TRANSMISSION
[i12]       </SecurityCategory>
[i13]     </ConfidentialityLabel>
[i14]     <Object MimeType="image/jpg" Encoding="base64">
[i15]      9j/4AAQSkZJRgABAgAAZABkAAD/7AARRHVja3kAAQAEAAAAPAAA/+4AJkFk
[i16]      b2JlAGTAAAAAAQMAFQQDBgoNAAAFBgAACJcAAA0rAAATN//bAIQABgQEBAUE
[   ]      ...
[i17]     </Object>
[i18]   </LabeledObjectGroup>
      </SecurityLabel>
```

[i14-17] The `IncludedObject` element contains the data object as content. The data object can be an octet stream as in this case or itself be a node-set.

A4.2.3 Enveloped Example

This example shows an enveloped security label:

```
[e01] <target>
[e02] <coordinates>
[e03]    <x>12345</x>
[e04]    <y>67890</y>
[e05] </coordinates>
[e06]   <SecurityLabel
[e06a]      xmlns:slab="http://nc3a.nato.int/2004/06/xmlslab#">
[e07]     <LabeledObjectGroup Id="myLabeledObject">
[e08]       <ConfidentialityLabel>
[e09]         <SecurityPolicyIdentifier>
[e10]           NATO
[e11]         </SecurityPolicyIdentifier>
[e12]         <SecurityClassification>
[e13]           UNCLASSIFIED
[e14]         </SecurityClassification>
[e15]         <SecurityCategory type="Permissive">
[e16]           RELEASABLE FOR INTERNET TRANSMISSION
[e17]         </SecurityCategory>
[e18]       </ConfidentialityLabel>
[e27]     </LabeledObjectGroup>
[e28]   </SecurityLabel>
[e29] </target>
```

[e06-28] The `SecurityLabel` element appears as child of another element.

A4.2.4 Core Security Label Syntax

Please note that the core security label syntax has not been updated to the alternate version!

The general structure of an XML security label is described in Security Label Overview and Examples (Section A4.2). This section provides detailed syntax of the core security label. Features described in this section are mandatory to implement unless otherwise indicated. The syntax is defined via DTDs and [XML-Schema] with the following XML preamble, declaration, and internal entity.

```
Schema Definition:

<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE schema
  PUBLIC "-//W3C//DTD XMLSchema 200102//EN"
  "http://www.w3.org/2001/XMLSchema.dtd"
 [
   <!ATTLIST schema
     xmlns:sl CDATA #FIXED http://www.nc3a.nato.int/2004/06/xmlslab#>
   <!ENTITY slab 'http://www.nc3a.nato.int/2004/06/xmlslab#'>
   <!ENTITY % p ''>
   <!ENTITY % s ''>
  ]>
  <schema xmlns=http://www.w3.org/2001/XMLSchema
          xmlns:slab=http://www.nc3a.nato.int/2004/06/xmlslab#
          xmlns:dsig="http://www.w3.org/2000/09/xmldsig#"
          targetNamespace="http://www.nc3a.nato.int/2004/06/xmlslab#"
          elementFormDefault="qualified" version="0.1">

DTD:

<!ENTITY % Category.ANY ''>
<!ENTITY % Object.ANY ''>
<!ENTITY % Method.ANY ''>
<!ENTITY % Transform.ANY ''>
```

A4.2.4.1 The `SecurityLabel` Element

The `SecurityLabel` element is the root element of an XML Security Label. It contains one or more `LabeledObjectGroup` elements and zero or one `dsig:Signature` element. An Iimplementation MUST generate laxly schema valid [XML-schema] `SecurityLabel` elements as specified by the following schema:

```
Schema Definition:

<element name="SecurityLabel" type="slab:SecurityLabelType"/>
  <complexType name="SecurityLabelType">
  <sequence>
    <sequence maxOccurs="unbounded">
      <element ref="slab:LabeledObjectGroup"/>
    </sequence>
    <element ref="dsig:Signature" minOccurs="0"/>
  </sequence>
  <attribute name="Id" type="ID" use="optional"/>
</complexType>

DTD:

<!ELEMENT SecurityLabel (LabeledObjectGroup, Signature?)>
<!ATTLIST SecurityLabel
 xmlns   CDATA   #FIXED 'http://www.nc3a.nato.int/2004/06/xmlslab'
 Id      ID #IMPLIED
>
```

A4.2.4.2 The `LabeledObjectGroup` Element

The `LabeledObjectGroup` element must contain one `Confidentialitylabel` element. It binds the information on security classification specified in the `Confidentialitylabel` element to the content of its `dsig:Object` child elements. If the label has no children, the label applies to its parent, but it does not apply to its parents children. The `LabeledObjectGroup` element may contain an optional ID attribute that will allow it to be referenced, e.g. by the signature element.

```
Schema Definition:

<element name="LabeledObjectGroup"
    type="slab:LabeledObjectGroupType"/>
<complexType name="LabeledObjectGroupType">
  <sequence>
    <element ref="slab:ConfidentialityLabel"/>
    <choice maxOccurs="unbounded">
      <element ref="dsig:Reference"/>
      <element ref="dsig:Object"/>
    </choice>
  </sequence>
  <attribute name="Id" type="ID" use="optional"/>
</complexType>

DTD:

<!ELEMENT LabeledObjectGroup (ConfidentialityLabel,
 (dsig:Reference | dsig:Object)+)>
<!ATTLIST LabeledObjectGroup
  Id ID #IMPLIED
>
```

A4.2.4.3 The `ConfidentialityLabel` Element

`ConfidentialityLabel` is a required element that specifies the security classification of the data objects included or referenced in its `LabeledObjectGroup` parent; it may contain an optional ID attribute.
The structure of the `ConfidentialityLabel` element is modeled after the X.841 [X.841] confidentiality label.

```
Schema Definition:

<element name="ConfidentialityLabel"
  type="slab:ConfidentialityLabelType"/>
<complexType name="ConfidentialityLabelType">
  <sequence>
    <element ref="slab:SecurityPolicyIdentifier"/>
    <element ref="slab:SecurityClassification"/>
    <element ref="slab:PrivacyMark" minOccurs="0"/>
    <element ref="slab:SecurityCategory" minOccurs="0"
      maxOccurs="unbounded"/>
  </sequence>
</complexType>

DTD:

<!ELEMENT ConfidentialityLabel (SecurityPolicyIdentifier,
  SecurityClassification, PrivacyMark?, SecurityCategory*)>
```

```
<!ATTLIST ConfidentialityLabel
  Id ID #IMPLIED
>
```

A4.2.4.4 The `SecurityPolicyIdentifier` Element

`SecurityPolicyIdentifier` is a required element that specifies the security policy, e.g. NATO.

```
Schema Definition:

<element name="SecurityPolicyIdentifier" type="token"/>

DTD:

<!ELEMENT SecurityPolicyIdentifier (#PCDATA)>
```

A4.2.4.5 The `SecurityClassification` Element

`SecurityClassification` is a required element that specifies the sensitivity of the data objects.

```
Schema Definition:

<element name="SecurityClassification"
  type="slab:SecurityClassificationType"/>
<simpleType name="SecurityClassificationType">
<restriction base="token">
  <enumeration value="UNMARKED"/>
  <enumeration value="UNCLASSIFIED"/>
  <enumeration value="RESTRICTED"/>
  <enumeration value="CONFIDENTIAL"/>
  <enumeration value="SECRET"/>
  <enumeration value="TOP SECRET"/>
</restriction>

DTD:

<!ELEMENT SecurityClassification (#PCDATA)>
```

A4.2.4.6 The `PrivacyMark` Element

`PrivacyMark` is an optional element. The INFOSEC Technical and Implementation Guidance for Electronic Labeling of NATO Information [NATO-Labeling] states:

> "There is no plan for NATO to use the privacy-mark field. However NATO systems may encounter the PrintableString 'CLEAR' to represent the ACP 127 Clear Service in this field."

```
Schema Definition:

<element name="PrivacyMark" type="string"/>

DTD:

<!ELEMENT PrivacyMark (#PCDATA)>
```

A4.2.4.7 The `SecurityCategory` Element

`SecurityCategory` is a optional element that according to the INFOSEC Technical and Implementation Guidance for Consistent Marking of NATO Information in C3 Systems [NATO-Marking] serves as an indicator of:

1. an additional specific sensitivity;
2. a dissemination control; or
3. an informational marking on which no automated access control is performed.

```
Schema Definition:

<element name="SecurityCategory" type="slab:SecurityCategoryType"/>
<complexType name="SecurityCategoryType" mixed="true">
  <attribute name="Type">
    <simpleType>
      <restriction base="token">
        <enumeration value="RESTRICTIVE"/>
        <enumeration value="PERMISSIVE"/>
        <enumeration value="INFORMATIVE"/>
      </restriction>
    </simpleType>
  </attribute>
</complexType>

DTD:

<!ELEMENT SecurityCategory (#PCDATA %Category.ANY;)>
<!ATTLIST SecurityCategory
  Type (RESTRICTIVE | PERMISSIVE | INFORMATIVE) #IMPLIED
>
```

A4.2.4.8 The `dsig:Object` Element

`dsig:Object` is an optional element that may occur one or more times. When present, this element may contain any data. The `dsig:Object` is of type `dsig:ObjectType`. Further information can be found in section 4.5 of the XML Signature specification [XML-Sig].

A4.3 ALGORITHMS

Digest and encoding algorithms to be supported by security label applications are the same as for XML signatures [XML-Sig]; the statements made in that reference about RECOMMENDED, OPTIONAL, and REQUIRED features also apply for this specification. Transform algorithms are also the same with the exception of the Enveloped Signature Transform. Instead, the Enveloped Security Label Transform MUST be supported.

This section specifies the required Enveloped Security Label Transform algorithm; it removes the `SecurityLabel` element from the calculation of the digest when the security label is within the content that it is being labeled. This MAY be implemented via the RECOMMENDED XPath specification specified in section 5.2; it MUST have the same effect as that specified by the XPath Transform.

An additional XPath Transform Sign Security Label is specified in section 5.3; it may be used by the signature element of a security label to reference the corresponding `LabeledObjectGroup` element.

A4.3.1 XPath Filtering

The XPath Filtering Transform processing instructions are the same as for XML signatures; further information can be found in section 6.6.3 of the XML Signature specification [XML-Sig].

A4.4 REFERENCES

HTTP

RFC 2616. *Hypertext Transfer Protocol -- HTTP/1.0*. J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee. June 1999. http://www.ietf.org/rfc/rfc2616.txt.

MIME

RFC 2045. *Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies*. N. Freed & N. Borenstein. November 1996. http://www.ietf.org/rfc/rfc2045.txt.

NATO-Labeling

INFOSEC Technical and Implementation Guidance for Electronic Labeling of NATO Information, NATO C3 Board.

NATO-Marking

INFOSEC Technical and Implementation Guidance for Consistent Marking of NATO Information in C3 Systems, NATO C3 Board.

Unicode

The Unicode Consortium. *The Unicode Standard.* http://www.unicode.org/unicode/standard/standard.html.

UTF-16

RFC 2781. *UTF-16, an encoding of ISO 10646.* P. Hoffman , F. Yergeau. February 2000. http://www.ietf.org/rfc/rfc2781.txt.

UTF-8

RFC 2279. *UTF-8, a transformation format of ISO 10646*. F. Yergeau. January 1998. http://www.ietf.org/rfc/rfc2279.txt.

URI

RFC 2396. *Uniform Resource Identifiers (URI): Generic Syntax.* T. Berners-Lee, R. Fielding, L. Masinter. August 1998. http://www.ietf.org/rfc/rfc2396.txt.

URL

RFC 1738. *Uniform Resource Locators (URL)*. T. Berners-Lee, L. Masinter, and M. McCahill. December 1994. http://www.ietf.org/rfc/rfc1738.txt.

URN

RFC 2141. *URN Syntax.* R. Moats. May 1997. http://www.ietf.org/rfc/rfc2141.txt.
RFC 2611. *URN Namespace Definition Mechanisms.* L. Daigle, D. van Gulik, R. Iannella, P. Falstrom. June 1999. http://www.ietf.org/rfc/rfc2611.txt.

XML

Extensible Markup Language (XML) 1.0 (Second Edition). W3C Recommendation. T. Bray, E. Maler, J. Paoli, C. M. Sperberg-McQueen. October 2000. http://www.w3.org/TR/2000/REC-xml-20001006.

XML-C14N

Canonical XML. W3C Recommendation. J. Boyer. March 2001. http://www.w3.org/TR/2001/REC-xml-c14n-20010315. http://www.ietf.org/rfc/rfc3076.txt.

XML-ns

Namespaces in XML. W3C Recommendation. T. Bray, D. Hollander, A. Layman. January 1999. http://www.w3.org/TR/1999/REC-xml-names-19990114.

XML-schema

XML Schema Part 1: Structures. W3C Recommendation. D. Beech, M. Maloney, N. Mendelsohn, H. Thompson. May 2001. http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/.
XML Schema Part 2: Datatypes W3C Recommendation. P. Biron, A. Malhotra. May 2001. http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/.

XML-Sig

RFC 3275. XML Signature Syntax and Processing. http://www.w3.org/TR/xmldsig-core/ http://www.ietf.org/rfc/rfc3275.txt.

XPath

XML Path Language (XPath) Version 1.0. W3C Recommendation. J. Clark, S. DeRose. October 1999. http://www.w3.org/TR/1999/REC-xpath-19991116.

XPointer

XML Pointer Language (XPointer). W3C Candidate Recommendation. S. DeRose, R. Daniel, E. Maler. January 2001. http://www.w3.org/TR/2001/CR-xptr-20010911/.

XSL

Extensible Stylesheet Language (XSL). W3C Recommendation. S. Adler, A. Berglund, J. Caruso, S. Deach, T. Graham, P. Grosso, E. Gutentag, A. Milowski, S. Parnell, J. Richman, S. Zilles. October 2001. http://www.w3.org/TR/2001/REC-xsl-20011015/.

XSLT

XSL Transforms (XSLT) Version 1.0. W3C Recommendation. J. Clark. November 1999. http://www.w3.org/TR/1999/REC-xslt-19991116.html.

X.841

Information technology – Security techniques –
Security information objects for access control, ITU-T X841 | ISO/IEC 15816, October 2000.

Appendix 5: XML Security Label Guidance and Matching Rules

A5.1 XML LABEL GUIDANCE

The syntax for the XML Security Label selected for the demonstrator is defined in Section A4.2.4 of Appendix 4. For the CWID06 demonstrator, the following additional restrictions SHALL apply:

A5.1.1 LabeledObjectGroup

A SecurityLabel element SHALL only include a single LabeledObjectGroup. The ID if the LabeledObjectGroup SHALL be one of the following:

- informationSecurityLabel
- privilegeSecurityLabel

The LabeledObjectGroup SHALL only include a single ConfidentialityLabel.

A5.1.2 SecurityPolicyIdentifier

The SecurityPolicyIdentifier SHALL be RTO-IST-061-Mission.

A5.1.3 SecurityClassification

The classification TOP SECRET SHALL not be used.

A5.1.4 PrivacyMark

PrivacyMark element SHALL not be present.

A5.1.5 SecurityCategory

To simply prosessing, a SecurityCategory PCDATA SHALL consist of the following components:

<CATEGORY SET> <CATEGORIES>

Each category in <CATEGORIES> SHALL be separated by space.

The following sets SHALL be supported:

Category Set	Category type	Allowed elements (categories)
EYES ONLY	Permissive	• Any 2 letter country code
DEPARTMENT	Restrictive	• INTELLIGENCE • LOGISTICS • CRYPTO
RELEASABLE TO	Permissive	• RTO-IST-061 • Any 2 letter country code for countries not in the group
RELEASABLE TO	Informative	• INTERNET

Additional sets may be added on demand later.

Examples:

A Norwegian privilegeSecurityLabel will usually include these categories:

