D3.1.2

WP3.1 Detailed Overall Architecture

V1.0

WP 3.1 Overall Architecture Definition and Layer Integration

Dissemination Level: Public

Lead Editor: Jürgen Jähnert, USTUTT/RUS

15/02/2006

Status: Final

SIXTH FRAMEWORK PROGRAMME

PRIORITY IST-2002-2.3.1.18

Grid for complex problem solving

Proposal/Contract no.: 004293

2. Fair Use Rights. Nothing in this license is intended to reduce, limit, or restrict any rights arising from fair use, first sale or other limitations on the exclusive rights of the copyright owner under copyright law or other applicable laws.

3. License Grant. Subject to the terms and conditions of this License, Licensor hereby grants You a worldwide, royalty-free, non-exclusive, perpetual (for the duration of the applicable copyright) license to exercise the rights in the Work as stated below:

a. to reproduce the Work, to incorporate the Work into one or more Collective Works, and to reproduce the Work as incorporated in the Collective Works;
b. to distribute copies or phonorecords of, display publicly, perform publicly, and perform publicly by means of a digital audio transmission the Work including as incorporated in Collective Works.
c. For the avoidance of doubt, where the work is a musical composition:
    i. **Performance Royalties Under Blanket Licenses**. Licensor waives the exclusive right to collect, whether individually or via a performance rights society (e.g. ASCAP, BMI, SESAC), royalties for the public performance or public digital performance (e.g. webcast) of the Work.
    ii. **Mechanical Rights and Statutory Royalties**. Licensor waives the exclusive right to collect, whether individually or via a music rights society or designated agent (e.g. Harry Fox Agency), royalties for any phonorecord You create from the Work ("cover version") and distribute, subject to the compulsory license created by 17 USC Section 115 of the US Copyright Act (or the equivalent in other jurisdictions).
d. **Webcasting Rights and Statutory Royalties**. For the avoidance of doubt, where the Work is a sound recording, Licensor waives the exclusive right to collect, whether individually or via a performance-rights society (e.g. SoundExchange), royalties for the public digital performance (e.g. webcast) of the Work, subject to the compulsory license created by 17 USC Section 114 of the US Copyright Act (or the equivalent in other jurisdictions).

The above rights may be exercised in all media and formats whether now known or hereafter devised. The above rights include the right to make such modifications as are technically necessary to exercise the rights in other media and formats, but otherwise you have no rights to make Derivative Works. All rights not expressly granted by Licensor are hereby reserved.

4. Restrictions. The license granted in Section 3 above is expressly made subject to and limited by the following restrictions:

a. You may distribute, publicly display, publicly perform, or publicly digitally perform the Work only under the terms of this License, and You must include a copy of, or the Uniform Resource Identifier for, this License with every copy or phonorecord of the Work You distribute, publicly display, publicly perform, or publicly digitally perform. You may not offer or impose any terms on the Work that alter or restrict the terms of this License or the recipients' exercise of the rights granted hereunder. You may not sublicense the Work. You must keep intact all notices that refer to this License and to the disclaimer of warranties. You may not distribute, publicly display, publicly perform, or publicly digitally perform the Work with any technological measures that control access or use of the Work in a manner inconsistent with the terms of this License Agreement. The above applies to the Work as incorporated in a Collective Work, but this does not require the Collective Work apart from the Work itself to be made subject to the terms of this License. If You create a Collective Work, upon notice from any Licensor You must, to the extent practicable, remove from the Collective Work any credit as required by clause 4(b), as requested.
b. If you distribute, publicly display, publicly perform, or publicly digitally perform the Work or Collective Works, You must keep intact all copyright notices for the Work and provide, reasonable to the medium or means You are utilizing: (i) the name of the Original Author (or pseudonym, if applicable) if supplied, and/or (ii) if the Original Author and/or Licensor designate another party or parties (e.g. a sponsor institute, publishing entity, journal) for attribution in Licensor's copyright notice, terms of service or by other reasonable means, the name of such party or parties; the title of the Work if supplied; and to the extent reasonably practicable, the Uniform Resource Identifier, if any, that Licensor specifies to be associated with the Work, unless such URI does not refer to the copyright notice or licensing information for the Work. Such credit may be implemented in any reasonable manner; provided, however, that in the case of a Collective Work, at a minimum such credit will appear where any other comparable authorship credit appears and in a manner at least as prominent as such other comparable authorship credit.

