HMA-T/G-POD Gateway
System Architecture
Document
Terradue
<table>
<thead>
<tr>
<th>Short Title</th>
<th>HMA-T/G-POD Gateway System Architecture Document</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prepared by</td>
<td>Terradue, RAL</td>
</tr>
<tr>
<td>Approved by</td>
<td></td>
</tr>
<tr>
<td>Reference</td>
<td>T2-ESA-HMAT-DD-08-023</td>
</tr>
<tr>
<td>External Reference</td>
<td>HMAT-ADD-0003-T2</td>
</tr>
<tr>
<td>Issue</td>
<td>0</td>
</tr>
<tr>
<td>Revision</td>
<td>1</td>
</tr>
<tr>
<td>Date of issue</td>
<td>N/A</td>
</tr>
<tr>
<td>Status</td>
<td>DRAFT</td>
</tr>
<tr>
<td>Document type</td>
<td>System Architecture Document</td>
</tr>
<tr>
<td>Distribution</td>
<td>ESA, SPACEBEL</td>
</tr>
<tr>
<td>Category</td>
<td></td>
</tr>
<tr>
<td>Keywords</td>
<td>HMA-T, GPOD</td>
</tr>
<tr>
<td>Comments</td>
<td></td>
</tr>
</tbody>
</table>
## Approval

<table>
<thead>
<tr>
<th>title</th>
<th>HMA-T/G-POD Gateway System Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>issue</td>
<td>1</td>
</tr>
<tr>
<td>revision</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>author</th>
<th>Terradue, RAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>date</td>
<td>2008-09-30</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>approved by</th>
<th>Fabrice Brito</th>
</tr>
</thead>
<tbody>
<tr>
<td>date</td>
<td>N/A</td>
</tr>
</tbody>
</table>

## Change Log

<table>
<thead>
<tr>
<th>reason for change</th>
<th>issue</th>
<th>revision</th>
<th>date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial version</td>
<td>1</td>
<td>0</td>
<td>2008-09-03</td>
</tr>
</tbody>
</table>
Table of Contents

1 Introduction ............................................................................................................. 1
  1.1 Purpose of the document .................................................................................. 1
  1.2 Objectives ......................................................................................................... 1
  1.3 Content Overview ............................................................................................ 1
2 Applicable and Reference Documents .................................................................... 2
  2.1 Applicable Documents ..................................................................................... 2
  2.2 Reference Documents ....................................................................................... 2
3 Terms, Definitions and Abbreviated Terms ............................................................. 3
  3.1 Definition of Terms .......................................................................................... 3
  3.2 Abbreviated Terms .......................................................................................... 4
4 System Design Overview ......................................................................................... 6
  4.1 Background ....................................................................................................... 6
    4.1.1 HMA and HMA-T ....................................................................................... 6
    4.1.2 User Management OGC Specification ....................................................... 6
    4.1.3 gridify - Service Application Environment .............................................. 7
    4.1.4 ESA Grid Processing on-Demand (G-POD) .............................................. 8
  4.2 Architecture Overview ...................................................................................... 11
    4.2.1 Hardware Architecture .............................................................................. 11
    4.2.2 System Security ......................................................................................... 11
    4.2.3 System Architecture .................................................................................. 12
  4.3 Interface Context ............................................................................................... 13
5 System Software Design ......................................................................................... 14
  5.1 Overall Architecture ......................................................................................... 14
    5.1.1 Concept ....................................................................................................... 14
    5.1.2 System Interfaces ....................................................................................... 15
      5.1.2.1 Internal Interfaces .............................................................................. 15
      5.1.2.2 External Interfaces ............................................................................. 15
  5.2 System Software Components Description ....................................................... 16
    5.2.1 HGG Service Invoker ................................................................................ 17
    5.2.2 HGG Access Point .................................................................................... 20
      5.2.2.1 HGG Access Point Web Service description Error! Bookmark not defined.
      5.2.2.2 HGG Access Point Web service.... Error! Bookmark not defined.
    5.2.3 HGG Identity Handler ................................................................................. 22
5.2.4 HGG Dynamic User Registry ................................................................. 23
5.2.5 HGG G-POD Web Service Client ......................................................... 23
6 REQUIREMENTS TO DESIGN COMPONENTS TRACEABILITY .................. 28
List of Tables
Table 1: HMA-TGP-POD Internal Interfaces ........................................ 15
Table 2: HMA-TGP-POD External Interfaces ........................................ 16

List of Figures
Figure 1: G-POD Web Portal based on gridify .................................... 9
Figure 2: G-POD Job Submission Web Service flow ............................... 10
Figure 3: Functional Architecture ....................................................... 11
Figure 4: HMA-TG-POD Gateway Architecture .................................... 12
Figure 5 HMA-T G-POD Detailed Architecture ..................................... 14
1 Introduction

1.1 Purpose of the document

The purpose of this document is to detail the software architectural design and software detailed design of the HMA-T/G-POD Gateway System hereafter called HGG.

1.2 Objectives

This document is part of the Design Definition File (DDF) and is the main input of the Critical Design Review (CDR).

1.3 Content Overview

Section 1
This section.

Section 2
This section gives the list of all documents applicable to this Software Design Document and the list of the project document used as reference.

Section 3
This section list all abbreviation and terms used throughout the document.

Section 4
This section gives a general description of the project.

Section 5
This section gives a general description of the architecture of the system and a detailed description of the software and its components.

