f

D O C U M E N T

document title/ titre du document

HETEROGENEOUS MISSION ACCESSIBILITY

TESTBED HMAT

TOOLBOX SOFTWARE REQUIREMENT DOCUMENT

(SECURITY LAYER)

prepared by/*préparé par* S. Puri

reference/*réference* HMAT-SRD-1200-INT-1.0
issue/*édition* 1
revision/*révision* 0
date of issue/*date d'édition* 21/11/2009
status/*état* Delivery
Document type/*type de document* Software Requirement
Distribution/*distribution*

a

ESA-ESRIN
Via G. Galilei, Frascati (Rome), Italy

A P P R O V A L

Title *titre*	HMAT Toolbox Software Requirement Document (security layer)	issue *issue*	1 revision *revision* 0
author *auteur*	S. Puri	date *date*	21/11/2009
approved by *approuvé by*	S. Gianfranceschi	date *date*	21/11/2009

C H A N G E L O G

reason for change /*raison du changement*	issue/*issue*	revision/*revision*	date/*date*
First version	1	0	21/11/2009

C H A N G E R E C O R D

Issue: 1 Revision: 0

reason for change/*raison du changement*	page(s)/*page(s)*	Paragraph(s)/*paragraph(s)*

TABLE OF CONTENTS

LIST OF FIGURES

1 INTRODUCTION

Earth Observation (EO) data users require accessing multiple data sources from different providers. It has been evaluated that more than 60% of the efforts of the Value-Adding Services is used for the EO data access.

The Heterogeneous Mission Accessibility - Interoperability (HMA-I) program started in 2005 in the framework of the Global Monitoring for Environment and Security (GMES) Preparatory activities with the purpose of defining the interoperability concept across the ground segments of the European and Canadian missions which will contribute to the GMES initial phase. These missions have developed or are in the process of developing EO satellite that can offer essential capacity to the GMES Space Component according to their own objective and now need to be adapted to these requirements.

In the framework of the HMA-I activities, the Agency has defined in collaboration with these organisations the ground segment architecture and interoperability standards for an across-missions harmonised data access that is general and independent from the set of missions supported and includes:

- Collection and service discovery
- Catalogue search
- Programming and Order
- Mission planning
- Data quality and product formats
- User management

The aim of this project is to support the user management interfaces for EO services specified in the [RD1], in particular regarding to the authorization processes to be used to pass identity information to Web services.

Concerning the technologies, the Heterogeneous Mission Accessibility - Testbed (HMA-T) project will be based upon existing specifications from W3C and OASIS; these technologies will be integrated in the SSE Toolbox, an open source tool developed in another ESA contract.

2 REFERENCES

2.1 Applicable documents

[AD1] European Cooperation for Space Standardization, Part 2: Document Requirements Definitions (DRDs), ECSS-E-40 Part 2B, 31/03/2005.

[AD2] HMA- T – 2 proposal HMA-T-2-MNG-PROP-136-08-SPPI, issue 1.0, 14/04/2008

[AD3] SSE Server and Services TOOLBOX Requirement Baseline, SAS-RB-510-INT Issue 1, Revision 3, 14/03/2008.

[AD4] Web Services Security: SOAP Message Security (WS-Security 2004), OASIS Standard Specification, 1 February 2006.

[AD5] Web Services Security: SAML Token Profile 1.1, Oasis Standard, 1 February 2006.

2.2 Reference documents

The following list of reference documents is for general guidance only and need not to be applied, but they should be given precedence over other documents covering similar topics.

[RD1] User Management Interface for Earth Observation Services, Open Geospatial Consortium Inc., 2008-04-03, ref. 07-118r1

[RD2] HMA-I User Management High Level Requirements, 08-02-07, ref. HMA-RS-SPB-EN-002.