```
<SecurityLabel>
   <LabeledObjectGroup Id="privilegeSecurityLabel">
     <ConfidentialityLabel>
       <SecurityPolicyIdentifier>
         RTO-IST-061-Mission
       </SecurityPolicyIdentifier>
       <SecurityClassification>
         SECRET
       </SecurityClassification>
       <SecurityCategory type="Permissive">
         NO EYES ONLY
       </SecurityCategory>
       <SecurityCategory type="Permissive">
         RELEASABLE TO RTO-IST-061
       </SecurityCategory>
     </ConfidentialityLabel>
   </LabeledObjectGroup>
 </SecurityLabel>
```

Some informationSecurityLabel categories that may be defined by this user:

```
<SecurityCategory type="Permissive">
   NO EYES ONLY
</SecurityCategory>
```

The labeled information shall not be distributed to other countries.

```
<SecurityCategory type="Permissive">
   NO PO EYES ONLY
</SecurityCategory>
```

The labeled information is restricted to a subset of the RTO-IST-061 group

```
<SecurityCategory type="Permissive">
   RELEASABLE TO RTO-IST-061 UK
</SecurityCategory>
```

The labeled information may be distributed to a country not in the group.

```
<SecurityCategory type=" Informative">
   RELEASABLE TO INTERNET
</SecurityCategory>
```

In addition, the DEPARTMENT set may be used to further restrict access, i.e. to a subset of users within the RTO-IST group or countries.

A5.2 SECURITY LABEL MATCHING RULES

When comparing a Security Label vs. another as a part of an access control check (e.g. privilege vs. information) the following rule SHALL apply:

A Security label A shall only have access to a Security Label B if and only if all of the following is true:

SecurityPolicyIdentifier

1. The SecurityPolicyIdentifiers are be identical. (I.g. both NATO)

SecurityClassification

2. The SecurityClassification of label A is equvalent or higher than label B

SecurityCategories, restrictive type

3. If label B has a restrictive category set, label A has an identical set and **ALL** categories in B are present in A.

SecurityCategories, permissive type

4. If label B has a permissive category set, label A has an identical set and **AT LEAST ONE** of the categories in B is present in A.

Note: Informative category sets shall be ignored.

Appendix 6: MTI Tracks Model

A6.1 MTI TRACKS XML SCHEMA

Here is the Xml Schema file representing the MTI Tracks data model:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSPY v2004 rel. 3 U (http://www.xmlspy.com) by Thales
(THALES COMMUNICATION) -->
<!-- edited with XMLSpy v2006 sp2 U (http://www.altova.com) by THALES
COMMUNICATIONS (THALES COMMUNICATIONS) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified" version="1">
      <!--
      : XML schema definition for No UAV MTI exchange between ISTARs in Cwid
2006

      : Property of Thales Communications France

      : February, 20th 2006

      -->
      <!-- ************************** -->
      <!--          Type definition      -->
      <!-- ************************** -->
      <!-- ************************** -->
      <!--          Type Simple          -->
      <!-- ************************** -->
      <!--  Type Date & Heure  -->
      <xs:simpleType name="type_Date_Time">
            <xs:restriction base="xs:dateTime">
                  <xs:minInclusive value="1901-01-01T00:00:00"/>
                  <xs:maxInclusive value="2036-12-31T23:59:59"/>
                  <xs:pattern value=".{19}"/>
            </xs:restriction>
      </xs:simpleType>
      <!--  Type TIME_ZONE  -->
      <xs:simpleType name="TIMEZONE_TYPE">
            <xs:restriction base="xs:string">
                  <xs:enumeration value="A"/>
                  <xs:enumeration value="B"/>
                  <xs:enumeration value="C"/>
                  <xs:enumeration value="D"/>
                  <xs:enumeration value="E"/>
                  <xs:enumeration value="F"/>
                  <xs:enumeration value="G"/>
                  <xs:enumeration value="H"/>
                  <xs:enumeration value="I"/>
                  <xs:enumeration value="J"/>
                  <xs:enumeration value="K"/>
                  <xs:enumeration value="L"/>
                  <xs:enumeration value="M"/>
                  <xs:enumeration value="N"/>
                  <xs:enumeration value="O"/>
                  <xs:enumeration value="P"/>
                  <xs:enumeration value="Q"/>
```

```
                <xs:enumeration value="R"/>
                <xs:enumeration value="S"/>
                <xs:enumeration value="T"/>
                <xs:enumeration value="U"/>
                <xs:enumeration value="V"/>
                <xs:enumeration value="W"/>
                <xs:enumeration value="X"/>
                <xs:enumeration value="Y"/>
                <xs:enumeration value="Z"/>
            </xs:restriction>
        </xs:simpleType>
        <!--  Types Geographiques  -->
        <xs:simpleType name="type_Latitude">
            <xs:restriction base="xs:double">
                <xs:minInclusive value="-90"/>
                <xs:maxInclusive value="90"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="type_Longitude">
            <xs:restriction base="xs:double">
                <xs:minInclusive value="-180"/>
                <xs:maxInclusive value="180"/>
            </xs:restriction>
        </xs:simpleType>
        <!--  Type MOB_TYPE  -->
        <xs:simpleType name="MOB_TYPE">
            <xs:restriction base="xs:string">
                <xs:enumeration value="UNKNOWN"/>
                <xs:enumeration value="TRACKED"/>
                <xs:enumeration value="WHEELED"/>
                <xs:enumeration value="ROTARY_WINGS"/>
            </xs:restriction>
        </xs:simpleType>
        <!--  Type TRACK_DOT_TYPE  -->
        <xs:simpleType name="TRACK_DOT_TYPE">
            <xs:annotation>
                <xs:documentation>The items has to be defined according to
the capacity of the sensor</xs:documentation>
            </xs:annotation>
            <xs:restriction base="xs:string">
                <xs:enumeration value="UNKNOWN"/>
                <xs:enumeration value="ON_ROAD"/>
                <xs:enumeration value="OFF_ROAD"/>
                <xs:enumeration value="AIR"/>
                <xs:enumeration value="SURFACE"/>
            </xs:restriction>
        </xs:simpleType>
        <!--  Type Ellipse  -->
        <xs:complexType name="ELLIPSE_TYPE">
            <xs:sequence>
                <xs:element name="FIRST_AXIS" type="xs:integer"/>
                <xs:element name="FIRST_AXIS_ORIENTATION">
                    <xs:simpleType>
                        <xs:restriction base="xs:double">
                            <xs:minInclusive value="-180"/>
                            <xs:maxInclusive value="180"/>
                        </xs:restriction>
                    </xs:simpleType>
```

```
                </xs:element>
                <xs:element name="SECOND_AXIS" type="xs:integer"/>
            </xs:sequence>
        </xs:complexType>

        <!-- Type GEOREF -->
        <xs:simpleType name="ELLIPSOID_TYPE">
            <xs:restriction base="xs:string">
                <xs:enumeration value="WGS84"/>
            </xs:restriction>
        </xs:simpleType>


        <!-- ************************** -->
        <!--         Type Complex      -->
        <!-- ************************** -->
        <!--  Type TRACK_STATE  -->
        <xs:complexType name="TRACK_STATE">
            <xs:all>
                <xs:element name="LAT" type="type_Latitude"/>
                <xs:element name="LON" type="type_Longitude"/>
                <xs:element name="ALTITUDE" type="xs:double" minOccurs="0"/>
                <xs:element        name="ELLIPSE"        type="ELLIPSE_TYPE"
minOccurs="0"/>
                <xs:element name="DTG" type="type_Date_Time"/>
                <xs:element name="ROUTE" type="xs:decimal" minOccurs="0"/>
                <xs:element name="SPEED" type="xs:decimal" minOccurs="0"/>
            </xs:all>
        </xs:complexType>
        <!--  Type TRACK  -->
        <xs:complexType name="TRACK_TYPE">
            <xs:sequence>
                <xs:element name="NAME" type="xs:string" minOccurs="0"/>
                <xs:element name="ID" type="xs:integer"/>

                <xs:element name="MOB_TYPE" type="MOB_TYPE"/>
                <xs:element name="SYST_STATUS" type="TRACK_SYST_STATUS"/>
                <xs:element       name="DOT_TYPE"       type="TRACK_DOT_TYPE"
minOccurs="0"/>

                <xs:element          name="STATE"          type="TRACK_STATE"
maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>


        <!-- Type HEADER_TYPE -->
        <xs:complexType name="HEADER_TYPE">
            <xs:sequence>

                <xs:element name="MSGCONTENTS" type="xs:string"/>

                <xs:element name="MSG_TIMESTAMP" type="type_Date_Time"/>
                <xs:element name="ELLIPSOID" type="ELLIPSOID_TYPE"/>
                <xs:element name="TIMEZONE" type="TIMEZONE_TYPE"/>
            </xs:sequence>
        </xs:complexType>
        <!--  Type SENSOR_DESCRIPTION  -->
```

```
      <xs:complexType name="SENSOR_DESCRIPTION">
            <xs:sequence>
                  <xs:element name="NAME" type="xs:string" minOccurs="0"/>
                  <xs:element name="ID"/>
                  <xs:element name="TYPE" type="SENSOR_TYPE"/>
                  <xs:element name="LAT" type="type_Latitude" minOccurs="0"/>
                  <xs:element name="LON" type="type_Longitude" minOccurs="0"/>
                  <xs:element name="ALTITUDE" type="xs:double" minOccurs="0"/>
                  <xs:element     name="TIMELINE"     type="TIME_SEQUENCE_TYPE"
minOccurs="0"/>
            </xs:sequence>
      </xs:complexType>
      <!--  Type SUR_REPORT_DESCRIPTION  -->
      <xs:complexType name="SURV_REPORT_DESCRIPTION">
            <xs:sequence>
                  <xs:element name="HEADER" type="HEADER_TYPE"/>
                  <xs:element   name="SENSOR_USED"   type="SENSOR_DESCRIPTION"
minOccurs="0" maxOccurs="unbounded"/>
                  <xs:element   name="TRACK"   type="TRACK_TYPE"   minOccurs="0"
maxOccurs="unbounded">
                        <xs:keyref name="Track_Sensor_Ref" refer="Sensor_Key">
                              <xs:selector xpath="."/>
                              <xs:field xpath="SENSOR_ID"/>
                        </xs:keyref>
                  </xs:element>
            </xs:sequence>
      </xs:complexType>
      <!-- ************************** -->
      <!--       Core document       -->
      <!-- ************************** -->
      <xs:element name="SURV_REPORT" type="SURV_REPORT_DESCRIPTION">
            <xs:annotation>
                  <xs:documentation>Surveillance  report  export  from  No  UAV
MTI</xs:documentation>
            </xs:annotation>
            <xs:key name="Sensor_Key">
                  <xs:selector xpath="SENSOR_USED"/>
                  <xs:field xpath="ID"/>
            </xs:key>
      </xs:element>
</xs:schema>
```

A6.2 MTI TRACKS ADDITIONAL INFORMATION

HEADER INFORMATION

1. Purpose

To give general information related to the Surv Report.