Section 6
This section gives the traceability matrix between the Software Requirement and the software Element of the HGG.
2 Applicable and Reference Documents

2.1 Applicable Documents

<table>
<thead>
<tr>
<th>Document ID</th>
<th>Description</th>
<th>Reference</th>
<th>Date</th>
</tr>
</thead>
</table>

2.2 Reference Documents

<table>
<thead>
<tr>
<th>Document ID</th>
<th>Description</th>
<th>Reference</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>[R1]</td>
<td>SoW For HMA Testbed HMA-T Phase 2</td>
<td>SPB-HMA-T-SOW-002</td>
<td>31-01-2008</td>
</tr>
<tr>
<td>[R2]</td>
<td>OGC 07-118 User Management Services For Earth Observation</td>
<td>07-118r1 User Management Interfaces for Earth Observation 0.0.2</td>
<td>23-04-2008</td>
</tr>
</tbody>
</table>
3 Terms, Definitions and Abbreviated Terms

3.1 Definition of Terms

This document and its appendices use the terms:

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authentication</td>
<td>To confirm a system entity’s asserted principal identity with a specified, or understood, level of confidence.</td>
</tr>
<tr>
<td>Circle of trust</td>
<td>A federation of service providers and identity providers within which service providers accept the authentication asserted by the identity provider.</td>
</tr>
<tr>
<td>Client</td>
<td>Software component that can invoke an operation from a server</td>
</tr>
<tr>
<td>Interface</td>
<td>Named set of operations that characterise the behaviour of an entity</td>
</tr>
<tr>
<td>Identifier</td>
<td>A character string that may be composed of numbers and characters that is exchanged between the client and the server with respect to a specific identity of a resource</td>
</tr>
<tr>
<td>Identity Provider</td>
<td>A kind of service provider that creates, maintains, and manages identity information for principals and provides principal authentication to other service providers within a federation, such as with web browser profiles.</td>
</tr>
<tr>
<td>Interface</td>
<td>Named set of operations that characterise the behaviour of an entity [ISO 19119]</td>
</tr>
<tr>
<td>Operation</td>
<td>Specification of a transformation or query that an object may be called to execute</td>
</tr>
<tr>
<td>Parameter</td>
<td>Variable whose name and value are included in an operation request or response</td>
</tr>
<tr>
<td>Request</td>
<td>Invocation of an operation by a client</td>
</tr>
<tr>
<td>Response</td>
<td>Result of an operation, returned from a server to a client</td>
</tr>
<tr>
<td>The Prime Contractor</td>
<td>Spacebel is the ESA appointed Prime Contractor for the project Heterogeneous Missions Accessibility Testbed “HMA Testbed” (HMA-T). In the following Spacebel is referred to as the “Prime Contractor”.</td>
</tr>
<tr>
<td>The Agency</td>
<td>To indicate the European Space Agency (ESA)</td>
</tr>
<tr>
<td>Service</td>
<td>Capability which a service provider entity makes available to a service user entity at the interface between those entities</td>
</tr>
<tr>
<td>Service Interface</td>
<td>Shared boundary between an automated system or human being and another automated system or human being</td>
</tr>
<tr>
<td>Service Provider</td>
<td>A role donned by a system entity where the system entity provides services to principals or other system entities.</td>
</tr>
<tr>
<td>Shall</td>
<td>To indicate a firm requirement</td>
</tr>
<tr>
<td>Should</td>
<td>To indicate a highly desirable requirement</td>
</tr>
<tr>
<td>TBD</td>
<td>To be defined by the Agency, or in agreement with the Agency</td>
</tr>
<tr>
<td>TBC</td>
<td>To be conformed by the Agency, or in agreement with the Agency</td>
</tr>
<tr>
<td>Transfer protocol</td>
<td>Common set of rules for defining interactions between distributed</td>
</tr>
</tbody>
</table>
### 3.2 Abbreviated Terms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>C#</td>
<td>C# (pronounced C Sharp) is a multi-paradigm programming language that encompasses functional, imperative, generic, object-oriented (class-based), and component-oriented programming disciplines. It was developed by Microsoft as part of the .NET initiative and later approved as a standard by ECMA</td>
</tr>
<tr>
<td>CDR</td>
<td>Critical Design Review</td>
</tr>
<tr>
<td>CE</td>
<td>Computing Element</td>
</tr>
<tr>
<td>CERN</td>
<td>Conseil Européen pour la Recherche Nucléaire</td>
</tr>
<tr>
<td>CITE</td>
<td>OGC Compliance &amp; Interoperability Testing &amp; Evaluation</td>
</tr>
<tr>
<td>COTS</td>
<td>Commercial off-the-shelf</td>
</tr>
<tr>
<td>DDF</td>
<td>Design Definition File</td>
</tr>
<tr>
<td>EC</td>
<td>European Commission</td>
</tr>
<tr>
<td>EO</td>
<td>Earth Observation</td>
</tr>
<tr>
<td>EO-DAIL</td>
<td>Earth Observation Data Access Integration Layer</td>
</tr>
<tr>
<td>ESA</td>
<td>European Space Agency</td>
</tr>
<tr>
<td>ESRIN</td>
<td>European Space Research Institute</td>
</tr>
<tr>
<td>G-POD</td>
<td>Grid Processing on-Demand</td>
</tr>
<tr>
<td>HGG</td>
<td>HMA-T/G-POD Gateway</td>
</tr>
<tr>
<td>HGG-AP</td>
<td>HGG Access Point</td>
</tr>
<tr>
<td>HGG-DUR</td>
<td>HGG Dynamic User Registry</td>
</tr>
<tr>
<td>HGG-IH</td>
<td>HGG Identity Handler</td>
</tr>
<tr>
<td>HGG-GWSC</td>
<td>HGG G-POD Web Service Client</td>
</tr>
<tr>
<td>HGG-SI</td>
<td>HGG Service Invoker</td>
</tr>
<tr>
<td>HMA</td>
<td>Heterogeneous Mission Accessibility</td>
</tr>
<tr>
<td>HW</td>
<td>Hardware</td>
</tr>
<tr>
<td>IdP</td>
<td>Identity Provider</td>
</tr>
<tr>
<td>ICD</td>
<td>Interface Control Document</td>
</tr>
<tr>
<td>ISO</td>
<td>International Organization for Standardization</td>
</tr>
<tr>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>OGC</td>
<td>Open Geospatial Consortium</td>
</tr>
<tr>
<td>OS</td>
<td>Operating System</td>
</tr>
<tr>
<td>PAC</td>
<td>Processing and Archiving Centre</td>
</tr>
<tr>
<td>PEP</td>
<td>Policy Enforcement Point</td>
</tr>
<tr>
<td>RB</td>
<td>Requirements Baseline</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>SAML</td>
<td>Security Assertion Markup Language</td>
</tr>
<tr>
<td>SOAP</td>
<td>Simple Object Access Protocol</td>
</tr>
<tr>
<td>SP</td>
<td>Service Provider</td>
</tr>
<tr>
<td>SW</td>
<td>Software</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
<tr>
<td>WSDL</td>
<td>Web Service Definition Language</td>
</tr>
</tbody>
</table>
4 System Design Overview