3 TERMS, DEFINITIONS AND ABBREVIATED TERMS.

EO Earth Observation

ESA	European Space Agency
OASIS	Advancing Open Standards for the Information Society
GMES	Global Monitoring for Environment and Security
HMA	Heterogeneous Mission Accessibility – Interoperability
HTTP	Hyper Text Transfer Protocol
JDBC	Java DataBase Connectivity
PEP	Policy Enforcement Point. The location at which the authorisation or access rights to a service are enforced.
SAML	Security Assertion Markup Language
SOAP	Simple Object Access Protocol
SSE	Service Support Environment
WSDL	Web Services Description/Definition Language
XACML	eXtensible Access Control Markup Language
XML	eXtensible Markup Language

4 SOFTWARE OVERVIEW

4.1 Function and Purpose

4.1.1 The Toolbox

The Toolbox is a configurable application that helps the Service Provider to easily convert its legacy services into SOAP based services. It is usually used to integrate different kind of services and/or catalogues into the SSE infrastructure.

Different kinds of back-end communication systems are foreseen: File exchange, File Transfer Protocol, Hyper Text Transfer Protocol, API support, Script support, JDBC, SOAP and Email.

Furthermore the Toolbox provides an easy mechanism to convert the incoming XML files into other files (based on XML or on a proprietary format) or data structures suitable to be used for the communication with the back-end systems.

The Toolbox supports both synchronous and an asynchronous communication mechanism:

- Synchronous: in this solution, the SSE Portal is a SOAP client and the Toolbox acts only as a SOAP server. It only responds to SOAP requests. The XML response is sent back to the client during the same HTTP exchange.
- Asynchronous: in this solution, the service sends a SOAP message back to the client when a result is ready. Thus the Toolbox implements a SOAP server as well as a SOAP client. The Toolbox uses the WS-Addressing protocol.

The approach used to build a configurable application that will help the Service Provider to easily convert its service into a SOAP based service is based on XML scripting. In order to convert a legacy Service into a SOAP service suitable to be plugged into the SSE framework the Service Provider has to provide a number of XML scripting files (depending on the number of services to deploy on the SSE Portal, on the number of operations supported by such services and on the related communication mechanism). These files will describe the operations needed to complete a request coming from the client.

The configuration of the Toolbox, as well as the configuration and monitoring of the services, occur through the "Toolbox Administrator" Web application coming with the Toolbox.
The Toolbox has been developed in the MASS project and has been upgraded with new functionalities in the MASS-ENV project.
In the NSI project, the Toolbox has been upgraded in terms of easy utilization and faster services deployment. One of the main upgrade has been the development of the Toolbox Development Environment.
Thus, now the Toolbox is composed by two separate applications: the Toolbox RE (runtime environment) and the Toolbox DE (Development Environment).

The Toolbox RE is the web application developed in the previous projects and updated with additional functionalities such as:

- New look and feel: the look and feel has been aligned with the SSE portal.
- Monitoring center: the logging functionalities has been upgraded, now it is possible to trace the lifecycle of a request , display the incoming/outgoing messages and inspect the service execution by means of some graphical flow.
- New installation procedure: a graphical installer has been provided in order to ease the installation procedure.
- New deployment procedure: the deployment procedure has been simplified. It is now possible to download the SSE Schema from the portal and install them automatically in the Toolbox.
- New upgrade procedures: the Toolbox check for new versions of the tool and for new versions of the Schema and notify the user.

The Toolbox DE is an Eclipse[1] plugin, written in Java that extends some already existing functionality of the IDE itself and some other exported by the Web Tools Platform (WTP)[2] plugin. It can be used to write an XML script to be deployed into the Toolbox RE. It provides

[1] http://www.eclipse.org/

[2] http://www.eclipse.org/webtools/

built-in features like graphical vs textual visualization, tag suggestion, online validation. It supports the following functionalities:

- Toolbox Script editor: the plug-in allows building a Toolbox script
- Launcher: the plug-in allows executing a Toolbox script file.
- Debug: the plug-in allows managing the breakpoint for a Toolbox script file in the default editor Script.
- Check if more recent versions are available on the Toolbox server.