2. Format

Complex type.

3. Example

N/A

4. Comment

All types of Surv Report shall have the same Header.

5. MSGCONTENTS

- Purpose

A short description of contents of the message.

- Format

Free text

- Example

Sensor information indicates enemy activity south of SOUTHAMPTON.

- Comment

This information must be available in the list of reports.

6. MSG TIMESTAMP

- Purpose

Stating the time for the issue of the message.

- Format

Defined in the XSD.

- Example

2004-10-05 T17:41:45

- Comment

7. ELLIPSOID

- Purpose

Stating the geographic reference for all positions in the message.

- Format

Enumeration value defined in the XSD.

- Example

WGS84

- Comment

8. TIMEZONE

- Purpose

Stating the time zone for all timing in the message.

- Format

Enumeration values defined in the XSD.

- Example

Z

- Comment

TRACK

9. NAME

- Purpose

Give short information of the track.

- Format

Free text

- Example

AKS ISTARS TRACK 2006_01_25_01.

- Comment

10. ID

- Purpose

Giving a unique identification of the track. If an ID is reused, this shall be considered as an update of the previous track.

- Format
- Example
- Comment

The ID convention needs to be defined

11. MOB TYPE

- Purpose

Stating the kind of object.

- Format

Enumeration values defined in the XSD.

- Example

TRACKED

- Comment

13. DOT_TYPE

- Purpose

Stating the objects capability or possibility to follow roads.

- Format
- Example
- Comment

13. SENSOR ID

- Purpose

Identify the sensor(s) that provided the raw information.

- Format
- Example
- Comment

The referenced sensors must be transmitted in the same message.

14. STATE

- Purpose

Defines one single state (adjusted plot).

- Format

Complex type

- Example

N/A

- Comment

N/A

- LAT
    - Purpose:

The north position of the State in decimal degrees according the ellipsoid of the header.

    - Format:
    - Example:
    - Comment
- LON
    - Purpose:

The east position of the state in decimal degrees according the ellipsoid of the header.

    - Format:
    - Example:
    - Comment
- ALTITUDE
    - Purpose:

The altitude above sea level of the State in decimal meters.

    - Format:
    - Example:
    - Comment
- ELLIPSE
    - Purpose:

Indicates the possible uncertainty in the position (by stating an ellipse wherein the position will be).

    - Format:
    - Example:
    - Comment
    - First Axis

Purpose:
Indicates the length of the first axis in meters.
Format:
Integer
Example:
100
Comment

- First Axis orientation

Purpose:
Indicates the orientation of the first axis in degrees (0 degrees = North).
Format:
Integer
Example:
100
Comment

- Second Axis

Purpose:
Indicates the length of the second axis in meters.
Format:
Integer
Example:
100
Comment

- DTG
  - Purpose:

States the time of the detected plot.

  - Format:
  - Example:
  - Comment
- ROUTE
  - Purpose:

States the direction in decimal degrees from the north of the object that the plot represents.

  - Format:
  - Example:
  - Comment
- SPEED
  - Purpose:

States the speed in $m.s^{-1}$ of the object that the plot represents.

  - Format:
  - Example:
  - Comment

Appendix 7: XML Schema for the UDDI Publishing API Extensions

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2006 sp1 U (http://www.altova.com) by
Raymond Haakseth (Norwegian Defence Research Establishment) -->
<xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema
xmlns:sp="http://www.rtg027.nato/xmlns/cwid/servicePublishing"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:uddi="urn:uddi-org:api_v3"
targetNamespace=http://www.rtg027.nato/xmlns/cwid/servicePublishing
elementFormDefault="qualified"
attributeFormDefault="unqualified">
   <!-- author: Raymond Haakseth (Raymond.Haakseth@ffi.no) -->
   <xs:import namespace="urn:uddi-org:api_v3" schemaLocation="uddi_v3.xsd"/>
   <xs:element name="publishServices" type="sp:publishServices"
                   final="restriction">
      <xs:annotation>
         <xs:documentation>
            Represent the message to be used to publish services.
         </xs:documentation>
      </xs:annotation>
   </xs:element>
   <xs:complexType name="publishServices" final="restriction">
      <xs:sequence>
         <xs:element name="authToken" type="uddi:authToken"/>
         <xs:element name="service" type="sp:service" maxOccurs="unbounded"/>
      </xs:sequence>
   </xs:complexType>
   <xs:complexType name="service" final="restriction">
      <xs:sequence>
         <xs:element name="serviceName" type="xs:string"/>
         <xs:element name="description" type="xs:string"/>
         <xs:element name="validUntil" type="xs:dateTime"/>
         <xs:element name="serviceEndpoint" type="xs:anyURI"/>
         <xs:element name="distinguishedName" type="xs:string"/>
         <xs:element name="wsdlReference" type="sp:wsdlReference">
            <xs:annotation>
               <xs:documentation>
                  Contains a reference to  the abstract part of a
                  well defined wsdl document.
               </xs:documentation>
            </xs:annotation>
         </xs:element>
         <xs:element name="businessIdString" type="xs:string"/>
         <xs:element name="serviceKey" type="uddi:serviceKey" minOccurs="0">
            <xs:annotation>
               <xs:documentation>
                  The service key is needed if this is an update.
               </xs:documentation>
            </xs:annotation>
         </xs:element>
         <xs:element name="serviceTaxonomy" type="xs:string" minOccurs="0"/>
         <xs:element name="topic" type="sp:topic" minOccurs="0"
                         maxOccurs="unbounded"/>
         <xs:element name="position" type="xs:anyURI" minOccurs="0">
            <xs:annotation>
               <xs:documentation>
```

```
                  URI to the Web Service implementing platform position.
               </xs:documentation>
            </xs:annotation>
         </xs:element>
         <xs:element         name="coverageArea"         type="sp:coverageArea"
minOccurs="0"/>
      </xs:sequence>
   </xs:complexType>
   <xs:simpleType name="wsdlReference">
      <xs:restriction base="xs:string">
         <xs:pattern value="ACP_.*Interface"/>
      </xs:restriction>
   </xs:simpleType>
   <xs:simpleType name="topic" final="restriction">
      <xs:restriction base="xs:string">
         <xs:pattern value="ACP_.*Topic"/>
      </xs:restriction>
   </xs:simpleType>
   <xs:complexType name="coverageArea" final="restriction">
      <xs:sequence>
         <xs:element name="upperLeft" type="sp:upperLeft"/>
         <xs:element name="lowerRight" type="sp:lowerRight"/>
      </xs:sequence>
   </xs:complexType>
   <xs:complexType name="upperLeft" final="restriction">
      <xs:sequence>
         <xs:element name="longitude" type="sp:Longitude"/>
         <xs:element name="latitude" type="sp:Latitude"/>
      </xs:sequence>
   </xs:complexType>
   <xs:complexType name="lowerRight" final="restriction">
      <xs:sequence>
         <xs:element name="longitude" type="sp:Longitude"/>
         <xs:element name="latitude" type="sp:Latitude"/>
      </xs:sequence>
   </xs:complexType>
   <xs:complexType name="Longitude" final="restriction">
      <xs:sequence>
         <xs:element name="degrees" type="sp:LongitudeDegree">
            <xs:annotation>
               <xs:documentation>
                  Value range: -180 - 180
               </xs:documentation>
            </xs:annotation>
         </xs:element>
         <xs:element name="minutes" type="sp:Minutes">
            <xs:annotation>
               <xs:documentation>
                  Value range: 0 - 59
               </xs:documentation>
            </xs:annotation>
         </xs:element>
         <xs:element name="seconds" type="sp:Seconds">
            <xs:annotation>
               <xs:documentation>
                  Value range: 0 - 59
               </xs:documentation>
            </xs:annotation>
         </xs:element>
      </xs:sequence>
```

```
   </xs:complexType>
   <xs:complexType name="Latitude" final="restriction">
      <xs:sequence>
         <xs:element name="degrees" type="sp:LatitudeDegree">
            <xs:annotation>
               <xs:documentation>
                  Value range: -90 - 90
               </xs:documentation>
            </xs:annotation>
         </xs:element>
         <xs:element name="minutes" type="sp:Minutes">
            <xs:annotation>
               <xs:documentation>
                  Value range: 0 - 59
               </xs:documentation>
            </xs:annotation>
         </xs:element>
         <xs:element name="seconds" type="sp:Seconds">
            <xs:annotation>
               <xs:documentation>
                  Value range: 0 - 59
               </xs:documentation>
            </xs:annotation>
         </xs:element>
      </xs:sequence>
   </xs:complexType>
   <xs:simpleType name="LatitudeDegree" final="restriction">
      <xs:restriction base="xs:int">
         <xs:minInclusive value="-90"/>
         <xs:maxInclusive value="90"/>
      </xs:restriction>
   </xs:simpleType>
   <xs:simpleType name="Seconds" final="restriction">
      <xs:restriction base="xs:int">
         <xs:minInclusive value="0"/>
         <xs:maxInclusive value="59"/>
      </xs:restriction>
   </xs:simpleType>
   <xs:simpleType name="Minutes" final="restriction">
      <xs:restriction base="xs:int">
         <xs:minInclusive value="0"/>
         <xs:maxInclusive value="59"/>
      </xs:restriction>
   </xs:simpleType>
   <xs:simpleType name="LongitudeDegree" final="restriction">
      <xs:restriction base="xs:int">
         <xs:minInclusive value="-180"/>
         <xs:maxInclusive value="180"/>
      </xs:restriction>
   </xs:simpleType>
   <xs:element          name="resetRegistry"          type="sp:resetRegistry"
final="restriction">
      <xs:annotation>
         <xs:documentation>
            Element used to reset the registered services of the nation
            selected in nation. Strictly used for demo administration.
         </xs:documentation>
      </xs:annotation>
   </xs:element>
   <xs:complexType name="resetRegistry" final="restriction">
```