4.1 Background

4.1.1 HMA and HMA-T

HMA implements an open and distributed solution to support interoperability among partners ground segments. This interaction is still strongly focused on catalogue services integration among heterogeneous infrastructures and their interactions. To achieve the desired interoperability goals, HMA addresses the connection with high value processing services, integration of user and added value service companies. Apart from simple geographic services like conversions of data, data extraction, generalization or resampling, HMA still needs to pave the ground to a true interoperability with processing infrastructures.

To improve on the points raised earlier in phase 1, the Agency initiated the HMA-T project to address the conformance testing and to provide a testbed. This testbed will allow the consolidation and evolution of the HMA standards and their conformance testing. There are three main objectives defined in the second phase of this project:

1. The first deals with the development of implementations of the interoperability standards in COTS and open-source.
2. The second contemplates conformance testing and the support to industry and institutions to test their own implementations.
3. The third goal of HMA-T is to support the take up of HMA standards.

Overall, the testbeds are designed to conduct research, prototype, and to document the entire procedure. They will give insights to potential interoperability challenges and demonstrate the wide scope of the specifications. With this in mind, the present proposal will focus on several COTS and infrastructures that transversely cross the application domain of the specification. From catalogues, processing infrastructures, security, data server and clients this proposal intends to present a coherent yet independent view of the overall applicability of the HMA interoperability effort.

4.1.2 User Management OGC Specification

The OGC User Management Interfaces for Earth Observation describes how user and identity management information is included the protocol specification for Earth Observation Services. It explains how to pass identity information to Web Services. This specification aims at describing the federated management interface supported by Earth Observation data and service providers.

It proposes the following approach:

1. The Identity Provider accepting user information like username and password returns a SAML token which authenticates the user;
2. All client requests include the SAML token in the SOAP header of its messages;
3. The Web service provider only accepts requests based on the content of the message header.

The SAML token provided by the identity provider is based on the Web Service Client. The Web Service Client (SAML) is an XML-based standard for enabling the exchange of authentication and authorization data between security domains to solve the Single Sign-On issue. In this exchange occurs between producer of assertions, the identity provider, and the assertion consumer, the service provider. SAML assumes the user or the principal is recognized by at least one identity provider and this last is expected to provide the user with local authentication services. Then, the service provider that in the scope of this document is the HMA-T/G-POD Gateway, relies on the identity provider to identify the user. Thus, the client presents the SAML assertion supplied by the Identity Provider to the service provider and it is on the basis of this assertion that the access control decision is taken.

A SAML assertion is a package of information containing a number of statements made by the Identity Provider. The basic EO DAIL proposes a minimum profile with the statements to be included in the SAML token:

- Unambiguous HMA identity
- Country of origin
- Organization
- hmaProjectName
- hmaAccount
- hmaServiceName

The password and the enabled/disabled state of a user is held in the Identity Provider registry but this information is not included in the SAML statements.

4.1.3 gridify - Service Application Environment

Terradue’s gridify is a service application environment solution that presents a new vision to the production, management and distribution of spatial data where grid technology joins geographical data inventory and query systems. It enables the implementation and configuration of composite services requiring the use of substantial computer and data resources.

Environmental services and applications are increasingly collaborative and multi-disciplinary, and it is not unusual for teams to span institutions, states, countries and continents. Grid technologies seek in a flexible and controlled way to link data, computers and resources into single virtual organizations by providing the protocols and services. With the gridify development, Terradue works to exploit and strengthen best practices in distributed data processing, archiving and discovery for Earth sciences, finance and medicine. The emphasis is on the immediate delivery of robust operational systems while keeping a concrete roadmap to build the next generation data processing and storage systems. Collaboratively with best-known open source projects Terradue works to optimize data discovery, processing, archiving and distribution with peer-to-peer technology.

Coupled with a networks of computers, gridify provides the flexibility for the integration of applications with data workflows, distributed data access, grid
computing and storage capability. With a common subset of well-known standards for geographic metadata, gridify delivers a set of services for indexing, searching, sharing and storing very large spatial data sets. Together, it constitutes the core of what is needed to participate in a spatial data infrastructure.