5. Representations, Warranties and Disclaimer

UNLESS OTHERWISE MUTUALLY AGREED TO BY THE PARTIES IN WRITING, LICENSOR OFFERS THE WORK AS-IS AND MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND CONCERNING THE MATERIALS, EXPRESS, IMPLIED, STATUTORY OR OTHERWISE, INCLUDING, WITHOUT LIMITATION, WARRANTIES OF TITLE, MERCHANTIBILITY, FITNESS FOR A PARTICULAR PURPOSE, NONINFRINGEMENT, OR THE ABSENCE OF LATENT OR OTHER DEFECTS, ACCURACY, OR THE PRESENCE OF ABSENCE OF ERRORS, WHETHER OR NOT DISCOVERABLE. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES, SO SUCH EXCLUSION MAY NOT APPLY TO YOU.

6. Limitation on Liability. EXCEPT TO THE EXTENT REQUIRED BY APPLICABLE LAW, IN NO EVENT WILL LICENSOR BE LIABLE TO YOU ON ANY LEGAL THEORY FOR ANY SPECIAL, INCIDENTAL, CONSEQUENTIAL, PUNITIVE OR EXEMPLARY DAMAGES ARISING OUT OF THIS LICENSE OR THE USE OF THE WORK, EVEN IF LICENSOR HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

7. Termination

a. This License and the rights granted hereunder will terminate automatically upon any breach by You of the terms of this License. Individuals or entities who have received Collective Works from You under this License, however, will not have their licenses terminated provided such individuals or entities remain in full compliance with those licenses. Sections 1, 2, 5, 6, 7, and 8 will survive any termination of this License.

b. Subject to the above terms and conditions, the license granted here is perpetual (for the duration of the applicable copyright in the Work). Notwithstanding the above, Licensor reserves the right to release the Work under different license terms or to stop distributing the Work at any time; provided, however that any such election will not serve to withdraw this License (or any other license that has been, or is required to be, granted under the terms of this License), and this License will continue in full force and effect unless terminated as stated above.

8. Miscellaneous

a. Each time You distribute or publicly digitally perform the Work, the Licensor offers to the recipient a license to the Work on the same terms and conditions as the license granted to You under this License.
b. If any provision of this License is invalid or unenforceable under applicable law, it shall not affect the validity or enforceability of the remainder of the terms of this License, and without further action by the parties to this agreement, such provision shall be reformed to the minimum extent necessary to make such provision valid and enforceable.
c. No term or provision of this License shall be deemed waived and no breach consented to unless such waiver or consent shall be in writing and signed by the party to be charged with such waiver or consent.
d. This License constitutes the entire agreement between the parties with respect to the Work licensed here. There are no understandings, agreements or representations with respect to the Work not specified here. Licensor shall not be bound by any additional provisions that may appear in any communication from You. This License may not be modified without the mutual written agreement of the Licensor and You.

Context

Activity 3	**Design**
WP 3.1	**Overall Architecture Definition and Layer Integration**
Task 3.1.1-5	**-**
Dependencies	The Description of Work, DoW, places the Overall Architecture to be developed in this Deliverable D3.1.2 of WP3.1 in relation to the following Workpackages and their Deliverables: D3.1.2 is influenced by • WP1.1 Market and Regulations, WP2.2 Environment, WP2.3 Test-bed Definition o D2.1.1 o D2.1.2 o D2.2.1 o D2.2.4 Report on State of the Art o D2.3.1 Test-bed Definition o All available Deliverables from AC4 D3.1.2 influences and is influenced by • D3.1.1 which can be regarded as first version of this document • WP3.2 Business Modelling and its forthcoming Deliverables o D3.2.1 The Akogrimo consolidated Value Chain o D3.2.2 The Akogrimo Business Modelling Framework Finally, D3.1.2 influences the implementation Workpackages WP4.1, Network, WP4.2 Network Middleware, WP4.3, Grid Infrastructure, and WP4.4 Grid Applications – according to the specializations and restrictions of the Overall Architecture elaborated in WP2.3/D2.3.2, 'Validation Scenarios'.