```
      <xs:sequence>
         <xs:element name="authToken" type="uddi:authToken"/>
         <xs:element name="nation" type="sp:nation"/>
      </xs:sequence>
   </xs:complexType>
   <xs:simpleType name="nation" final="restriction">
      <xs:restriction base="xs:string">
         <xs:enumeration value="FR"/>
         <xs:enumeration value="GE"/>
         <xs:enumeration value="NL"/>
         <xs:enumeration value="NO"/>
         <xs:enumeration value="PL"/>
      </xs:restriction>
   </xs:simpleType>
</xs:schema>
```

Appendix 8: Position Service Description

The positionService.wsdl file describes a simple request/response web service that can be used to retrieve position information. The format of the return value from this service is specified in the position.xsd file.

A service/ asset that expose information of its current position must implement the position service described below.

A8.1 POSITIONSERVICE.WSDL

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:pos="http://www.rtg027.nato/xmlns/cwid/positionService/position"
xmlns:tns="http://www.rtg027.nato/xmlns/cwid/positionService"
targetNamespace="http://www.rtg027.nato/xmlns/cwid/positionService">
	<!-- TYPES-->
	<types>
		<xs:schema targetNamespace="http://www.rtg027.nato/xmlns/cwid/positionService">
			<xs:import schemaLocation="position.xsd"
namespace="http://www.rtg027.nato/xmlns/cwid/positionService/position"/>
		</xs:schema>
	</types>
	<!-- MESSAGES -->
	<message name="getPositionRequest">
	</message>
	<message name="getPositionResponse">
		<part name="result" element="pos:position"/>
	</message>
	<message name="getPositionFault"/>
	<!--PORT TYPE -->
	<portType name="positionServicePortType">
		<operation name="getPosition">
			<input message="tns:getPositionRequest"/>
			<output message="tns:getPositionResponse"/>
			<fault name="getPositionFault" message="tns:getPositionFault"/>
		</operation>
	</portType>
	<!-- BINDING -->
	<binding name="positionServiceBinding" type="tns:positionServicePortType">
		<soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
		<operation name="getPosition">
			<input>
				<soap:body use="literal"/>
			</input>
			<output>
				<soap:body use="literal"/>
```

```
                </output>
                <fault name="getPositionFault">
                        <soap:fault name="faultMessage" use="literal"/>
                </fault>
        </operation>
</binding>
<!-- SERVICE -->
<service name="positionService">
        <port name="position" binding="tns:positionServiceBinding">
                <soap:address location="http://some.endpoint"/>
        </port>
</service>
</definitions>
```

A8.2 POSITION.XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2006 sp1 U (http://www.altova.com) by Raymond Haakseth (Norwegian Defence Research Establishment) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:tns="http://www.rtg027.nato/xmlns/cwid/positionService/position"
targetNamespace="http://www.rtg027.nato/xmlns/cwid/positionService/position"
elementFormDefault="qualified" attributeFormDefault="unqualified">
        <xs:element name="position">
                <xs:complexType>
                        <xs:sequence>
                                <xs:element name="longitude" type="tns:Longitude"/>
                                <xs:element name="latitude" type="tns:Latitude"/>
                        </xs:sequence>
                </xs:complexType>
        </xs:element>
        <xs:complexType name="Longitude">
                <xs:sequence>
                        <xs:element name="degrees" type="tns:LongitudeDegree">
                                <xs:annotation>
                                        <xs:documentation>Value range: -180 - 180</xs:documentation>
                                </xs:annotation>
                        </xs:element>
                        <xs:element name="minutes" type="tns:Minutes">
                                <xs:annotation>
                                        <xs:documentation>Value range: 0 - 59</xs:documentation>
                                </xs:annotation>
                        </xs:element>
                        <xs:element name="seconds" type="tns:Seconds">
                                <xs:annotation>
                                        <xs:documentation>Value range: 0 - 59</xs:documentation>
                                </xs:annotation>
                        </xs:element>
                </xs:sequence>
        </xs:complexType>
        <xs:complexType name="Latitude">
                <xs:sequence>
```

```
                <xs:element name="degrees" type="tns:LatitudeDegree">
                        <xs:annotation>
                                <xs:documentation>Value range: -90 - 90</xs:documentation>
                        </xs:annotation>
                </xs:element>
                <xs:element name="minutes" type="tns:Minutes">
                        <xs:annotation>
                                <xs:documentation>Value range: 0 - 59</xs:documentation>
                        </xs:annotation>
                </xs:element>
                <xs:element name="seconds" type="tns:Seconds">
                        <xs:annotation>
                                <xs:documentation>Value range: 0 - 59</xs:documentation>
                        </xs:annotation>
                </xs:element>
            </xs:sequence>
        </xs:complexType>
        <xs:simpleType name="LongitudeDegree">
            <xs:restriction base="xs:int">
                <xs:minInclusive value="-180"/>
                <xs:maxInclusive value="180"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="Minutes">
            <xs:restriction base="xs:int">
                <xs:minInclusive value="0"/>
                <xs:maxInclusive value="59"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="Seconds">
            <xs:restriction base="xs:int">
                <xs:minInclusive value="0"/>
                <xs:maxInclusive value="59"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="LatitudeDegree">
            <xs:restriction base="xs:int">
                <xs:minInclusive value="-90"/>
                <xs:maxInclusive value="90"/>
            </xs:restriction>
        </xs:simpleType>
</xs:schema>
```

Appendix 9: The XML Schema of the LDAP Synchronization Component

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:sbn="http://www.rtg027.nato/xmlns/cwid/synchronization_between_nations"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.rtg027.nato/xmlns/cwid/synchronization_between_nations"
elementFormDefault="qualified" attributeFormDefault="qualified" version="draft">
	<!-- Basic Types -->
	<xsd:simpleType name="LDIFDataType">
		<xsd:restriction base="xsd:string"/>
	</xsd:simpleType>
	<xsd:simpleType name="LDAPSynchronisationCounterType">
		<xsd:restriction base="xsd:positiveInteger"/>
	</xsd:simpleType>
	<xsd:simpleType name="LDAPSynchronisationTimestampType">
		<xsd:restriction base="xsd:dateTime"/>
	</xsd:simpleType>
	<!-- Type representing the header of an LDAP directory synchronisation topic -->
	<xsd:complexType name="LDAPSynchronisationTopicHeaderType">
		<xsd:sequence>
			<xsd:element name="counter" type="sbn:LDAPSynchronisationCounterType"/>
			<xsd:element name="timestamp"
type="sbn:LDAPSynchronisationTimestampType"/>
		</xsd:sequence>
	</xsd:complexType>
	<!-- Type representing the payload of an LDAP directory synchronisation topic -->
	<xsd:complexType name="LDAPSynchronisationTopicPayLoadType">
		<xsd:sequence>
			<xsd:element name="LDIFData" type="sbn:LDIFDataType"/>
		</xsd:sequence>
	</xsd:complexType>
	<!-- Type representing an LDAP directory synchronisation topic -->
	<xsd:complexType name="LDAPSynchronisationTopicType">
		<xsd:sequence>
			<xsd:element name="header"
type="sbn:LDAPSynchronisationTopicHeaderType"/>
			<xsd:element name="payload"
type="sbn:LDAPSynchronisationTopicPayLoadType"/>
		</xsd:sequence>
	</xsd:complexType>
	<!-- The LDAP directory synchronisation topic itself -->
	<xsd:element name="LDAPSynchronisationTopic"
type="sbn:LDAPSynchronisationTopicType"/>
</xsd:schema>
```

Appendix 10: MIP Elements Selection for RTG Demonstration

Version of object-oriented or RDBMS (To be decided) MIP XML Schema to use: C2IEDM 6.15e
France will provide a discussion paper beginning next week (6 February).

The following list of MIP objects should be included (the MIP elements in grey will not be used for demonstration purposes):

ObjectType
- FacilityType
  - BridgeType
  - MilitaryObstacleType
- FeatureType
  - ControlFeatureType
  - GeographicFeatureType
- MaterielType
  - EquipmentType
    - AircraftType
    - VehicleType
    - VesselType
  - ConsumableMaterielType
    - AmmunitionType
- OrganisationType
  - GovernmentOrganisationType
    - MilitaryOrganisationType
      - UnitType

ObjectItem
- Organisation
  - Unit
- Materiel
- Facility
  - Bridge
  - MilitaryObstacle
    - Minefield
- Feature
  - GeographicFeature
  - ControlFeature

ObjectItemStatus
- GeographicFeatureStatus
- ControlFeatureStatus
- FacilityStatus
- MaterielStatus
- OrganisationStatus

Affiliation
- AffiliationExerciseGroup
- AffiliationFunctionalGroup
- AffiliationGeopolitical

ObjectItemAffiliation
ObjectItemAssociation
ObjectItemAssociationStatus

ObjectItemType
Holding mandatory for AbstractObjectItem
ReportingData
    ReportingDataAbsoluteTiming
Reference mandatory for AbstractRepportingData
Action
    ActionEvent
ObjectItemLocation
Location
    Line
    Point
        AbsolutePoint
    Surface
        PolygonArea
LinePoint

Context
ContextAssesment indirectly mandatory for AbstractLocation
ContextReportingDataAssociation indirectly mandatory for AbstractLocation
VerticleDistance

We now have a preliminary selection of objects.
The next steps should be:

- examine all attributes in these objects to select which ones to use (in light of optionality and requirements)
- examine values in all *-code attributes, and maybe select a reduced set to implement where the number of values is large (like in ActionEventCategoryCode)

A10.1 ATTRIBUTE SELECTION

As a general rule we keep all mandatory attributes (MA). If a mandatory attribute is not used for the demonstration, we use the value “unknown” if possible. Otherwise we can use an existing value to be used as “unknown” specifically for the demonstration.

Also as a general rule we leave out all optional (OP) attributes (not shown in table below). If a need for an optional attribute arises, it may be added after discussion in the group.

The following table shows the entities with their attributes and optionality (entities marked yellow are not relevant for the demonstration).

Entity Name	Attribute Names	Optionality
ABSOLUTE-POINT	absolute-point-id (PK) (FK) absolute-point-latitude-coordinate absolute-point-longitude-coordinate absolute-point-angular-precision-code absolute-point-vertical-distance-id (FK)	MA MA MA OP OP
ACTION	action-id (PK) action-category-code action-name	MA MA OP
ACTION-EVENT	action-event-id (PK) (FK) action-event-category-code	MA MA
AFFILIATION	affiliation-id (PK) affiliation-category-code	MA MA
AFFILIATION-EXERCISE-GROUP	affiliation-id (PK) (FK) affiliation-exercise-group-name	MA MA
AFFILIATION-FUNCTIONAL-GROUP	affiliation-id (PK) (FK) affiliation-functional-group-code affiliation-functional-group-name	MA MA MA
AFFILIATION-GEOPOLITICAL	affiliation-id (PK) (FK) affiliation-geopolitical-code	MA MA
AIRCRAFT-TYPE	aircraft-type-id (PK) (FK) aircraft-type-category-code aircraft-type-subcategory-code	MA MA OP
AMMUNITION-TYPE	ammunition-type-id (PK) (FK) ammunition-type-category-code ammunition-type-calibre-text	MA MA OP
BRIDGE	bridge-id (PK) (FK) bridge-longest-span-length-dimension bridge-span-quantity bridge-usage-code	MA OP OP OP
BRIDGE-TYPE	bridge-type-id (PK) (FK) bridge-type-design-type-code	MA MA
CONSUMABLE-MATERIEL-TYPE	consumable-materiel-type-id (PK) (FK) consumable-materiel-type-category-code consumable-materiel-type-subcategory-code consumable-materiel-type-hazard-code consumable-materiel-type-issuing-element-code consumable-materiel-type-issuing-quantity consumable-materiel-type-issuing-unit-of-measure-code consumable-materiel-type-issuing-weight-quantity consumable-materiel-type-perishability-indicator-code	MA MA OP OP OP OP OP OP OP
CONTROL-FEATURE	control-feature-id (PK) (FK) control-feature-category-code	MA MA
CONTROL-FEATURE-STATUS	control-feature-status-id (PK) (FK) object-item-status-index (PK) (FK) control-feature-status-investigation-status-code control-feature-status-nbc-threat-level-code control-feature-status-security-status-code control-feature-status-usage-status-code	MA MA OP OP OP MA
CONTROL-FEATURE-TYPE	control-feature-type-id (PK) (FK) control-feature-type-category-code	MA MA
EQUIPMENT-TYPE	equipment-type-id (PK) (FK) equipment-type-category-code equipment-type-loaded-weight-quantity equipment-type-unloaded-weight-quantity	MA MA OP OP