The gridify product is built on the heritage of the European Space Agency (ESA) GRID Processing on-Demand infrastructure (G-POD) initiated in 2003, developed and maintained by Terradue as an European Space Agency's incubated spin-off. Through the incubation process started in 2006 Terradue pushed the development of gridify into a generic product enlarging its use beyond the Earth observation applications and services. The product is focused for two main usages:

- An off-the-shelf solution for a seamless implementation and support of high performance computing architectures for industries or research facilities with intensive services and applications
- An interoperable connectivity with external shared resources promoting collaboration and developing bridges between the Industry/Research institutions to share mutual and dynamically available resources for institutions with low computational and storage resources with a sporadic need to run demanding services and applications.

The gridify web portal is capable of managing users account and files, applications and services, instantiating and scheduling tasks, submitting and retrieving status query of tasks on different grid infrastructures and middlewares.

Working together with an abstraction layer, it builds a reusable framework taking seamless care of the data housekeeping and basic workflow processing tasks recurrent to most data processing applications. The creation and connection to secure grid sessions, file registry and retrieval, job submission together with session and job status are included among these capabilities. All the necessary grid operations performed in all phases, such as applications and data files transfer, grid job status, exception and error management were virtualized to different grid middlewares.

The gridify is currently the software that powers the ESA Earth Observation Grid Processing on-Demand, the Earth observation business experiment in the BEinGrid European Commission (EC) project, the ESA Ionia GlobCover Data Distribution Center and it will also power the EC’s 7th framework programme project GENESI-DR (Ground European Network for Earth Science Interoperations - Digital Repositories) application web portal.

4.1.4 ESA Grid Processing on-Demand (G-POD)

The validation of earth remote sensing satellite data, and associated algorithms, often requires a large amount of processing resources and highly interactive access to the large amount of satellite data. On the other-hand, the cost of the necessary computing infrastructure coupled with the access to large volumes of satellite data is often too high to motivate local investments. As such, providing a common shared infrastructure accessible to science users is a very valuable and cost effective approach to support EO science and application development. Grid
Processing on-Demand\textsuperscript{1}, an ESA initiated project and then industrialized by Terradue provides a “user-segment” putting EO data and processors together.

ESA G-POD is a generic GRID-based operational environment for Earth Observation applications where specific data handling applications can be seamlessly plugged into system. Coupled with high-performance and sizeable computing resources managed by GRID technologies, it provides the necessary flexibility for building an application virtual environment with quick accessibility to data, computing resources and results.

\textbf{Figure 1: G-POD Web Portal based on gridify}

The gridify web portal component used in G-POD (Error! Reference source not found.) delivers a flexible, secure, generic and distributed platform where the user can easily manage all its tasks. From the creation of a new task to the result publication, passing by the data selection and the job monitoring, the user goes trough a friendly and intuitive interface accessible from everywhere. Using this infrastructure, ESA also delivers a regular sustained opportunity available to

\begin{itemize}
  \item \textsuperscript{1} http://gpod.eo.esa.int
\end{itemize}
scientists. This opportunity, referred to as the "G-POD CAT-1\(^2\), creates a partnership opportunity for conducting Earth Science research activities through grid technology. Within this framework, ESA offers on-line access to data, together with the G-POD attached computing infrastructure to host and run the partner's applications.

The grid model has proven to overcome known problems that haunt Earth observations services with the paradigm change in the data and application flows. This paradigm change reduces the strain on the network, optimizes the use processing resources, reduces substantially or eliminates completely data access time delays and shipping issues and provides a framework for IPR enforcement and management. By reducing or overcoming these obstacles, the G-POD model is currently enthusiastically accepted by the research community. Adoption of the HMA will allow for wider use of Earth observation services and expand its research user community.

In parallel to the G-POD Web Portal, the G-POD services are published via a job-control interface implemented using SOAP. These services can be invoked via SOAP Web Services using the interface protocol implemented by GRID Processing on-Demand (G-POD) for remote job submission and control from client computers [R3].

The G-POD Web service invocation follows a protocol sequence summarized below and depicted in Figure 2:

- First the client computer shall initialize a session with the G-POD WS server and authenticate itself
- Then, the G-POD WS application services available to the authenticated client may be triggered and monitored independently
- Finally, the client optionally releases its active session with the server

![Figure 2: G-POD Job Submission Web Service flow](http://eopi.esa.int/G-POD)
The session initialization is performed via a SOAP service call to the ifyService port on the G-POD Web service server requiring user authentication (previously registered in the G-POD User Registry).

The Connect call returns a G-POD session token to the client, to be used as reference in all subsequent calls to the interface within the defined session. The session token is used as SOAP header defined as a SessionKeyType in any subsequently call to the G-POD Web service. The validity and certification of the token is made in the subsequent requests (in other terms, a session token does not guarantee the ability to perform any service execution, it just authenticates the user). The G-POD application service process invocation and monitoring can only be performed once a session is open with the G-POD Web service server and application services with a session token linking to an authenticated-user can be triggered within the parent session.

4.2 Architecture Overview

4.2.1 Functional Architecture

The Functional Architecture of the HGG is given in Figure 3.

The system is composed of two sub-systems:

- The HGG Server System hosting the HMA-T/G-POD Gateway;

![Figure 3: Functional Architecture](image)

Each Element is further explained in section 4.2.3.
4.2.2 System Security
The HGG System is protected from external intrusion by the **OS kernel firewall**.

4.2.3 System Architecture

Figure 4 gives the internal decomposition of the HGG and its relation with external systems and/or actors.