In the context of the HMA project the Toolbox will be further upgraded and will provide the support for authorization as specified in [RD1]. In particular the Toolbox will be extended by introducing a Web security stack that will be used to provide Web security access to the published services, moreover the user will be allowed to specify enterprise policies that will constraint the access to a given service.
Accordingly to [RD1] the security model will be based on WS-Security SAML (Security Assertion Markup Language) token profile [AD5].

The tools and technologies that has been selected to add WS-Security functionalities to the Toolbox are the Apache Axis2 open source together with the Rampart module, in particular the OpenSAML library will be used as implementation of the SAML specification. Concerning enterprise policies specification and enforcement, the XACML library has been selected. In the next sections a brief overview of these tools and technologies is provided.

4.1.2 Axis2

Apache Axis is a Java-based implementation of both the client and server sides of the SOAP ("Simple Object Access Protocol") submission to W3C. Apache Axis2, the third generation Web services engine is more efficient, more modular and more XML-oriented than its predecessor Apache Axis. Axis2 provides an object model and a modular architecture that allows adding functionality and support for new Web services-related specifications and recommendations.

Axis2 enables to perform the following tasks:

- Send SOAP messages
- Receive and process SOAP messages
- Create a Web service out of a plain Java class
- Create implementation classes for both the server and client using WSDL
- Retrieve the WSDL for a service
- Send and receive SOAP messages with attachments
- Create or utilize a REST-based Web service
- Create or utilize services that take advantage of the WS-Security, WS-ReliableMessaging, WS-Addressing, WS-Coordination, and WS-Atomic Transaction recommendations
- Use Axis2's modular structure for extensions

4.1.3 Rampart

Apache Rampart/Java is the tool kit that provides the implementation of security related WS-* specifications for Apache Axis2. The latest version 1.4 implements the following specifications:

- WS - Security 1.0
- WS - Security 1.1
- WS - Secure Conversation - February 2005
- WS - Security Policy - 1.1 - July 2005
- WS - Security Policy - 1.2
- WS - Trust - February 2005
- WS - Trust - WS-SX spec - EXPERIMENTAL

The Rampart configuration model uses WS-SecurityPolicy together with a set of Apache Rampart specific policy assertions, this is because the policy assertions provided in WS-SecurityPolicy specification are not sufficient to meet all the configuration requirements of Rampart.

Rampart v. 1.4 makes use of OpenSAML v.1 library to manage SAML v.1 token assertion stored into WS-Security SOAP messages.

4.1.4 XACML

XACML (eXtensible Access Control Markup Language) is an XML-based language for access control that has been standardized in OASIS. XACML describes both an access control policy language and a request/response language. The policy language is used to express access control policies (who can do what/when). The request/response language expresses queries about whether a particular access should be allowed (requests) and describes answers to those queries (responses).

4.2 Environmental considerations

4.2.1 Development Environment

4.2.1.1 Hardware Environment

The hardware environment that will be used for developing purposes will comprehend PC running both Windows and Linux (Fedora Core 9) OS. All implied hosts would be connected through a LAN that will let local communications. If it will be necessary to connect to remote hosts, a connection to Internet will be made available.

4.2.1.2 Operating Environment

As stated above, development machine will run both Windows and Fedora Core 9 OS, letting an extensive test on both systems. In order to properly running Toolbox, a Java SDK 5 and a Tomcat installation are needed; moreover the Toolbox security stack requires Axis2 with Rampart module deployed Tomcat.

4.3 Relation to other systems

The Toolbox will enable Service Provider to create services that can be connected to the HMA infrastructure and which access is regulated through WS-Security policies enforcement. The product described by this document is finalized of providing WS-Security at Ground Segment level, enabling existing GS to wrap and connect their own catalogues to the HMA infrastructure.

Next figure, originally taken from [RD2], illustrates how the parts that will be developed during this project will fit into the HMA infrastructure.