Entity Name	Attribute Names	Optionality
FACILITY	facility-id (PK) (FK) facility-category-code facility-primary-construction-material-code facility-height-dimension facility-length-dimension facility-width-dimension	MA MA OP OP OP OP
FACILITY-STATUS	facility-status-id (PK) (FK) object-item-status-index (PK) (FK) facility-status-category-code facility-status-demolition-status-code facility-status-mine-presence-code facility-status-occupation-program-indicator-code facility-status-operational-status-code facility-status-operational-status-qualifier-code facility-status-reserve-indicator-code facility-status-security-status-code facility-status-usage-status-code	MA MA MA OP OP OP MA OP OP OP OP
FACILITY-TYPE	facility-type-id (PK) (FK) facility-type-category-code	MA MA
FEATURE	feature-id (PK) (FK) feature-category-code	MA MA
FEATURE-TYPE	feature-type-id (PK) (FK) feature-type-category-code	MA MA
GEOGRAPHIC-FEATURE	geographic-feature-id (PK) (FK)	MA
GEOGRAPHIC-FEATURE-STATUS	geographic-feature-status-id (PK) (FK) object-item-status-index (PK) (FK) geographic-feature-status-code geographic-feature-status-demolition-status-code geographic-feature-status-mine-presence-code geographic-feature-status-recirculation-indicator-code geographic-feature-status-surface-condition-code geographic-feature-status-surface-firmness-code	MA MA MA OP OP OP OP OP
GEOGRAPHIC-FEATURE-TYPE	geographic-feature-type-id (PK) (FK) geographic-feature-type-category-code	MA MA
GOVERNMENT-ORGANISATION-TYPE	government-organisation-type-id (PK) (FK) government-organisation-type-category-code government-organisation-type-main-activity-code	MA MA OP
HOLDING	object-item-id (PK) (FK) object-type-id (PK) (FK) holding-index (PK) holding-operational-quantity holding-total-quantity reporting-data-id (FK)	MA MA MA OP OP MA
LINE	line-id (PK) (FK)	MA
LINE-POINT	line-id (PK) (FK) line-point-index (PK) line-point-sequence-quantity line-point-point-id (FK)	MA MA MA MA
LOCATION	location-id (PK) location-category-code	MA MA

Entity Name	Attribute Names	Optionality
MATERIEL	materiel-id (PK) (FK)	MA
	materiel-serial-number-identification-text	OP
	materiel-lot-identification-text	OP
	materiel-body-colour-code	OP
	materiel-marking-code	OP
	materiel-marking-colour-code	OP
MATERIEL-STATUS	materiel-status-id (PK) (FK)	MA
	object-item-status-index (PK) (FK)	MA
	materiel-status-category-code	MA
	materiel-status-demolition-status-code	OP
	materiel-status-operational-status-code	MA
	materiel-status-operational-status-qualifier-code	OP
	materiel-status-operational-status-mode-code	OP
	materiel-status-reserve-indicator-code	OP
	materiel-status-safety-status-code	OP
	materiel-status-usage-status-code	OP
MATERIEL-TYPE	materiel-type-id (PK) (FK)	MA
	materiel-type-category-code	MA
	materiel-type-reportable-item-text	OP
	materiel-type-stock-number-text	OP
	materiel-type-supply-class-code	OP
	materiel-type-maximum-height-dimension	OP
	materiel-type-maximum-length-dimension	OP
	materiel-type-maximum-width-dimension	OP
MILITARY-OBSTACLE	military-obstacle-id (PK) (FK)	MA
	military-obstacle-category-code	MA
MILITARY-OBSTACLE-TYPE	military-obstacle-type-id (PK) (FK)	MA
	military-obstacle-type-category-code	MA
MILITARY-ORGANISATION-TYPE	military-organisation-type-id (PK) (FK)	MA
	military-organisation-type-category-code	MA
	military-organisation-type-service-code	MA
MINEFIELD	minefield-id (PK) (FK)	MA
	minefield-depth-placement-code	OP
	minefield-mine-spacing-dimension	OP
	minefield-pattern-code	OP
	minefield-persistence-code	OP
	minefield-purpose-code	OP
	minefield-stopping-power-code	OP
OBJECT-ITEM	object-item-id (PK)	MA
	object-item-category-code	MA
	object-item-name	MA
	object-item-alternate-identification-text	OP
OBJECT-ITEM-AFFILIATION	object-item-id (PK) (FK)	MA
	affiliation-id (PK) (FK)	MA
	object-item-affiliation-index (PK)	MA
	reporting-data-id (FK)	MA
OBJECT-ITEM-ASSOCIATION	object-item-association-subject-object-item-id (FK) (PK)	MA
	object-item-association-object-object-item-id (FK) (PK)	MA
	object-item-association-index (PK)	MA
	object-item-association-category-code	MA
	object-item-association-subcategory-code	OP
	action-task-id (FK)	

Entity Name	Attribute Names	Optionality
OBJECT-ITEM-ASSOCIATION-STATUS	object-item-association-subject-object-item-id (FK) (PK)	MA
	object-item-association-object-object-item-id (FK) (PK)	MA
	object-item-association-index (FK) (PK)	MA
	object-item-association-status-index (PK)	MA
	object-item-association-status-category-code	MA
	reporting-data-id (FK)	MA
OBJECT-ITEM-LOCATION	object-item-id (PK) (FK)	MA
	location-id (PK) (FK)	MA
	object-item-location-index (PK)	MA
	object-item-location-accuracy-quantity	OP
	object-item-location-bearing-angle	OP
	object-item-location-bearing-accuracy-angle	OP
	object-item-location-speed-rate	OP
	object-item-location-speed-accuracy-rate	OP
	object-item-location-use-category-code	OP
	reporting-data-id (FK)	MA
OBJECT-ITEM-STATUS	object-item-id (PK) (FK)	MA
	object-item-status-index (PK)	MA
	object-item-status-category-code	MA
	object-item-status-hostility-code	MA
	object-item-status-booby-trap-indicator-code	OP
	object-item-status-emission-control-code	OP
	reporting-data-id (FK)	MA
OBJECT-ITEM-TYPE	object-item-id (PK) (FK)	MA
	object-type-id (PK) (FK)	MA
	object-item-type-index (PK)	MA
	reporting-data-id (FK)	MA
OBJECT-TYPE	object-type-id (PK)	MA
	object-type-category-code	MA
	object-type-dummy-indicator-code	MA
	object-type-name	MA
ORGANISATION	organisation-id (PK) (FK)	MA
	organisation-category-code	MA
	organisation-nickname-name	OP
ORGANISATION-STATUS	organisation-status-id (PK) (FK)	MA
	object-item-status-index (PK) (FK)	MA
	organisation-status-operational-status-code	MA
	organisation-status-operational-status-qualifier-code	OP
	organisation-status-availability-code	OP
	organisation-status-command-and-control-role-code	OP
	organisation-status-commitment-status-code	OP
	organisation-status-fire-mode-code	OP
	organisation-status-nbc-dress-state-code	OP
	organisation-status-radiation-dose-code	OP
	organisation-status-readiness-code	OP
	organisation-status-readiness-duration	OP
	organisation-status-reinforcement-code	OP
	organisation-status-reserve-indicator-code	OP
	organisation-status-usage-status-code	OP
ORGANISATION-TYPE	organisation-type-id (PK) (FK)	MA
	organisation-type-category-code	MA
	organisation-type-command-function-indicator-code	MA
	organisation-type-command-and-control-category-code	OP
	organisation-type-description-text	OP

Entity Name	Attribute Names	Optionality
POINT	point-id (PK) (FK)	MA
	point-category-code	MA
REFERENCE	reference-id (PK)	MA
	reference-description-text	OP
	reference-security-classification-code	OP
	reference-source-text	OP
	reference-transmittal-type-code	OP
REPORTING-DATA	reporting-data-id (PK)	MA
	reporting-data-accuracy-code	OP
	reporting-data-category-code	MA
	reporting-data-counting-indicator-code	OP
	reporting-data-credibility-code	OP
	reporting-data-reliability-code	OP
	reporting-data-source-type-code	OP
	reporting-data-reporting-date	MA
	reporting-data-reporting-time	MA
	reporting-data-timing-category-code	MA
	reference-id (FK)	OP
	reporting-data-reporting-organisation-id (FK)	MA
REPORTING-DATA-ABSOLUTE-TIMING	reporting-data-absolute-timing-reporting-data-id (PK) (FK)	MA
	reporting-data-absolute-timing-effective-start-date	MA
	reporting-data-absolute-timing-effective-start-time	OP
	reporting-data-absolute-timing-effective-end-date	OP
	reporting-data-absolute-timing-effective-end-time	OP
SURFACE	surface-id (PK) (FK)	MA
	surface-category-code	MA
UNIT	unit-id (PK) (FK)	MA
	unit-formal-abbreviated-name	MA
UNIT-TYPE	unit-type-id (PK) (FK)	MA
	unit-type-category-code	MA
	unit-type-arm-category-code	MA
	unit-type-arm-specialisation-code	OP
	unit-type-supplementary-specialisation-code	OP
	unit-type-general-mobility-code	OP
	unit-type-qualifier-code	OP
	unit-type-size-code	MA
	unit-type-principal-equipment-type-id (FK)	OP
	unit-type-supported-military-organisation-type-id (FK)	OP
VEHICLE-TYPE	vehicle-type-id (PK) (FK)	MA
	vehicle-type-category-code	MA
VESSEL-TYPE	vessel-type-id (PK) (FK)	MA
	vessel-type-category-code	MA
	vessel-type-subcategory-code	OP

The following table shows the mandatory attributes from the table above that are not PKs.
Attributes with a list of codes that must be analyzed are marked in bold. The number of values is indicated, and the values for some with only a few values allowed.

Entity Name	Attribute Names	Number of values	
ABSOLUTE-POINT	absolute-point-latitude-coordinate absolute-point-longitude-coordinate		
ACTION	**action-category-code**	**2**	**ACTEV** **ACTTA**
ACTION-EVENT	**action-event-category-code**	**323**	
AFFILIATION	**affiliation-category-code**	**6**	**Only use geopolitical code**
AFFILIATION-EXERCISE-GROUP	affiliation-exercise-group-name		
AFFILIATION-FUNCTIONAL-GROUP	**affiliation-functional-group-code** affiliation-functional-group-name	**3**	**CRIMIN** **MULTIN** **TERRST**
AFFILIATION-GEOPOLITICAL	**affiliation-geopolitical-code**	**241**	Use only FXX,NL,GE, PL,NO,SP and NOS
AIRCRAFT-TYPE	**aircraft-type-category-code**	**8**	
AMMUNITION-TYPE	**ammunition-type-category-code**	**36**	
BRIDGE-TYPE	**bridge-type-design-type-code**	**16**	
CONSUMABLE-MATERIEL-TYPE	**consumable-materiel-type-category-code**	**19**	
CONTROL-FEATURE	**control-feature-category-code**	**2**	**NOS** **ROUTE**
CONTROL-FEATURE-STATUS	**control-feature-status-usage-status-code**	**2**	**ACTIVE** **DEACTV**
CONTROL-FEATURE-TYPE	**control-feature-type-category-code**	**262**	
EQUIPMENT-TYPE	**equipment-type-category-code**	**9**	**Only use Aircraft, Vessel, Vehicle**
FACILITY	**facility-category-code**	**7**	
FACILITY-STATUS	**facility-status-category-code** **facility-status-operational-status-code**	**2** **6**	**MEDFST** **NOS**
FACILITY-TYPE	**facility-type-category-code**	**188**	
FEATURE	**feature-category-code**	**4**	**CF** **GF** **METFT** **NOS**
FEATURE-TYPE	**feature-type-category-code**	**4**	**CF** **GF** **METFTT** **NOS**
GEOGRAPHIC-FEATURE-STATUS	**geographic-feature-status-code**	**7**	
GEOGRAPHIC-FEATURE-TYPE	**geographic-feature-type-category-code**	**31**	
GOVERNMENT-ORGANISATION-TYPE	**government-organisation-type-category-code**	**5**	**Only use MILORG**
HOLDING	reporting-data-id (FK)		
LINE-POINT	line-point-sequence-quantity line-point-point-id (FK)		

Entity Name	Attribute Names	Number of values	
LOCATION	**location-category-code**	**4**	**VL** **LN (line)** **PT (Point)** **SURFAC**
MATERIEL-STATUS	**materiel-status-category-code** **materiel-status-operational-status-code**	**2** **6**	**Only use NOS** **MOPS** **NKN** **NOP** **OPR** **SOPS** **TNOPS**
MATERIEL-TYPE	**materiel-type-category-code**	**3**	**CM** **EQ(Equipment)** **NOS**
MILITARY-OBSTACLE	**military-obstacle-category-code**	**2**	**MINE** **NOS**
MILITARY-OBSTACLE-TYPE	**military-obstacle-type-category-code**	**29**	
MILITARY-ORGANISATION-TYPE	**military-organisation-type-category-code** **military-organisation-type-service-code**	**5** **15**	**Only use UNIT-TYPE** **ARMY** **NAVY**
OBJECT-ITEM	**object-item-category-code** object-item-name	**6**	**FA** **FE** **MA (Material)** **OR (Organis.)** **PE** **NKN**
OBJECT-ITEM-AFFILIATION	reporting-data-id (FK)		
OBJECT-ITEM-ASSOCIATION	**object-item-association-category-code** action-task-id (FK)	**94**	**See note 4.**
OBJECT-ITEM-ASSOCIATION-STATUS	**object-item-association-status-category-code** reporting-data-id (FK)	**2**	**Only use START**
OBJECT-ITEM-LOCATION	reporting-data-id (FK)		
OBJECT-ITEM-STATUS	**object-item-status-category-code** **object-item-status-hostility-code** reporting-data-id (FK)	**6** **14**	**OR** **Assumed** **Friend** **Assumed hostile** **Friend** **Hostile** **Neutral** **Pending Unknown**
OBJECT-ITEM-TYPE	reporting-data-id (FK)		
OBJECT-TYPE	**object-type-category-code** **object-type-dummy-indicator-code** object-type-name	**6** **2**	**MA (Material)** **OR (Organis.)** **NO** **YES**