**Figure 4: HMA-TG-POD Gateway Architecture**

As a result of the System Design phase, eight main elements have been identified to cover the System requirements:

- A HGG Service Invoker (HGG-SI);
- A HGG Access Point (HGG-AP);
- A HGG Policy Enforcement Point (HGG-PEP);
- A HGG Identity Handler (HGG-IH);
- A HGG Dynamic User Registry (HGG-DUR);
- A HGG G-POD Web Service Client (HGG-GWSC);
- The G-POD Web Service;
- The G-POD Web Portal.

Each Main element can be further decomposed in sub-elements, to map the various system functionalities:

- HGG Service Invoker (HGG-SI):
  - Invoke the HGG Web Service methods.
  - Send requests to the HGG-AP
  - Receive responses from the HGG-AP
- HGG Access Point (HGG-AP):
  - Handle the SOAP messages sent by the HGG-SI.
- HGG Access Point (HGG-PEP):
Dispatching the request processing according to the authentication state.

- **HGG-IH Identity Handler (HGG-IH):**
  - Check the SAML assertion signature;
  - Decrypt the SAML assertion;
  - Extract the user identity from the SAML assertion.

- **HGG-DUR Dynamic User Registry (HGG-DUR):**
  - Check if there is a G-POD session token for the SAML token;
  - Request a G-POD session token;
  - Associate the G-POD and SAML tokens.

- **HGG G-POD Web Service Client (HGG-GWSC):**
  - Invoke the G-POD Web Service.

- **G-POD Web Service:**
  - Receive requests from the HGG-GWSC;
  - Send responses to the HGG-GWSC;
  - Communicate with the G-POD Portal

- **G-POD Web Portal:**
  - Handle the G-POD Web Service communications.

### 4.3 Interface Context

Two external interfaces have been identified:

- The End Users who connect to the HGG via the clients (e.g. the HGG Service Invoker);
- The G-POD Web Service.

No specific Interface Control Documents have been generated within the scope of the HGG Project for these two interfaces.
5 System Software Design

5.1 Overall Architecture

Figure 5 gives the HGG component view and the corresponding dependencies between these components.

![Figure 5 HMA-T G-POD Detailed Architecture](image)

5.1.1 Concept

The HMA-T/G-POD Gateway is the access point to G-POD services for users authenticated via the SAML assertion mechanism. In particular, the HMA-T/G-POD Gateway acts as a Service Provider through its Web service interface while G-POD Web Service is only accessed by the gateway. In its current implementation, G-POD Web Service can only identify a user request via an initial connection where a session token is provided upon a successful authentication request.

The objective of the HGG System is to provide users with a valid SAML token returned by an authentication Web Service (the authentication service is out of scope for this activity) the possibility to invoke through it the G-POD Web service. This way, the HGG System provides a G-POD Policy Enforcement Point (PEP) where a valid SAML token must be presented in order to access the G-POD Web service. This PEP is able to decide based on the content of the message header (including the authentication token) of the client requests whether to accept or refuse the service request and forward it to the G-POD Web service.

The authentication service via an Identity provider is out of the scope of the HGG System and it is assumed the client has already a SAML token included in all its requests. Within this activity, a Service Invoker is nevertheless provided to invoke the HGG Web service acting as a client. In a real life scenario, the client would have to invoke the authenticate method of an authentication Web service to obtain a SAML token and then invoke the HGG members.

The Service Invoker sends requests (e.g. Service submission or description) containing the SAML token to the HGG Access Point. The HGG Access Point handles all the SOAP requests and is able to submit the SAML token to the HGG Identity Handler.

At this point, the SAML token signature is verified using the IdP’s public key. If the signature cannot be successfully verified, the request is refused and a
message is sent back to the client. Once the SAML token is successfully verified, the SAML token is decrypted and the identity of the user is extracted.

This information is used by the HGG Dynamic User Registry to look for a G-POD session token match. If none is found, it means that this request is the first one and a contact with the G-POD Web service is required to obtain a G-POD session token.

This step is managed by the HGG G-POD Web service Client (HGG-GWSC). It invokes the connect method of the G-POD Web Service with the username and password extracted from the SAML token. If successful, the user is authenticated and authorized in the G-POD Portal. At this point the HGG Dynamic User Registry creates a new entry containing the SAML token, the G-POD token and the user details with at least the username and password. The request can, at this point, be transmitted to the G-POD Web Service.

5.1.2 System Interfaces

The section contains the HGG System internal and external interfaces.

5.1.2.1 Internal Interfaces

Table 1 gives a summary of the HMA-T/G-POD Gateway internal interfaces.

Table 1: HMA-TGP-POD Internal Interfaces

<table>
<thead>
<tr>
<th>Service Invoker</th>
<th>Access Point</th>
<th>Policy Enforcement Point</th>
<th>Identity Handler</th>
<th>Dynamic User Registry</th>
<th>G-POD Web Service Client</th>
<th>G-POD Web Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Invoker</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Access Point</td>
<td></td>
<td>API</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Policy Enforcement Point</td>
<td>API</td>
<td>API</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Identity Handler</td>
<td></td>
<td>API</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dynamic User Registry</td>
<td></td>
<td>API</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G-POD Web Service Client</td>
<td></td>
<td>API</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G-POD Web Service</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5.1.2.2 External Interfaces

There are two actors interacting with the HGG System, namely:
• The Service Invoker;
• The G-POD Web Service.

Table 2 gives a summary of the HMA-T/G-POD Gateway external interfaces.