List of Authors

Jürgen Jähnert	USTUTT
Patrick Mandic	USTUTT
Robert Piotter	USTUTT
Ignaz Müller	USTUTT
Ruth del Campo	USTUTT
Annalisa Terracina	Datamat
Giuseppe Laria	CRMPA
Julian Gallop	CCLRE
Brian Ritchie	CCLRC
Damian Ma Randall	CCLRC
Tom Kirkham	CCLRC
Antonis Litke	NTUA
Brynjar Age Vicken	Telenor
Per Oddvar Osland	Telenor
Rosa Vieira	TID
Nuno Inacio	IT-Av

Review: Stefan Wesner, USTUTT

Table of Contents

1. Executive Summary 14
   1.1. Scope of this document 14
   1.2. Rationale and overall view 14
   1.3. Selected issues 15
2. Introduction 16
3. Concepts and Terms 17
   3.1. Identity, Domain and User Profile 17
      3.1.1. Domain and Trust in Akogrimo 18
      3.1.2. Personalization: User Profiles and Identities 19
      3.1.3. Privacy: Anonymity and Pseudonymity 19
      3.1.4. Single Sign On with multiple identities 20
   3.2. Mobility Concepts 20
      3.2.1. Terminal Mobility 20
      3.2.2. User Mobility 21
      3.2.3. Session Mobility 21
   3.3. Context and Device Capabilities 21
   3.4. Policies 23
   3.5. Business Process and Workflow 23
   3.6. Service Level Description and Service Specification 25
4. Derived Requirements and Technological Context 30
   4.1. Requirements from the Scenarios 30
   4.2. Requirements from the Business Modelling 30
   4.3. Requirements derived from the BAC 30
   4.4. Selected Technologies 32
      4.4.1. Profile and Identity 32
      4.4.2. Mobility and IMS 32
      4.4.3. Context and Device Capabilities 33
      4.4.4. Business Process and Workflow 34
      4.4.5. Grid Infrastructure 36
5. Architectural Overview 38
   5.1. Architectural Goals and Constraints 38
      5.1.1. Opportunities and Challenges 38
      5.1.2. The concept of Cross-Layer Cooperation 39
   5.2. The Generic Akogrimo Architecture 40

5.3. Service Structure 41
5.3.1. Steps of an Application to Become Part of the Akogrimo Platform 41
6. Akogrimo Sub-Systems and their relations and interfaces 44
6.1. Network Services Layer Architecture 44
6.1.1. IPv6/MIPv6 infrastructure 44
6.1.2. Service Description and Provisioning 46
6.1.3. Network Management 46
6.1.4. Network Resource Management 47
6.2. Network Middleware Services Layer Architecture 48
6.2.1. Context Management 48
6.2.2. Service Discovery 49
6.2.3. A4C 51
6.3. Grid Infrastructure Services Layer Architecture 51
6.3.1. Execution Management Services 51
6.3.2. Data Management Services 54
6.3.3. Monitoring Services 56
6.3.4. Service Level Agreement Enforcement 57
6.3.5. Metering Services 59
6.3.6. Policy Management Services 60
6.4. Application Support Services Layer Architecture 61
Figure 10: WP4.4 Main Components 62
6.4.1. VO Management 63
6.4.2. Business Process Enactment (BPE) 67
6.4.3. SLA High Level Services 71
7. USE Case Specification 74
7.1. Context adaptation 74
7.2. SIP integration 75
7.3. Data management/Execution 77
7.4. OP Vo creation 80
8. Selected key Scenarios 82
8.1. Akogrimo Log-In 82
8.2. Service Invocation and Charging 84
8.3. Base VO Registration 86
8.4. OpVO Creation 89
8.5. Context Sensitive Services 91