Entity Name	Attribute Names	Number of values	
ORGANISATION	**organisation-category-code**	**3**	CO **UN (Unit)** NOS
ORGANISATION-STATUS	**organisation-status-operational-status-code**	**6**	**MOPS** **NKN** **NOP** **OPR** **SOPS** **TNOPS**
ORGANISATION-TYPE	**organisation-type-category-code** **organisation-type-command-function-indicator-code**	**5** **2**	**Only need GVTORG (Government)** **NO** **YES**
POINT	**point-category-code**	**2**	**Only use ABS**
REPORTING-DATA	**reporting-data-category-code** reporting-data-reporting-date reporting-data-reporting-time **reporting-data-timing-category-code** reporting-data-reporting-organisation-id (FK)	**6** **3**	**Only need Reported** **Only use RDABST**
REPORTING-DATA-ABSOLUTE-TIMING	reporting-data-absolute-timing-effective-start-date		
SURFACE	**surface-category-code**	**7**	
UNIT	unit-formal-abbreviated-name		
UNIT-TYPE	**unit-type-category-code** **unit-type-arm-category-code** **unit-type-size-code** **unit-type-arm-specialisation-code (OP)**	**5** **35** **28** **(130)**	**Need: Combat** **Combat Service Support** **Combat Support** **Not known** **See note 1.** **See note 1.** **See note 2.** **See note 1.**
VEHICLE-TYPE	**vehicle-type-category-code**	**34**	**Ambulance** **Armoured** **Armoured-personnel-carrier** **General-purpose** **NKN**
VESSEL-TYPE	**vessel-type-category-code**	**3**	**NKN** SUBSRF **SURFAC**
	vessel-type-subcategory-code (OP)	**(62)**	**NKN** **See note 3.**

Note 1: We select a limited set of combinations (10-12) of **unit-type-arm-category-code** and **unit-type-arm-specialisation-code** to be sure that every system can display the corresponding symbol.

unit-type-category-code	**unit-type-arm-category-code**	**unit-type-arm-specialisation-code**
Not known	Not known	NULL
Combat	Air defence	NULL
Combat	Anti armour	NULL
Combat	Armour	NULL
Combat	Aviation, rotary wing	Medical evacuation aviation
Combat	Aviation	NULL
Combat	Engineer	Engineer, reconnaissance
Combat	Infantry	NULL
Combat	Reconnaissance	Cavalry
Combat	Not otherwise specified	NULL
Combat support	Signal	NULL
Combat support	NBC	NULL
Combat service support	Maintenance	NULL
Combat service support	Supply	NULL

Note 2: We select a set tailored to the scenario for **unit-type-size-code**:

Corps, Division, Brigade, Battalion, Company, Section, Platoon, Not otherwise specified.

Note 3: Not subsurface types. We select a limited set, including frigates, patrol vessel, merchant ships, fishing boats. And additionally, Landing boat : LNDCRF

Note 4: Only two relations are remaining: between organisation and materiel, and between materiel and materiel.

- Relation between **organisation** (subject) and **materiel** (object) : Controls, Employs, Not known
- Relation between **materiel** and **materiel** : Not otherwise specified

Appendix 11: Evaluation of Compression Methods

A11.1 INTRODUCTION

This appendix summarizes evaluation methodology and performance data that have been collected during tests of a few XML compression methods for the CWID'06 demonstrator.

A11.2 ASSUMPTIONS

We assume that we need to select a single compression/encoding scheme that would perform acceptably well (in terms of compression ratio and compression/decompression speed) and could be integrated with both Java and .NET, as both environments will be used in the demonstrator. This implies that either a C/C++ implementation should be available (then, it is possible to create a Java wrapper using Java Native Interface and a .NET wrapper, with Managed Extensions for C++) or that two separate, compatible implementations must exist.

We decided not to take the ability of a tool to query/modify an compressed document into account; this does not seem to be a real factor.

A11.3 COMPRESSION/ENCODING TECHNIQUES CONSIDERED

We have implemented a C#/.NET program that automatically tests compressors, computing performance statistics (compression ratio, compression and decompression times). It is assumed that every compressor inherits from the standard .NET `System.IO.Stream` class, and operates on memory buffers. Below we presented a few methods we investigated. Some of them have been implemented and tested in the tool; some others have been rejected for reasons listed below.

bzip2. This is available in the #ziplib library [SharpZip].

gzip. This is available in the #ziplib library [SharpZip].

XMill. XMill [XMill] is written in C++. We created a .NET wrapper using Managed Extensions for C++; thus, it may be used by .NET applications as a .NET component. Unlike many other XML compressors (see below), a newer version is not file-oriented and has an interface for compressing buffers. The original implementation has been seriously improved – numerous memory bugs (memory leaks, uninitialized memory reads) have been corrected. XMill's implementation contains a number of "suspicious" choices – e.g., it uses exceptions but does not use `auto_ptr`'s, memory allocation result is not always checked, exception objects are dynamically created with `new`, etc.; thus, the implementation does not seem reliable enough for a commercial implementation, although for the purposes of the demonstrator it has been validated to be sufficiently stable (after corrections).

XMLppm. XMLppm [XMLppm] is written in C++. Unfortunately, the software is file-oriented (it can only process files); moreover, it is in beta phase, it has some known bugs, and, to make things worse, there are calls to `exit()` from within library functions. Correcting these issues could require substantial work.

XGrind. XGrind [XGrind] is written in C++. The implementation has the following limitations: it is Linux(only)-targeted; it is mainly designed for file compression; it uses global variables (which excludes it from use in a multithreaded environment! – like ours). Also, it has a limit that it only supports up to 256 distinct element and attribute names.

FastInfoset. This is Sun's approach to Binary Web Services [FastInfoset]. There seems to be no .NET implementation available.

BiM. BiM [BiM] is based on the MPEG-7 standard and has been designed and implemented by Siemens and Expway. This is a commercial product (although an evaluation version is available). BiM generally

requires schemas for best performance (this limits flexibility). Implementations are available in Java and C++.

Hessian Binary Web Service Protocol. Hessian [Hessian] is yet another approach to Binary WS. Implementations available in Java and .NET. Not analyzed.

As one could see, we finally ended up in just three compressors, with just one being XML-specific, and two being general-purpose compressors. A sad remark is that most research projects on XML compression result in implementations that are quite difficult for a "serious" use – either because of bugs, limited functionality or lack of support. This does not always mean that their implementations are "lame" – just they have been built only in order to prove a particular method (a "proof-of-concept" approach)), not to provide a useful component for larger software systems. Modifying them for our purposes looks like a time-consuming task, and the real quality of the code (in terms of correct XML processing) is unknown.

A11.4 COMPARISON METHODOLOGY

The performance evaluation of the three tools have been performed in the following steps:

1. A tool has been implemented that automatically measures performance of compressors. It compresses and then decompresses all files in a specified directory, then compares the result of the cycle (i.e., whether a file after decompression is the same as the original file – which does need to be true, e.g., as XMill may remove white spaces). For each file, compression or decompression is repeated either a specified number of times (which is more convenient for large files) or through some time (more convenient for small files).
2. A set of 50 XML files has been prepared (in fact, these were random files found on our hard drive). As the comparison mainly targets SOAP messages, white spaces (indentations etc.) have been removed. This choice may result in slightly different results than published elsewhere (which probably assume typical XML documents).

   The final file sizes were between 124 bytes and 4.187.786 bytes. The detailed sizes are given in the next section.

   Note that as we will have a set of typical files exchanged within the demonstrator, it will be easy to repeat the test.
3. For all three compressors, three factors have been computed: the compression ratio, the average compression time (in seconds), and the average decompression time (in seconds); these times were computed for all files being (de)compressed at least 10 times, assuming that (de)compression is repeated until it lasts at least 10 seconds.
4. Finally, some rough performance estimation has been computed for a tool run with every file (de)compressed exactly 10 times. Despite some common additional activities were performed (e.g., a file was opened and read), this is quite reasonable performance factor for all three mechanisms.

Note that we compared implementations, not algorithms – for example, #ziplib is implemented entirely in C#, while XMill is implemented in C++ (and available through a .NET wrapper, written in Managed C++). (It is difficult to predict which approach should yield better performance – we just note the difference.) Also, all algorithms were tested with default settings (if not stated otherwise).

A11.5 RESULTS

All test were performed on my Windows XP Professional box with Intel Pentium 4 2.8GHz and 1GB memory. The .NET Framework version was 2.0.50727.

The following table correlates file numbers with sizes (as it can be observed, file numbers grow with size):

File number	File size in bytes
1	124
2	232
3	423
4	745
5	795
6	823
7	836
8	852
9	857
10	896
11	899
12	946
13	987
14	1026
15	1586
16	1899
17	2017
18	2050
19	2498
20	3041
21	3046
22	3183
23	3833
24	4129
25	4760
26	4893
27	5800
28	5853
29	6037
30	6405
31	7825
32	8026
33	9060
34	10624
35	23034
36	39644
37	48239
38	79371
39	114250
40	242753
41	261772
42	341470
43	576028
44	814088
45	1054298
46	1172925
47	1479264
48	2385220
49	3978661
50	4187786

The following figures present measured performance factors: compression ratios (Figure A11.1), compression times (Figure A11.2) and decompression times (Figure A11.3). bzip2 results are removed for clarity.

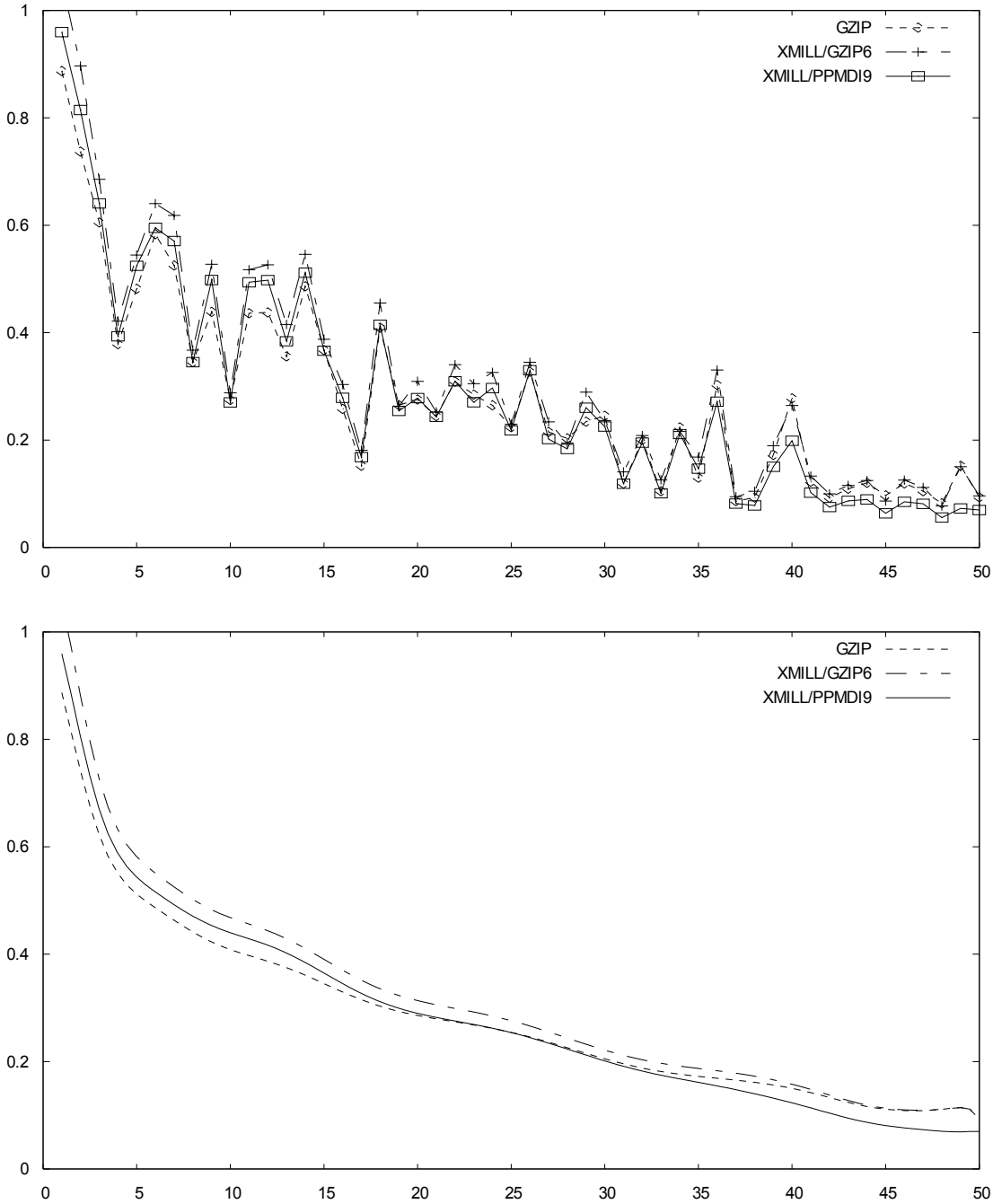

Figure A11.1: Compression Ratios (raw data & smoothed profiles).