<table>
<thead>
<tr>
<th>Service Invoker</th>
<th>Access Point</th>
<th>Policy Enforcement Point</th>
<th>Identity Handler</th>
<th>Dynamic User Registry</th>
<th>G-POD Web Service Client</th>
<th>G-POD Web Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Invoker</td>
<td>SOAP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Access Point</td>
<td>SOAP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Policy</td>
<td>SOAP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enforcement</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Point</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Identity</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Handler</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dynamic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Registry</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G-POD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Web</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Service</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Client</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G-POD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SOAP</td>
</tr>
<tr>
<td>Web</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Service</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SOAP</td>
</tr>
</tbody>
</table>

5.2 System Software Components Description

This section gives the detailed description of the elements listed in section 0:

• A HGG Service Invoker (HGG-SI);
• A HGG Access Point (HGG-AP);
• A HGG Access Point (HGG-PEP);
• A HGG Identity Handler (HGG-IH);
• A HGG Dynamic User Registry (HGG-DUR);
• A HGG G-POD Web Service Client (HGG-GWSC);
• The G-POD Web Service;

Each component is described using the following convention:

| ID  | Component unique identifier. |
### Type

- **Logical and physical characteristics.**
  - Logical characteristics = the package, library or class that the component belongs to.
  - Physical characteristics = the type of component (task, database).

### Purpose

The purpose of a component, which could be used to trace to the software requirements that it implements.

### Function

The function of a component (states what the component does).

This generally the description of the process (in structured English).

### Subordinate

The subordinates of the component (immediate children).

### Interfaces

Interface with other components.

### Resources

The resources’ needs of the component.

### Data

Data internal to a component.

## 5.2.1 HGG Service Invoker

A HGG client needs to implement the authenticate method of the OGC User Management Interfaces for Earth Observation 07-118r1 specification. The HGG Service Invoker behaves like a regular HGG client except it does not implement the authenticate method and already has a SAML token.

The HGG Service Invoker is a C# SOAP command line client exposing the methods listed below:

- **SetVariable** - Defines session states variables to enhance the job submission. The possible values for the Name variable are:
  - CE: defines the default CE (hostname) to launch all session’s tasks. This will be discarded if the service is configures to a single CE.
  - PRIORITY: Defines the priority to launch all the session’s tasks.

- **GetVariable** - Retrieves the session states variables. The possible values for the Name variable are:
  - CE: defines the default CE (hostname) to launch all session’s tasks. This will be discarded if the service is configures to a single CE.
  - PRIORITY: Defines the priority to launch all the session’s tasks.

- **SubmitExt** - The SubmitExt operation creates a task of the ServiceName type and submits it for execution. A unique task identifier (taskToken) is returned on submission success, otherwise a SOAP exception is returned.

- **Status** - Returns the current status of the service process referred by the value returned by the SubmitExt function. The keyword “DONE” is returned on task successful completion.

- **GetTaskMetadata** - This operation lists in a key name value pair array all the task parameters classified as metadata by the actual G-POD service being accessed.
- **DescribeService** – This operation lists in an array of feature type the necessary parameters for a given service and their actual meaning.
- **Results** – Returns an array of strings with the URL to access the results published on the portal associated to this task. If the results were published on external server the array will be empty.

The detailed description of the methods is provided below.

<table>
<thead>
<tr>
<th>ID</th>
<th>ServiceInvoker</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>C# Web service client</td>
</tr>
</tbody>
</table>

**Purpose**
Invoke the HGG-AP methods **SetVariable**, **GetVariable**, **SubmitExt**, **Status**, **GetTaskMetadata**, **DescribeService** and **Results**

**Function**

**SetVariable**

```csharp
string IfyService.SetVariable(
    string Name,
    string Value
)
```

Retrieves the session states variables. The possible values for the **Name** variable are:

- **CE**: defines the default CE (hostname) to launch all session’s tasks. This will be discarded if the service is configures to a single CE.
- **PRIORITY**: Defines the priority to launch all the session’s tasks.

**Parameters:**
- **string Name** - The name of the session state variable
- **string Value** - The new value of the session state variable

**Returned value:**
The new value of the session state variable name

**GetVariable**

**SubmitExt**

```csharp
string SubmitExt = IfyService.SubmitExt(
    string ServiceName,
    string Description,
    ArrayOfFeatureType Parameters,
    string OutputBoxUrl,
    string NotificationPort
)
```

The **SubmitExt** operation creates a task of the **ServiceName** type and submits it for execution. A unique task identifier
Parameters:

- **string ServiceName** – The service name. This is a keyword that should be delivered to the user when access is granted to the web service. Please contact site admin if this value is unknown.
- **string Description** – The new task description.
- **ArrayOfFeatureType Parameters** – An array of FeatureType with the pairs name-value for the service.
- **string OutputBoxUrl** – URL prefix of the output server, including protocol, server name and directory pointing to a user storage registered in the G-POD portal.
- **string NotificationPort** – SOAP port address hosting the IContact notification client-service. To be left Empty if explicit notification to the client is not required.

Returned value:

- A unique task identifier is returned on submission success, otherwise a SOAP exception is returned. This value must be used to query the status of the task.

**Status**

```plaintext
string IfyService.Status(string TaskToken)
```

Returns the current status of the service process referred by the value returned by the SubmitExt function. The keyword “DONE” is returned on task successful completion.

Parameters

- **string TaskToken** – As returned by the IfyService.SubmitExt operation

Returned Value:

One of the following strings:

- "PENDING" - Task is accepted and is waiting for resources availability
- "ACTIVE" - Task is being executed
- "DONE" - Task is successfully terminated
- "FAILED" - Task terminated with error

**GetTaskMetadata**

```plaintext
ArrayOfString IfyService.GetTaskMetadata(
```

classified as metadata by the actual G-POD service being accessed.