9. Conclusion and Outlook 94
10. REFERENCES 95

List of Figures

Figure 1: Akogrimo example of trust establishment 19
Figure 2: Structure of Akogrimo Context Awareness system 22
Figure 3: Example of stateful resources for the e-Health application 28
Figure 4: Integration between GASS and lower layers 29
Figure 5 Akogrimo Basic Components 40
Figure 6: Network Layer Diagram 44
Figure 7: Management components hierarchy (A4C Server not depicted) 47
Figure 8: Network middleware components and interfaces 48
Figure 9: From BVO to OpVO 62
Figure 10: WP4.4 Main Components 62
Figure 11: VO management modules Overview 65
Figure 12: BPE subsystem modules Overview 69
Figure 13: SLA HLS subsystem modules overview 72
Figure 14: Login Sequence (1) 83
Figure 15: Login Sequence (2) 84
Figure 16: Service Invocation and Charging 86
Figure 17: Base VO Registration 88
Figure 18: OpVO Creation (1) 90
Figure 19: OpVO Creation (2) 90
Figure 20: OpVO Creation (3) 91
Figure 21: Context Sensitive Workflow 92

List of Tables

Table 1: QoS Bundles 26
Table 2: Service Interactions 43
Table 3: Interfaces for the Execution Management component 54
Table 4: Interfaces for the Data Management component 55
Table 5: Interfaces for the Monitoring Component 56
Table 6: Interfaces for the SLA enforcement component 58
Table 7: Interfaces for the Metering component 59
Table 8: Interfaces for the Policy Manager 60
Table 9: Interfaces and interactions of VO management 66
Table 10: Involved Technologies 67
Table 11: Interfaces and interactions of BPE subsystem modules 70
Table 12: Involved Technologies 70
Table 13: Interfaces and interactions of SLAHLS subsystem modules 73
Table 14: Involved technologies 73

Definitions

Access

It refers to the process of interacting with a system entity in order to manipulate, use, gain knowledge of, and/or obtain a representation of some or all of a system entity's resources. [RFC2828]

Access Control

It is the protection of resources against unauthorized access; a process by which use of resources is regulated according to a security policy and is permitted by only authorized system entities according to that policy. [RFC2828]

A4C Client

An A4C Client is a component able to connect and exchange information with an A4C Server.

A4C Server

The A4C Server is the component in charge of:

- making authentication and authorization decisions
- collecting accounting and auditing data
- perform charging calculations

Domain

A domain is the administrative aspect of an *organization* (e.g. an Enterprise or a Company, a University Department or a Hospital) that comprises a set of individuals and resources. In a domain, the resources (hardware and software) are owned and managed by a common administrative authority. In our context, the domain term is not tied to the Domain Name System (DNS).

Home Domain

The Home Domain (HD) is the domain of an organization who owns user credentials. The term *home domain* can only be used in connection with a specific user. The HD will authenticate the user, create the authentication assertion and generate the IDToken.

Foreign Domain

T%he Foreign Domain (FD) can be any domain which is not the HD of a user. In networking, the FD is the domain the user is connected to while roaming. In Akogrimo, the FD for a user is the domain of any Service Provider who can provide a service to that user, if it is not his HD. A user can use services in multiple FDs at the same time.

Customer Domain

The Customer Domain (CD) is the domain of an Organization that buys services available in the Grid. It has

to have some Grid services in order to manage the interactions with the VO.

Base VO Domain / OpVO Domain

Is the domain of the Virtual Organization (BVO or OpVO). It wants to provide Grid services whose main purpose is the support of secure and controlled coordination and collaboration of provided services.

Service Provider Domain

SPD is the domain of a service provider who offers services to the Grid.

AVP

Attribute-Value-Pair: Diameter messages are composed of multiple AVPs. An AVP is a tuple (Attribute-Name, Attribute-Value). (eg. {Input-Octets, 10000} or {CPU_Cycles, 200000}

A4C Session

An A4C Session is a sequence of messages exchanged between an A4C Client and an A4C server with respect to one of the A4Cs tasks. In the case of accounting an A4C session should be the mapping of a service session inside the A4C.

Authentication

Authentication is the process of verifying a user's identity. The authenticating entity verifies that the user is who he claims to be. The main result of the authentication process is the decision whether the user was successfully identified or nor.

Authorization

Authorization is the process of granting or denying access to a resource or a service. The decision of the granting or denying is based on the identity or some attributes of an individual.

Accounting

Accounting is the process of collecting data about resource (service) usage. This includes e.g. the amount of time spent in the network, the services accessed while there and the amount of data transferred during the session. Accounting data can be used for trend analysis, capacity planning, billing, auditing and cost allocation. Every service that requires accounting needs to implement a *meter* which measures the degree of service consumption. The *meter* acts also as an A4C Client and sends the collected data to an A4C Server which will store it as *accounting records*.