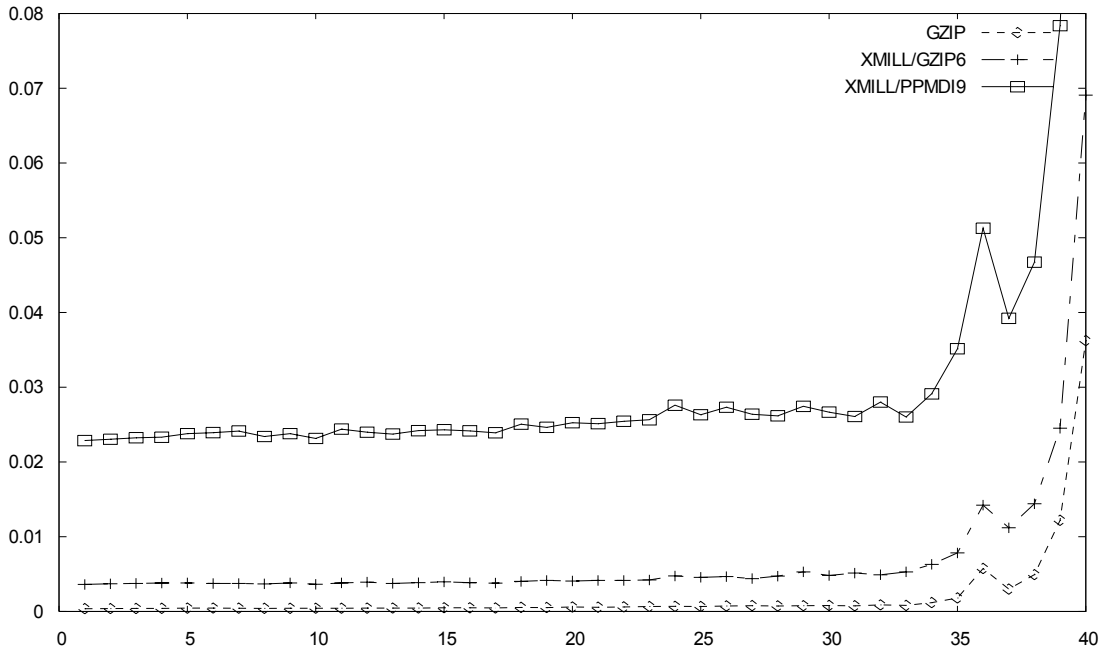

Figure A11.2: Compression Times.

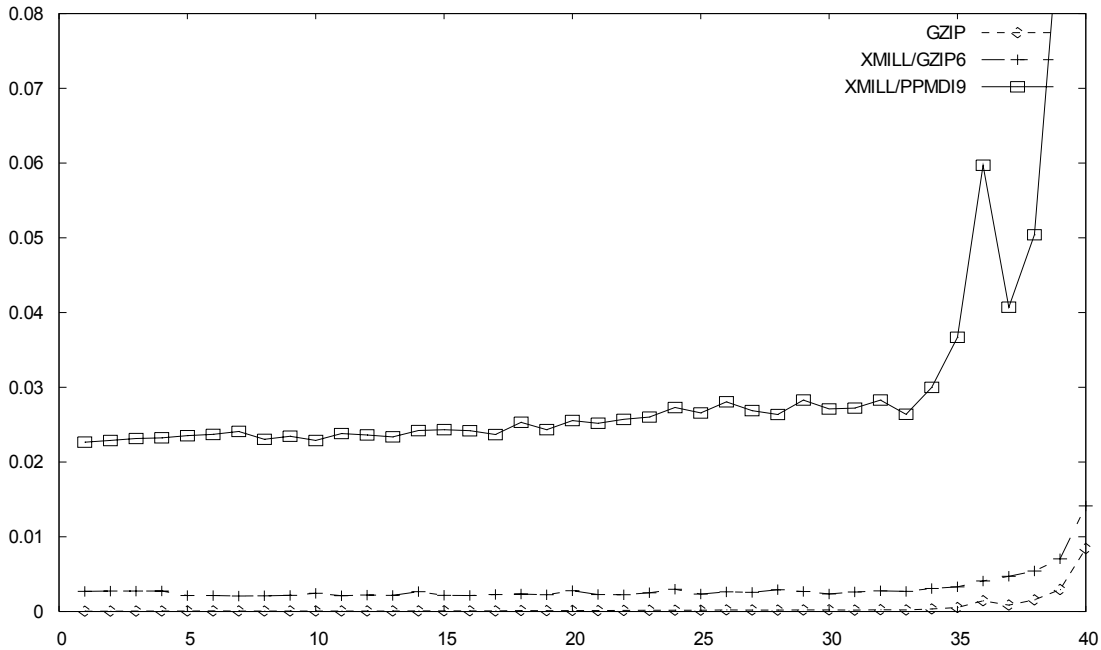

Figure A11.3: Decompression Times.

Rough performance comparison – a run with each file (de)compressed 10 times – are presented below.

Compressor	Compressor Options	Run Time [s]
bzip2		152.6
gzip		18.4
XMill/gzip level 6	-z -on	44.0
XMill/PPMDI level 9	-P -9 -on	142.5

We once again repeat that there were some (modest) common activities performed in each run, independently from a compressor type, so a real performance difference is bigger, and this is just a rough factor. Nonetheless, it is clearly visible that gzip is definitely fastest.

A11.6 CONCLUSIONS

Based on the study, one can draw the following conclusions:

- Even for smallest documents, all the compressors are able to further shrink the size of the file (with one exception – XMill/gzip for the smallest document). Thus, even if we decide to disable compression for documents below a given size, this size may be really small, maybe below 0.5 kB (the exact value remains to be verified).
- XMIll's results disappoint, at least for gzip/6, although we did not use additional, custom path expressions, which should improve its results considerably (as other publications report). It seems that with default settings, it does not work significantly better than gzip. On the other hand, applying such expressions to dynamically generated XML documents seems questionable.
- XMill's compression ratio improves with PPMDI/9, and becomes comparable to gzip, especially for large documents, for which it outperforms gzip.
- We do not confirm good results for bzip (reported elsewhere) – it seems to perform similarly to gzip (in terms of compression ratio) but is considerably slower.

Note, finally, that the test has been performed for a random set of XML documents. For messages exchanged within the demonstrator, these numbers may change.

A11.7 REFERENCES

[1] [BiM] http://www.expway.com/bim-technology.php.

[2] [DErlay] D. Erlay, "Adding a zip filter to web services", http://www.codeproject.com/cs/webservices/WebServiceZipFilter.asp?df=100&forumid=78022&exp=0&select=1322883.

[3] [FastInfoset] http://java.sun.com/developer/technicalArticles/xml/fastinfoset/.

[4] [Hessian] http://www.caucho.com/hessian/.

[5] [SharpZip] http://www.icsharpcode.net/OpenSource/SharpZipLib/.

[6] [XGrind] http://sourceforge.net/projects/xgrind/.

[7] [XMill] http://sourceforge.net/projects/xmill.

[8] [XMLppm] http://xmlppm.sourceforge.net/.

Annex B – TERMS OF REFERENCE

I. ORIGIN

A. Background

Current military C2, ISR and Combat Management systems are based on a federation of dedicated and heterogeneous systems, their operational integration deals with the following difficulties:

- Lack of operational interoperability due to the differences between information models used within the different (national) military Information Systems;
- Weak integration of Information Systems services for sharing of Situation Awareness from the strategic level to the tactical level; and
- Lack of global system management allowing the dynamic configuration of systems. This would enable these systems to deal with the diversity of operational scenarios going from intensive combat to peace keeping operations.

The military system evolution towards Network Enabled Capabilities (NEC) has to overcome these difficulties with the objective to achieve "Information Superiority", "Situation Awareness" and the capability to manage OODA (Observation, Orientation, Decision, Action) loops associated with the network integration of a "sensor-to-shooter" concept.

NEC will allow for seamless information exchange between all involved entities to achieve better and faster coordination of sensors, effectors and decision makers. The successful implementation of NEC systems requires the consistent integration of all the communication and information services implemented by the involved multi-platform Combat, Surveillance, Intelligence or Command systems.

Moving towards a Service Oriented Architecture (SOA) is a way to achieve the seamless information and service sharing required in a future NATO NEC. SOA will allow for available services and information to be published, discovered and shared in a flexible and dynamic manner over a communications network.

Security will be one of the greatest challenges in the development of a flexible and dynamic SOA. The need for a seamless information exchange requires changes in security policy as well as the introduction of end-to-end security solutions.

B. Justification (Relevance for NATO)

The realization of NATO NEC requires improved interoperability not only for C2 systems but also for ISR and Combat systems. According to the NATO NEC Feasibility Study (from NC3A), improvements in information exchange may be achieved by developing a Service Oriented Architecture (SOA). The development of a SOA in NATO will require interoperability between national systems at several levels from the data models describing the syntax and semantics of the information to the protocols for exchanging the information. The use of open standards will be essential in achieving this interoperability. Standards and specifications exist today that may be used to implement a SOA. The most common of these are the W3C/OASIS Web Services specifications, which are based on XML. An evaluation of the standards and specifications developed for XML, Web Services and XML Security, needs to be done in order to see whether they meet the military requirements defined for a future NATO NEC.

II. OBJECTIVES

This activity will include the following objectives:

A. The design, specification and development of a multi-national distributed demonstrator to be used to evaluate some of today's most common standards, specifications and technologies for implementing a Service Oriented Architecture. This involves:

   a) Development of a simple scenario describing interoperability of a joint and combined task force.

   b) Development of an XML data model for describing the syntax and semantics for the information to be exchanged between nations.

   c) Explore available COTS technology, especially the use of Web Services for service discovery, service subscription and service invocation.

   d) Use of end-to-end XML security solutions (including XML labeling and privileges). This involves the use of available XML security standards and specifications (as far as possible) for protecting the XML information exchanged using the Web Services technology.

B. The work on the demonstrator will be followed up by an evaluation phase, which will include:

   a) Evaluation of the solutions chosen and the technologies used in the SOA demonstrator.

   b) Evaluation of the civil specifications, standards and technology for use in military systems.

   c) Recommendations for changes and/or amendments to the specifications if required.

   d) Proposals for further work required for NATO and the nations in order to move towards a secure Service Oriented Architecture.

The team is expected to provide technical reports on the particular issues identified above.

The duration of the task group is two years.

III. RESOURCES

A. Membership

Members should preferably have been personally involved in the application of XML and related middleware technology to military CIS problems in recent years.

Lead nation: France and Norway will Co-Chair the group

B. National and/or NATO Resources Needed

At least 0.2 man years per scientist including travel funding for at least four meetings per year. Scientists should have access to national prototypes relevant to the specific goals above.

C. RTA Resources Needed

No NATO resources are identified.

IV. SECURITY CLASSIFICATION LEVEL

The initial, recommended security classification level for this activity is NATO UNCLASSIFIED. In the CWID 06 participation, all equipment will be classified NATO secret. All personnel involved must be at least cleared to NATO SECRET.

V. PARTICIPATION BY PARTNER NATIONS

Partner nations should not be invited to avoid complicating the discussion of national research results, prototypes, etc.

VI. LIAISON

Contact and collaboration with NC3A and related NATO working groups will be required.