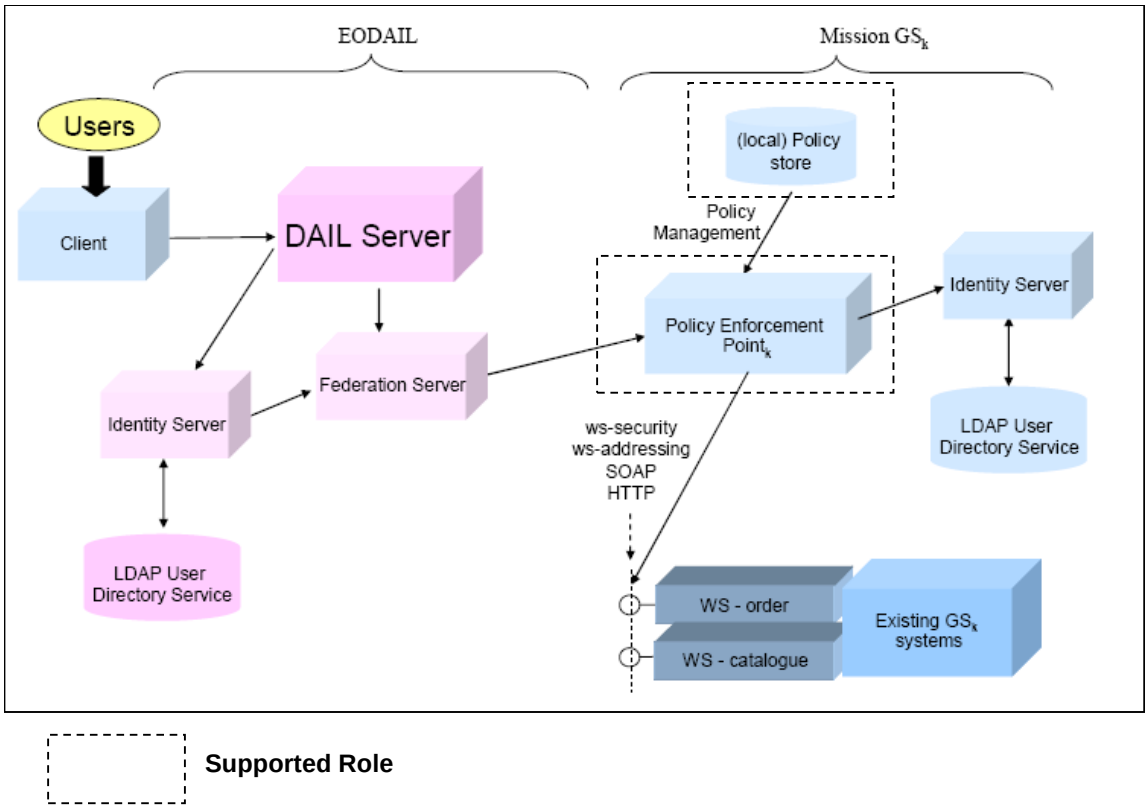

Figure 4-1 Infrastructure high-level diagram

4.4 Logical model description

Toolbox is a web application built on top of Apache Tomcat. It exposes web pages for administering it and provides the core logic for implementing new SOAP services.
The core logic is composed by two separate pieces:

- SOAP layer
- Application Layer.

The SOAP Layer is essentially responsible of handling all aspects of such communication protocol, handling also the process of validating the incoming messages.
The Application Layer is responsible of managing services and their instances, providing to them some supporting services like synchronous/asynchronous execution mode, FTP server.

Within this project an additional layer will be put between the SOAP layer and the Application Layer to provide Web Service security; accordingly to [RD1] the security model will be based on WS-Security SAML token profile which usage in the Toolbox environment can be summarized through Figure 4-2.
Figure 4-2 depicts the following scenario: the client first obtains a SAML token through an identity provider (not covered here) and then includes the token in the SOAP header of the service request. The Toolbox accepts the request and check it with the Policy Enforcement Point (PEP) layer, then the PEP decides based on the content of the message body, the contents of the message header (including the authenticating token) and the context (i.e. applicable policies) whether to accept or to refuse the service request.