Charging

Charging is the process of transforming the technical values about resource consumption in monetary units.

A4C Registration

The A4C registration is the process of adding a user in the A4C database. The A4C registration for a user takes place in the home domain of that user.

Trust

A trust relationship between A4C servers means that the servers are configured to know each other (the servers can be authenticated) and accept A4C messages from each other (e.g. the authentication decision is accepted by the other server). The configuration is done by the network administrator. In case the servers are in different domains, the domains have a contract and administrative means to trust each other.

1. Executive Summary

This document describes the Akogrimo architecture after the first iteration process of the architecture definition phase. The architecture is expected to be refined during the following iteration loop.

1.1. Scope of this document

This document is intended as an update to the initial architecture deliverable: D3.1.1 Overall Architecture. It should be seen as additional to, rather than replacing, the original architecture document, and therefore does not repeat what was in that document unless repetition is needed to ensure the comprehensibility of this document.

Since D3.1.1 was delivered, further detailed design has been carried out within Workpackages 4.1 to 4.4. These workpackages address the four main Akogrimo layers, as set out in the original architectural concept:

1. Mobile Network
2. Mobile Network Middleware
3. Grid Services Infrastructure
4. Grid Application Support Services

Each workpackage has delivered a detailed architecture document focussed on its own layer, and is currently working on the first prototype implementation. While each layer Workpackage obviously addresses its interactions with other layers, Workpackage 5.1 is concerned with the integration of these layers and will be producing its own deliverable on architecture integration. This document therefore provides a summary of the separate single layer architecture documents and provides additional cross-layer information that will feed into the deliverable D5.1.1 Architecture Integration Report.

Where possible, this document takes into account the limited experiences gained from the ongoing prototype implementation. This, however, is at a very early stage. Full account of the implementation experiences for prototypes and testbeds, together with the results of the architecture evaluation and the overall Phase 1 evaluation, will be fed into deliverable D3.1.3 The Mobile Grid Reference Architecture.

1.2. Rationale and overall view

The primary objective of this document is to provide an overview and summary of the detailed layer architectures contained in the deliverables D4.1.1, D4.2.1, D4.3.1 and D4.4.1. The core of the document is thus Chapter 6, which summarizes the detailed architecture of each layer in four subsections. The rest of the document is not layer specific.

The secondary objective of this document is to highlight the novel features of the architecture, primarily arising from the focus on mobility of users and services. This starts in chapter 3 with an introduction to the main concepts introduced by Akogrimo. This is followed up in Chapter 4 where, after summarizing the other requirements that have influenced the Akogrimo architecture, the specific technologies and techniques used to address these concepts and requirements are described. The justification for the technologies taken, and the discussion of alternatives, is covered in deliverable D3.1.1.

Chapter 5 widens the scope to the full Akogrimo system, summarizing the overall architecture. This is covered more discursively in deliverable D3.1.1. In keeping with the high level overview nature of this chapter, subsection 5.3 provides a user-oriented view of the Akogrimo system.

Finally, Chapters 7 and 8 provide sample use cases and scenarios to demonstrate the operation and cross-layer interactions of the components described in Chapter 6.

1.3. Selected issues

Some concepts and terms need special consideration in the Akogrimo architecture (chapter 3). They are influenced by mobility considerations and cross the main layers. These definitions and outlines act as a preparation, where needed, for the description of technological choices in section 4.4 and for the rest of the document. However since this document is an update, the aim is not to include all that was introduced in the 1st version of the overall architecture (reference to D3.1.1), but instead to provide new information. Federated identity is discussed in relation to the Grid. Mobility is presented in terms of terminals/devices, users and sessions and later also in terms of services which in normal use may experience an SLA failure. Context changes and device capabilities are coordinated through a context management system and monitored by the application support layer. Policy Management provides a framework for variation of local behaviour in relation to security and SLA and other topics. The implications of dynamic and mobile working for business processing and workflow are introduced. In section 3.6, the necessary support for a service provided by all layers is described starting with the lowest layers; included here are QoS, service discovery, A4C, Grid resources and VO Management.