Parameters
- string TaskToken - As returned by the IOServices.Submit operation

Returned Value:
- Key name and value array with the task metadata

**DescribeService**

ArrayOfFeatureType IfyService.DescribeService(string ServiceName)

List in an array of feature type the necessary parameters for a given service and their meaning.

Parameters
- string ServiceName – String Token identifying the service as given by the site admin

Returned Value
- Array with the name, description and mandatory values for a given service

**Results**

ArrayOfString IfyService.Results(string TaskToken)

Returns an array of strings with the URL to access the results published on the portal associated to this task. If the results were published on external server the array will be empty.

Parameters
- string TaskToken - value returned by the IfyService.SubmitExt operation

Returned Value
- List of URLs to access the files that were published on the portal

<table>
<thead>
<tr>
<th>Subordinate</th>
<th>N/A</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interfaces</td>
<td>SOAP reader/writer</td>
</tr>
<tr>
<td>Resources</td>
<td>N/A</td>
</tr>
<tr>
<td>Data</td>
<td>SAML token</td>
</tr>
</tbody>
</table>

### 5.2.2 HGG Access Point

The HGG Access Point (HGG-AP) is responsible for receiving requests from and sending responses to clients. Its tasks are mainly the transformation of a SOAP message into method calls (i.e. triggering the actual processing of the client
request) and of method results back into a SOAP message. This is usually done automatically by the runtime environment for the Web Service processing.

<table>
<thead>
<tr>
<th>ID</th>
<th>Purpose</th>
<th>Function</th>
<th>Subordinate</th>
<th>Interfaces</th>
<th>Resources</th>
<th>Data</th>
</tr>
</thead>
</table>
| HGG-AP | This component handles the client requests and responses | • Receive the client SOAP request message  
• Deserialize the SOAP request message  
• Call the appropriate Web Service method  
• Serialize result of the Web Service method in the SOAP response message  
• Send the SOAP response message to the Client  
• Provide the definition of the HGG Web Service in WSDL | N/A | SOAP reader/writer  
HGG-PEP API | N/A | N/A |

### 5.2.3 HGG Policy Enforcement Point

The HGG Policy Enforcement Point (HGG-PEP) is a C# SOAP Web Service class within an ASMX page. It is the core component of the HGG and implements the methods the clients can use (as described in section 5.2.1).

The HGG-PEP asks the HGG-IH for the verification of the user identity. According to the result of this check, the component reacts in different ways:

<table>
<thead>
<tr>
<th>Result of User Identity Handler check</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>User not authenticated</td>
<td>Refuse request</td>
</tr>
<tr>
<td>User authenticated, but no G-POD session information available</td>
<td>Instantiate HGG-GWSC, create new G-POD session and forward the request to the HGG-GWSC</td>
</tr>
<tr>
<td>User authenticated and G-POD session information available</td>
<td>Instantiate HGG-GWSC and forward the request to the HGG-GWSC</td>
</tr>
</tbody>
</table>

For details on the processing performed by the User Identity component, see section 5.2.4.
### Type
The HGG-AP-WS is a C# Web Service class

### Purpose
HGG core component for forwarding or refusing request.

### Function
- Receive request from HGG-AP
- Retrieve authentication information from HGG-IH
- Forward request to HGG-GWSC or refuse it depending on the authentication state of the requesting subject

### Subordinate
HGG-IH

### Interfaces
- HGG-AP API
- HGG-IH API
- HGG-GWSC API

### Resources
N/A

### Data
N/A

#### 5.2.4 HGG Identity Handler

<table>
<thead>
<tr>
<th>ID</th>
<th>HGG-IH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>The HGG-IH is a C# class</td>
</tr>
</tbody>
</table>

### Purpose
The purpose of the HGG-IH is to verify, decrypt and extract authentication information contained in the SAML assertion. The HGG-IH asks the HGG Dynamic User Registry whether a G-POD session is available and returns this information to the HGG-PEP.

### Function
- Verify the SAML assertion signature. The SAML assertion is signed by the Identity Provider using its public key. The SAML assertion signature is verified by the HGG-IH to make sure the signature is valid and not forged.
- Decrypt the SAML assertion. The SAML assertion is encrypted by the Identity Provider as such its content must be decrypted by the HGG-IG using the **IdP private key**.
- Extract the authentication information contained in the SAML assertion
- Interrogate the HGG-DUR on open G-POD sessions of the given user
- Return the result of the processing to the calling component (HGG-PEP).

### Subordinate
The subordinates of the component (immediate children).

### Interfaces
- HGG-PEP-WS API
- HGG-DUR API
5.2.5 HGG Dynamic User Registry

<table>
<thead>
<tr>
<th>ID</th>
<th>HGG-DUR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>The HGG-DUR is composed of a C# class and a relational database to store the client authentication information and the G-POD information such as the Web Service session token</td>
</tr>
<tr>
<td>Purpose</td>
<td>The HGG-DUR is a dynamic registry able to store the user authentication information extracted from the SAML assertion and the G-POD Web Service session token needed by G-POD to trace the request. The HGG provides the methods to check if the user has already a G-POD session token available and if so use it to pass the arguments to the HGG G-POD Web Service Client. If no match is found between the user authentication information and a G-POD token the HGG-DUR invokes the HGG G-POD Web Service Client connect method to obtain a G-POD session token. The HGG-DUR will then use this G-POD session token in all interaction with the HGG-Web Service Client.</td>
</tr>
</tbody>
</table>
| Function| • The HGG-DUR checks the existence of a G-POD Session token associated to the user authentication information passed by the HGG-IH.  
• If no association is found, the HGG-DUR contacts the HGG-GWSC to generate a G-POD session token to be used in all HGG-Client communications. |
| Subordinate| N/A |
| Interfaces| HGG-IH API  
HGG-GWSC API |
| Resources| N/A |
| Data| N/A |