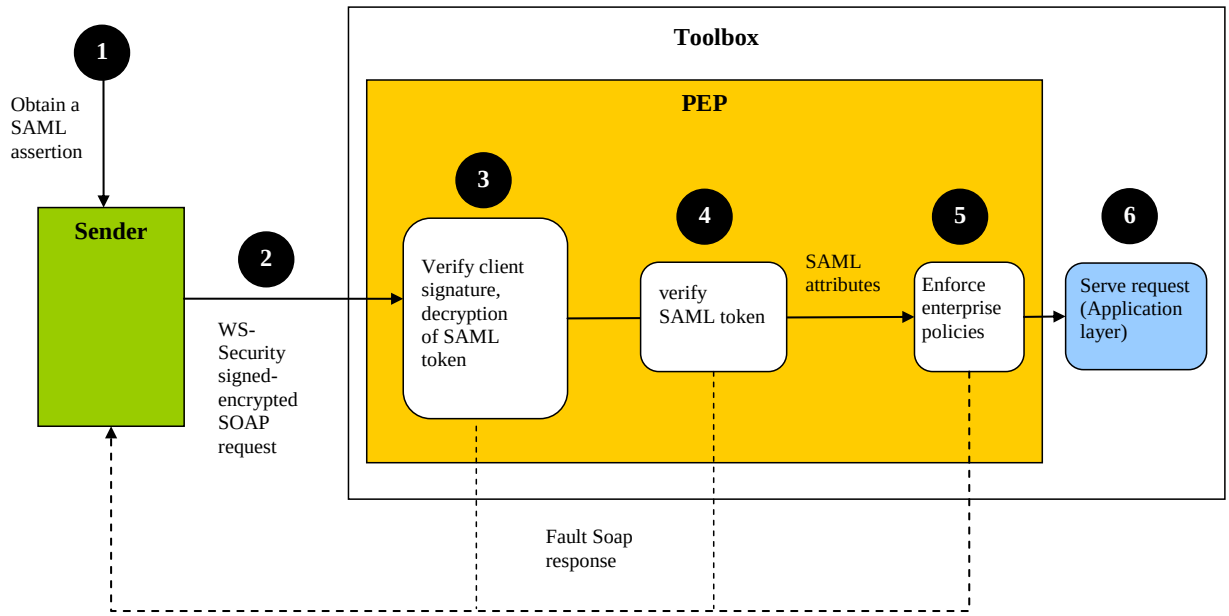

Figure 4-2 Toolbox security layer with SAML usage scenario.

5 SOFTWARE REQUIREMENTS

5.1 Overview

This chapter provides the complete list of software requirements applicable to the HMA-T project.

Each software requirement listed in this document is described by a table whose template is provided below:

HMAT-RB-XXX-YYY	Requirement Criticality	Verification Method
Requirement text		
Source:		

HMAT-RB-XXX-YYY	Requirement Criticality	Verification Method
Note:		

Below a description of all fields that builds the table:

5.1.1 HMAT-RB-XXX-YYY

This field identifies uniquely the requisite. It is composed by a fixed part (i.e. HMAT-RB) and by two variable parts, whose values can be the following:

- XXX: this field identifies the applicability scope of the requisite. Permitted values are:
  - FUN: value identifying functional requirements.
  - PER: value identifying performance requirements.
  - INT: value identifying interface requirements.
  - OPE: value identifying operational requirements.
  - RES: value identifying resource requirements.
  - DES: value identifying design requirements.
  - SEC: value identifying security and privacy requirements.
  - POR: value identifying portability requirements.
  - QLT: value identifying software quality requirements.
  - REL: value identifying software reliability requirements.
  - MAI: value identifying software maintainability requirements.
  - SAF: value identifying software safety requirements.
  - SCD: value identifying software configuration and delivery requirements.
  - DDB: value identifying data definition and database requirements.
  - HUM: value identifying human factors related requirements.
  - ADP: value identifying adaptation and installation requirements.
  - OTH: value identifying other requirements
  - VAL: value identifying validation requirements.

- YYY: this field is a numerical sequential value that uniquely identifies the requisite inside a scope. It is composed by at least by 3 digits.