Requirements that influence the Akogrimo architecture are derived from the specific scenarios constructed earlier in the Project (eHealth and eLearning); from a business modelling point of view which assigns roles to providers and customers; and from the Akogrimo Business Advisory Committee (BAC) (sections 4.1, 4.2 and 4.3).

The Akogrimo architecture also has to take account of the current state of technologies and standards. Section 4.4 describes some of the choices made.

The opportunities and challenges for the full Akogrimo system (section 5.1.1) include user and identity management for a potentially large number of users; accounting and charging across multiple, time-partial suppliers in the value chain; device and context awareness by mobile Grid applications; and defining an appropriate model for identity and accounting which includes Grids applications with multimedia collaborations. After showing in 5.2 how the four layers interrelate, section 5.3 describes how application services may be integrated into the Akogrimo platform.

In chapter 6, the structure of each the layers is described in turn, but in each case leaving full details to the document devoted to that layer (D4.1.1, D4.2.1, D4.3.1 and D4.4.1).

The use cases in section 7 and the scenarios in section 8 are chosen for their cross layer implications. The scenarios in section 8 are detailed and where necessary can be viewed in an expanded form online.

2. Introduction

This document describes the Akogrimo architecture after the second iteration process of the architecture definition phase. The architecture is expected to be refined during the following iteration loop. The major task of WP3.1 is to enable the parallel development on different layers within Activity 4.

The Akogrimo project is driven by the basic idea that Next Generation Grids should be built on Next Generation Networks. This means that an Akogrimo NGG must be able to address the needs of an environment where users experience potentially fast changing context (Bandwidth, Device capabilities, Location, …), different access network providers and local services while aiming to participate in complex collaborations using resources provided by service providers from different organisations.

The basic concepts and terms required in addressing this challenging task to build an overall architecture for such a highly dynamic environment are defined in this document. The approach chosen is to define the concepts on a conceptual layer using UML class diagrams showing the connections and dependencies between them. Additionally in a more technical layer important aspects, such as different types of mobility, are described in greater detail.

A specific solution of the problem of how to allocate responsibilities to different components is to look at issues that span several layers. These cross layer issues are crucial for a successful integrated Grid middleware. The major elements identified are problems around identity and security management, cross organisational accounting, the handling of context and the interrelation of signalling as basic protocol element. This is reflected in selected key scenarios and user cases with are presented at the end of the document.

The document is organized as following:

After the introduction, Chapter 3 provides an refined overview about the Akogrimo key concepts to be used within the document - and of course within Akogrimo. These concepts are Identity, Domain and Profiling, which are fundamental in each commercial infrastructure. Additionally the Mobility concept, Context Management and Device Capabilities, Policies, Business Processes and Workflow as well as cross layer Service Level description and Service Specification are discussed.

Chapter 4 described the constraints under which the architecture was designed. This comprises first a summary of the requirements which came from different directions. These are the scenarios, the business modelling and the BAC, an addition to the original project plan. The Chapter concludes with the initial selection of technologies to be used within Akogrimo.

Chapter 5 introduces the high level architecture, which is fully based on D3.1.1. Here, next to the already known key building blocks and components– the IT platform – the overall service provisioning to third parties i.e. users of the Akogrimo product is defined. This provides a description of requirements any commercial entity using the Akogrimo platform has to fulfil.

Chapter 6 summarized how the abstract components and modules defined in chapter 5 are addressed within Activity 4. Each section corresponds to a Workpackage inside Activity 4.

In order to visualize and clarify how the detailed interworking between theses components, chapter 7 defines selected use cases and chapter 8 provides a MSC-like specification of selected key scenarios with a special focus on cross-layer issues, which have not already been handled within the Deliverables of Activity 4.

The document concludes with a summary and a conclusion given in chapter 9.

3. Concepts and Terms

3.1. Identity, Domain and User Profile

In the Deliverable 3.1.1 (PM 12), the term identity was described in a general and abstract way, followed by the concrete description used by Akogrimo. This chapter intends to explain further investigated concepts related to the identity in Akogrimo which will contribute to the innovation in the Grid environment within the identity management area.