REPORT DOCUMENTATION PAGE

1. Recipient's Reference	2. Originator's References	3. Further Reference	4. Security Classification of Document
	RTO-TR-IST-061 AC/323(IST-061)TP/52	ISBN 978-92-837-0069-2	UNCLASSIFIED/ UNLIMITED

5. Originator

Research and Technology Organisation
North Atlantic Treaty Organisation
BP 25, F-92201 Neuilly-sur-Seine Cedex, France

6. Title

Secure Service Oriented Architectures (SOA) Supporting NEC

7. Presented at/Sponsored by

This Report documents the findings of NATO/RTO Task Group IST-061.

8. Author(s)/Editor(s)	9. Date
Multiple	January 2009

10. Author's/Editor's Address	11. Pages
Multiple	226

12. Distribution Statement

There are no restrictions on the distribution of this document. Information about the availability of this and other RTO unclassified publications is given on the back cover.

13. Keywords/Descriptors

Combined scenario
Data management
Data processing
Demonstrator
Information systems
Integrated systems
Interoperability
Joint scenario
Network Enabled Capability (NEC)
Operational effectiveness
Operations research
Scenarios
Secure communication
Service Oriented Architecture (SOA)
Standardization
Tactical data communication

14. Abstract

This report describes the work of NATO/RTO IST-061 RTG-027 on "Secure Service Oriented Architectures (SOA) Supporting Network Enabled Capabilities". The primary focus of this work has been the creation of a demonstrator for tactical data communication in a combined and joint scenario. In this scenario, Service Oriented Architecture technologies have been used to exchange information such as sensor data among different units belonging to different nations.

The report explains how SOA technologies can be used to interoperate legacy systems in a way compliant with requirements identified by the NATO NEC Feasibility Study. Additionally, it provides a generic overview over the concepts of SOA and the technologies used in the demonstrator. The demonstrator itself is described from a conceptual point of view as well as on a more detailed level from the perspective of the individual participating organizations and companies.

Finally, the results gained through the work of the group are explained, again from a group level perspective and the point of view of each organization or company. From these results, some recommendations for future NATO activities are drawn.

NORTH ATLANTIC TREATY ORGANISATION

BP 25
F-92201 NEUILLY-SUR-SEINE CEDEX • FRANCE
Télécopie 0(1)55.61.22.99 • E-mail mailbox@rta.nato.int

RESEARCH AND TECHNOLOGY ORGANISATION

DIFFUSION DES PUBLICATIONS RTO NON CLASSIFIEES

Les publications de l'AGARD et de la RTO peuvent parfois être obtenues auprès des centres nationaux de distribution indiqués ci-dessous. Si vous souhaitez recevoir toutes les publications de la RTO, ou simplement celles qui concernent certains Panels, vous pouvez demander d'être inclus soit à titre personnel, soit au nom de votre organisation, sur la liste d'envoi.

Les publications de la RTO et de l'AGARD sont également en vente auprès des agences de vente indiquées ci-dessous.

Les demandes de documents RTO ou AGARD doivent comporter la dénomination « RTO » ou « AGARD » selon le cas, suivi du numéro de série. Des informations analogues, telles que le titre est la date de publication sont souhaitables.

Si vous souhaitez recevoir une notification électronique de la disponibilité des rapports de la RTO au fur et à mesure de leur publication, vous pouvez consulter notre site Web (www.rto.nato.int) et vous abonner à ce service.

CENTRES DE DIFFUSION NATIONAUX

ALLEMAGNE
Streitkräfteamt / Abteilung III
Fachinformationszentrum der Bundeswehr (FIZBw)
Gorch-Fock-Straße 7, D-53229 Bonn

BELGIQUE
Royal High Institute for Defence – KHID/IRSD/RHID
Management of Scientific & Technological Research for Defence, National RTO Coordinator
Royal Military Academy – Campus Renaissance
Renaissancelaan 30, 1000 Bruxelles

CANADA
DSIGRD2 – Bibliothécaire des ressources du savoir
R et D pour la défense Canada
Ministère de la Défense nationale
305, rue Rideau, 9e étage
Ottawa, Ontario K1A 0K2

DANEMARK
Danish Acquisition and Logistics Organization (DALO)
Lautrupbjerg 1-5, 2750 Ballerup

ESPAGNE
SDG TECEN / DGAM
C/ Arturo Soria 289
Madrid 28033

ETATS-UNIS
NASA Center for AeroSpace Information (CASI)
7115 Standard Drive
Hanover, MD 21076-1320

FRANCE
O.N.E.R.A. (ISP)
29, Avenue de la Division Leclerc
BP 72, 92322 Châtillon Cedex

GRECE (Correspondant)
Defence Industry & Research General Directorate, Research Directorate
Fakinos Base Camp, S.T.G. 1020
Holargos, Athens

HONGRIE
Department for Scientific Analysis
Institute of Military Technology
Ministry of Defence
P O Box 26
H-1525 Budapest

ITALIE
General Secretariat of Defence and National Armaments Directorate
5th Department – Technological Research
Via XX Settembre 123
00187 Roma

LUXEMBOURG
Voir Belgique

NORVEGE
Norwegian Defence Research Establishment
Attn: Biblioteket
P.O. Box 25
NO-2007 Kjeller

PAYS-BAS
Royal Netherlands Military Academy Library
P.O. Box 90.002
4800 PA Breda

POLOGNE
Centralny Ośrodek Naukowej Informacji Wojskowej
Al. Jerozolimskie 97
00-909 Warszawa

PORTUGAL
Estado Maior da Força Aérea
SDFA – Centro de Documentação
Alfragide
P-2720 Amadora

REPUBLIQUE TCHEQUE
LOM PRAHA s. p.
o. z. VTÚLaPVO
Mladoboleslavská 944
PO Box 18
197 21 Praha 9

ROUMANIE
Romanian National Distribution Centre
Armaments Department
9-11, Drumul Taberei Street
Sector 6
061353, Bucharest

ROYAUME-UNI
Dstl Knowledge and Information Services
Building 247
Porton Down
Salisbury SP4 0JQ

SLOVENIE
Ministry of Defence
Central Registry for EU and NATO
Vojkova 55
1000 Ljubljana

TURQUIE
Milli Savunma Bakanlığı (MSB)
ARGE ve Teknoloji Dairesi Başkanlığı
06650 Bakanliklar
Ankara

AGENCES DE VENTE

NASA Center for AeroSpace Information (CASI)
7115 Standard Drive
Hanover, MD 21076-1320
ETATS-UNIS

The British Library Document Supply Centre
Boston Spa, Wetherby
West Yorkshire LS23 7BQ
ROYAUME-UNI

Canada Institute for Scientific and Technical Information (CISTI)
National Research Council Acquisitions
Montreal Road, Building M-55
Ottawa K1A 0S2, CANADA

Les demandes de documents RTO ou AGARD doivent comporter la dénomination « RTO » ou « AGARD » selon le cas, suivie du numéro de série (par exemple AGARD-AG-315). Des informations analogues, telles que le titre et la date de publication sont souhaitables. Des références bibliographiques complètes ainsi que des résumés des publications RTO et AGARD figurent dans les journaux suivants :

Scientific and Technical Aerospace Reports (STAR)
STAR peut être consulté en ligne au localisateur de ressources uniformes (URL) suivant: http://www.sti.nasa.gov/Pubs/star/Star.html
STAR est édité par CASI dans le cadre du programme NASA d'information scientifique et technique (STI)
STI Program Office, MS 157A
NASA Langley Research Center
Hampton, Virginia 23681-0001
ETATS-UNIS

Government Reports Announcements & Index (GRA&I)
publié par le National Technical Information Service
Springfield
Virginia 2216
ETATS-UNIS
(accessible également en mode interactif dans la base de données bibliographiques en ligne du NTIS, et sur CD-ROM)

NORTH ATLANTIC TREATY ORGANISATION

BP 25
F-92201 NEUILLY-SUR-SEINE CEDEX • FRANCE
Télécopie 0(1)55.61.22.99 • E-mail mailbox@rta.nato.int

RESEARCH AND TECHNOLOGY ORGANISATION

DISTRIBUTION OF UNCLASSIFIED RTO PUBLICATIONS

AGARD & RTO publications are sometimes available from the National Distribution Centres listed below. If you wish to receive all RTO reports, or just those relating to one or more specific RTO Panels, they may be willing to include you (or your Organisation) in their distribution.

RTO and AGARD reports may also be purchased from the Sales Agencies listed below.

Requests for RTO or AGARD documents should include the word 'RTO' or 'AGARD', as appropriate, followed by the serial number. Collateral information such as title and publication date is desirable.

If you wish to receive electronic notification of RTO reports as they are published, please visit our website (www.rto.nato.int) from where you can register for this service.

NATIONAL DISTRIBUTION CENTRES

BELGIUM
Royal High Institute for Defence – KHID/IRSD/RHID
Management of Scientific & Technological Research for Defence, National RTO Coordinator
Royal Military Academy – Campus Renaissance
Renaissancelaan 30
1000 Brussels

CANADA
DRDKIM2 – Knowledge Resources Librarian
Defence R&D Canada
Department of National Defence
305 Rideau Street, 9th Floor
Ottawa, Ontario K1A 0K2

CZECH REPUBLIC
LOM PRAHA s. p.
o. z. VTÚLaPVO
Mladoboleslavská 944
PO Box 18
197 21 Praha 9

DENMARK
Danish Acquisition and Logistics Organization (DALO)
Lautrupbjerg 1-5
2750 Ballerup

FRANCE
O.N.E.R.A. (ISP)
29, Avenue de la Division Leclerc
BP 72, 92322 Châtillon Cedex

GERMANY
Streitkräfteamt / Abteilung III
Fachinformationszentrum der Bundeswehr (FIZBw)
Gorch-Fock-Straße 7
D-53229 Bonn

GREECE (Point of Contact)
Defence Industry & Research General Directorate
Research Directorate, Fakinos Base Camp
S.T.G. 1020
Holargos, Athens

HUNGARY
Department for Scientific Analysis
Institute of Military Technology
Ministry of Defence
P O Box 26
H-1525 Budapest

ITALY
General Secretariat of Defence and National Armaments Directorate
5th Department – Technological Research
Via XX Settembre 123
00187 Roma

LUXEMBOURG
See Belgium

NETHERLANDS
Royal Netherlands Military Academy Library
P.O. Box 90.002
4800 PA Breda

NORWAY
Norwegian Defence Research Establishment
Attn: Biblioteket
P.O. Box 25
NO-2007 Kjeller

POLAND
Centralny Ośrodek Naukowej Informacji Wojskowej
Al. Jerozolimskie 97
00-909 Warszawa

PORTUGAL
Estado Maior da Força Aérea
SDFA – Centro de Documentação
Alfragide
P-2720 Amadora

ROMANIA
Romanian National Distribution Centre
Armaments Department
9-11, Drumul Taberei Street
Sector 6
061353, Bucharest

SLOVENIA
Ministry of Defence
Central Registry for EU and NATO
Vojkova 55
1000 Ljubljana

SPAIN
SDG TECEN / DGAM
C/ Arturo Soria 289
Madrid 28033

TURKEY
Milli Savunma Bakanlığı (MSB)
ARGE ve Teknoloji Dairesi Başkanlığı
06650 Bakanliklar – Ankara

UNITED KINGDOM
Dstl Knowledge and Information Services
Building 247
Porton Down
Salisbury SP4 0JQ

UNITED STATES
NASA Center for AeroSpace Information (CASI)
7115 Standard Drive
Hanover, MD 21076-1320

SALES AGENCIES

NASA Center for AeroSpace Information (CASI)
7115 Standard Drive
Hanover, MD 21076-1320
UNITED STATES

The British Library Document Supply Centre
Boston Spa, Wetherby
West Yorkshire LS23 7BQ
UNITED KINGDOM

Canada Institute for Scientific and Technical Information (CISTI)
National Research Council Acquisitions
Montreal Road, Building M-55
Ottawa K1A 0S2, CANADA

Requests for RTO or AGARD documents should include the word 'RTO' or 'AGARD', as appropriate, followed by the serial number (for example AGARD-AG-315). Collateral information such as title and publication date is desirable. Full bibliographical references and abstracts of RTO and AGARD publications are given in the following journals:

Scientific and Technical Aerospace Reports (STAR)
STAR is available on-line at the following uniform resource locator: http://www.sti.nasa.gov/Pubs/star/Star.html
STAR is published by CASI for the NASA Scientific and Technical Information (STI) Program
STI Program Office, MS 157A
NASA Langley Research Center
Hampton, Virginia 23681-0001
UNITED STATES

Government Reports Announcements & Index (GRA&I)
published by the National Technical Information Service
Springfield
Virginia 2216
UNITED STATES
(also available online in the NTIS Bibliographic Database or on CD-ROM)

ISBN 978-92-837-0069-2