5.1.2 Requirement Criticality

This field describes the importance of the system compliance with the requirement. The following values are used in this document:

- **Essential**: the requirement implementation is compulsory.
- **Desirable**: the requirement implementation is optional.

5.1.3 Verification Method

This field describes the method to be used for verifying the requirement compliance. The following values can be used:

- **Analysis [A]**

  This verification method implies use of analytical techniques (such as system engineering analysis, statistics, mathematical modelling, simulations) and shall be used to verify such requirements.
- **Review of Design [D]**

  This verification method may be used when approved Design reports, technical descriptions, engineering drawings unambiguously show that the requirement is met.
- **Inspection [I]**

  Verification by inspection is only done when testing is insufficient or inappropriate.

  This method of verification is for those requirements that are normally performed by some form of visual inspection. This would include workmanship, labelling, envelope requirements etc.
- **Demonstration [M]**

  This verification method may be used when actual conduct can verify achievement of requirements such as service and access, transportability,

human engineering features and processes hardware. A requirement which is of an operational or functional nature and is not quantified by a specific measurable parameter may be verified by demonstration.

- **Similarity [S]**
  This verification method may be used when there is proof that the item is similar or identical in design and manufacturing processes to another previously qualified to equivalent or more stringent criterion.
- **Test [T]**
  A requirement may be verified by test alone if the form of the specification is such that the requirement can be directly measured.

5.1.4 Requirement Text

This field contains the text of the requirement.

5.1.5 Note

Optional field which intends to provide additional information, justification, or explanation of the requirement. A note is for information only and is not considered part of the requirement.

5.1.6 Source

A source is an applicable document the requirement derives from. Wherever possible the original requirement number of the applicable document is specified

5.2 Functional Requirements

HMAT-RB-FUN-010	Requirement Criticality	Verification Method
Given a service published on the Toolbox, it shall be possible to specify that only authenticated clients are allowed to request the service.		
Source:		

HMAT-RB-FUN-010	Requirement Criticality	Verification Method
Note:		

HMAT-RB-FUN-020	Requirement Criticality	Verification Method
Given a service published on the Toolbox, it shall be possible to create/modify the policies that need to be satisfied by an authenticated client in order to authorize service requests.		
Source:		
Note:		

HMAT-RB-FUN-030	Requirement Criticality	Verification Method
The service request that can be wrapped by the Toolbox security stack shall be any of the operation defined in the catalogue (OGC 06-131), ordering (OGC 06-141) or programming (OGC 07-018) specifications; synchronous as well as asynchronous request shall be supported.		
Source:		
Note:		

5.3 Interface requirements.

HMAT-RB-PER-010	Requirement Criticality	Verification Method
Only SOAP messaging with the document/literal style shall be supported, the messages shall be conform to SOAP 1.2 and the message payload has to be in the body of the SOAP envelope.		
Source: [RD1] section 6.2		
Note:		

HMAT-RB-PER-020	Requirement Criticality	Verification Method
Security policies for a service shall be specified accordingly to the WS-Security Policy Language.		

HMAT-RB-PER-020	Requirement Criticality	Verification Method
Source:		
Note:		

HMAT-RB-INT-030	Requirement Criticality	Verification Method
Authentication shall be enforced by using Web Service Security-SAML v.1 Token Profile over HTTPS.		
Source: [RD1] section 6.4		
Note:		

HMAT-RB-INT-040	Requirement Criticality	Verification Method
A timestamp shall be included in the security header of a SOAP request.		
Source: [RD1] section 6.4.6.2.		
Note:		

HMAT-RB-INT-050	Requirement Criticality	Verification Method
Given a service with authenticated user required, any service request sent from the client to the Toolbox must carry the SAML token. Response from the Toolbox to the client need not to carry the SAML token.		
Source: [RD1] section 6.4.4.1.		
Note:		

HMAT-RB-INT-060	Requirement Criticality	Verification Method
The Toolbox security layer shall support asymmetric binding assertion with X509 certificates.		