5.2.6 HGG G-POD Web Service Client

<table>
<thead>
<tr>
<th>ID</th>
<th>HGG-GWSC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>The HGG-GWSC is a C# SOAP client</td>
</tr>
<tr>
<td>Purpose</td>
<td>The purpose of the HGG-GWSC is to handle all communications to and from the G-POD Web service via a SOAP interface. As a G-POD Web service it implements all the methods described in [R3].</td>
</tr>
</tbody>
</table>
### Connect

```csharp
string IfyService.Connect(
    string Username,
    string Password
)
```

A session token is returned is successful. A security exception is launched if the user or the password is invalid.

**Parameters**

- **string Username** – The user’s identity in the portal
- **string Password** – The user’s password

**Returned Value**

- This string must be used for any further contact with the web service as a header and defined as `KeyIdentifier` type

### Disconnect

```csharp
void IfyService.Disconnect()
```

Even if the task automatically disconnects after the task timeout defined in the G-POD being reached, it is possible to explicitly disconnect from a task using the Disconnect method.

**Note:** disconnecting from a session does not clear or delete any of the tasks nor invalidates any of the tokens obtained with the task submitted.

**Parameters**

- N/A

**Returned Value**

- N/A

### SetVariable

```csharp
string IfyService.SetVariable(
    string Name,
    string Value
)
```

Retrieves the session states variables. The possible values for the Name variable are:

- **CE:** defines the default CE (hostname) to launch all session’s tasks. This will be discarded if the service is configures to a single CE.
- **PRIORITY:** Defines the priority to launch all the session’s tasks.

**Parameters:**

- **string Name** - The session state variable name
- `string Value`: The new value of the session state variable name

**Returned value:**
- The new value of the session state variable name

### GetVariable:

### SubmitExt:

```csharp
string IfyService.SubmitExt(string ServiceName, string Description, ArrayOfFeatureType Parameters, string OutputBoxUrl, string NotificationPort)
```

The `SubmitExt` operation creates a task of the `ServiceName` type and submits it for execution. A unique task identifier (task token) is returned on submission success, otherwise a SOAP exception is returned.

**Parameters:**
- `string ServiceName`: The service name. This is a keyword that should be delivered to the user when access is granted to the web service. Please contact site admin if this value in unknown
- `string Description`: The new task description
- `ArrayOfFeatureType Parameters`: An array of `FeatureType` with the pairs name-value for the service
- `string OutputBoxUrl`: URL prefix of the output server, including protocol, server name and directory pointing to a user storage registered in the G-POD portal
- `string NotificationPort`: SOAP port address hosting the IContact notification client-service. To be left Empty if explicit notification to the client is not required.

**Returned value:**
- A unique task identifier is returned on submission success, otherwise a SOAP exception is returned. This value must be used to query the status of the task.

### Status:

```csharp
string IfyService.Status(string TaskToken)
```

Returns the current status of the service process referred by the value returned by the `SubmitExt` function. The keyword "DONE" is returned on task successful completion.

**Parameters**
string TaskToken - As returned by the IfyService.SubmitExt operation

Returned Value:

Possible values are:

- "PENDING" - Task is accepted and is waiting for resources availability
- "ACTIVE" - Task is being executed
- "DONE" - Task is successfully terminated
- "FAILED" - Task terminated with error

GetTaskMetadata:

ArrayOfString IfyService.GetTaskMetadata( string TaskToken )

List in a key name value pair array all the task parameters classified as metadata by the actual G-POD service being accessed.

Parameters

- string TaskToken - As returned by the IOServices.Submit operation

Returned Value:

- Key name and value array with the task metadata

DescribeService:

ArrayOfFeatureType IfyService.DescribeService( string ServiceName )

List in an array of feature type the necessary parameters for a given service and their meaning.

Parameters

- string ServiceName - String Token identifying the service as given by the site admin

Returned Value

- Array with the name, description and mandatory values for a given service

Results:

ArrayOfString IfyService.Results(string TaskToken)

Returns an array of strings with the URL to access the results published on the portal associated to this task. If the results were published on external server the array will be empty.
<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>- string TaskToken - value returned by the IfyService.SubmitExt operation</td>
<td></td>
</tr>
<tr>
<td>Returned Value</td>
<td></td>
</tr>
<tr>
<td>- The URLs to access the files that were published on the portal</td>
<td></td>
</tr>
<tr>
<td>Subordinate</td>
<td>N/A</td>
</tr>
<tr>
<td>Interfaces</td>
<td>HGG-PEP API</td>
</tr>
<tr>
<td></td>
<td>HGG-DUR API</td>
</tr>
<tr>
<td></td>
<td>G-POD Web Service SOAP writer/reader</td>
</tr>
<tr>
<td>Resources</td>
<td>N/A</td>
</tr>
<tr>
<td>Data</td>
<td>N/A</td>
</tr>
</tbody>
</table>
## 6 REQUIREMENTS TO DESIGN COMPONENTS

**TRACEABILITY**

<table>
<thead>
<tr>
<th>System Requirements</th>
<th>Component</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBW</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TBW</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>