HMAT-RB-INT-060	Requirement Criticality	Verification Method
Source:		
Note:		

HMAT-RB-INT-070	Requirement Criticality	Verification Method
The SAML token shall be encrypted with the public key related to the endpoint.		
Source: [RD1] section 6.4.4.1.		
Note: The SAML token shall be signed by the client with its private key.		

HMAT-RB-INT-080	Requirement Criticality	Verification Method
The body of the SOAP message should be signed with the client private key.		
Source: [RD1] section 6.4.4.1.		
Note:		

HMAT-RB-INT-090	Requirement Criticality	Verification Method
The order of Signature and Encryption operations to be applied on the SAML token shall be encryption first and then signature.		
Source: [RD1] section 6.4.4.1.		
Note:		

HMAT-RB-INT-100	Requirement Criticality	Verification Method
Regarding the SAML token, the encryption algorithm shall be the AES-128.		
Source: [RD1] section 6.4.1.		
Note:		

HMAT-RB-INT-110	Requirement Criticality	Verification Method
Regarding the SAML token, the security hash SHA1 digital signature message digest algorithm shall be used.		
Source: [RD1] section 6.4.2.		
Note:		

HMAT-RB-INT-120	Requirement Criticality	Verification Method
The following attributes in the SAML token shall be supported: • hmaID (unhambiguos HMA identity) • c (country of origin) • o (organization) • hmaProjectName (name of the projects with which the user is affiliated) • hmaAccount (The HMA account number) • hmaServiceName (associated services)		
Source: [RD1] section 6.4.5.		
Note:		

HMAT-RB-INT-130	Requirement Criticality	Verification Method
In case of an authorization error, a SOAP fault shall be returned		
Source: [RD1] section 7.2.3		
Note:		

HMAT-RB-PER-140	Requirement Criticality	Verification Method
Authorization policies for a service shall be specified accordingly to XACML.		
Source:		

HMAT-RB-PER-140	Requirement Criticality	Verification Method
Note:		

5.4 Operational requirements.

HMAT-RB-OPE-010	Requirement Criticality	Verification Method
Authorization policies shall be expressed in dedicated xml files.		
Source:		
Note:		

HMAT-RB-OPE-020	Requirement Criticality	Verification Method
Each Web service exposed shall express its requirements and capabilities through policies embedded in the service's WSDL description.		
Source:		
Note:		

HMAT-RB-OPE-030	Requirement Criticality	Verification Method
The creation of the WSDL with WS-Security policies shall automatically perform by the Toolbox.		
Source:		
Note:		

5.5 Design requirements

HMAT-RB- DES -010	Requirement Criticality	Verification Method
AXIS2 with Rampart module deployed on Tomcat shall be used for the SOAP layer.		
Source:		

HMAT-RB- DES -010	Requirement Criticality	Verification Method
Note:		

HMAT-RB- DES -020	Requirement Criticality	Verification Method
The integration and usage of the Toolbox security stack shall be completely transparent to Toolbox users.		
Source:		
Note:		

5.6 Portability requirements

HMAT-RB-POR-010	Requirement Criticality	Verification Method
The security stack to be developed for the Toolbox shall be implemented in Java language.		
Source:		
Note:		

5.7 Software configuration and delivery requirements

HMAT-RB-SCD-010	Requirement Criticality	Verification Method
Toolbox with security layer shall be delivered under GPL v2.0 license		
Source:		
Note:		

HMAT-RB-SCD-020	Requirement Criticality	Verification Method

HMAT-RB-SCD-020	Requirement Criticality	Verification Method
Toolbox with security layer shall be delivered via: • CD or DVD ROM • Sourceforge		
Source:		
Note:		

HMAT-RB-SCD-030	Requirement Criticality	Verification Method
The delivered CD or DVD shall also contain all COTS.		
Source:		
Note:		

5.8 Other requirements

HMAT-RB-OTH-010	Requirement Criticality	Verification Method
Toolbox security layer shall provide documentation that covers configuration tasks.		
Source:		
Note:		