The first efforts related to the identity management topic in the Internet were in the direction of the X.509 certificates. X.509 certificates usually fulfil better the requirements of a service's identity due to its static nature – the identity of a service does not normally change. However, X.509 certificates are not flexible, they are heavy weight and they can not be used to change identity according to the user's needs. Therefore, a more flexible and interoperable solution in the Internet world was needed. Within this context, the federated identity management appeared as a new initiative which intended to include the necessary flexibility and interoperability in the identity solution.

Federated Identity Management is a relatively new topic in the Internet. The first thoughts and ideas appear at the first two years of the 2000. Around the 2002-2003, several standards started (SAML v1.0) to consolidate the first ideas into the first standards related to this topic.

Nowadays, specifications and use cases of Identity Management can be found within the Web Services and plain Web-based environment. They provide several capabilities profitable both for users and service providers within this context.

Independent from this, the Grid initiative appeared as a solution for distributed environments with many benefits in the area of working groups with many resources in different organizations. However, the identity in Grid has been considered as something static, represented in the form of certificates, due to the much scientific nature of Grid. In this way, the user and or the organization presents one single certificate with the same identity – identifier and user profile – to all the different organizations and they map this user's identity to a local one representing the user in that local organization. The mapping is done with a manually configured Grid map-file.

Lately, there has been a closer cooperation between the Grid and the federated identity world. First, Globus Toolkit released a new version which could use SAML authorization assertions. This was a step forward. But the federated identity world makes use of SAML authentication and attribute assertions. With time, the Grid people realised that their certificates do not scale well and that many organizations already offer federated identities for their users, in business and in scientific environments. Federated identities offer benefits of dynamicity, security, automatic configuration…etc. From the federated identity system's point of view, they have also realised how important the Grid initiative is, and that they must provide special features not already thought as plain web-based services or stateless Web Services. Therefore, cooperation has been started between them in several areas and there are already research projects investigating how to merge both, like the GridShib project.

In Akogrimo, the contribution of the identity management for the Identity world and the problems and solutions we have achieved will be presented in this part of the architecture document.

The following subsections represent the architectural identity related concepts that Akogrimo project has deployed, based on the entities and their required relationships.

3.1.1. Domain and Trust in Akogrimo

In an Akogrimo scenario, there are several logical entities that need to be present: a user, a customer and a VO service. All of these entities are always related to an organization that represents them, called the Home Domain. Home Domain is a relative term, always bound to a user, a customer or a service. In the case of a user, the term Home Domain for Akogrimo is equivalent as the term Identity Provider Domain within the Federated Identity Management Glossary. (For further information, please see Definitions section). The Home Domain in Akogrimo manages users' information and has a contractual relationship to them. In Akogrimo, any type of organization (not only Network Operator) can be a Home Domain for a user.

From an identity point of view, Akogrimo is based on an identity information exchange between the involved entities. The entities – user, customer, service – can have the same Home Domain or different ones. However, the domains of the different entities need to establish some kind of relationships if they want to share information.

In order to set up an Akogrimo VO, there are several pre-requisites in terms of trust relationships between domains that need to be assured. These are the following:

- One domain or organization can represent several users, services or customers. It must at least represent one entity.
- All entities that want to form a VO need to be bound at least to one domain. That is, each entity has signed at least one contract with a specific organization. This organization represents the specified entity at others organizations and manages the entity's information.
- One user can have several Home Domains. However, at one time, the user can only be logged in at one Home Domain. There is no relationship between different Home Domains of the user. Their diverse information on this one user is managed independently and can not be linked.
- Each domain that represents a Grid service of an Akogrimo VO needs to have trust relationship established with the Base VO Domain.
- The customer's domain and the Home Domain of the user need to have trust relationships established.
- The customer's domain and the Base VO Domain need to have trust relationships established. The customer domain represents the user at the VO Domain.

The points explained above need to be applied before setting up the VO, in order to achieve the security requirements of Akogrimo.

The picture below represents an example of an Akogrimo VO with the required entities and the trust established following the requirements. In the picture, the user and the Base VO do not need to have a trust relationship. This is possible for Akogrimo, since it is the customer the one who acts as an intermediate. The user trusts the customer by delegating his user information and the Base VO trust the information presented by the customer. Additionally, there is no need of trust establishment between the domains of the Grid services and the domains of the user and/or customer. The Base VO domain acts as intermediary between the Grid service domain and the customer domain.

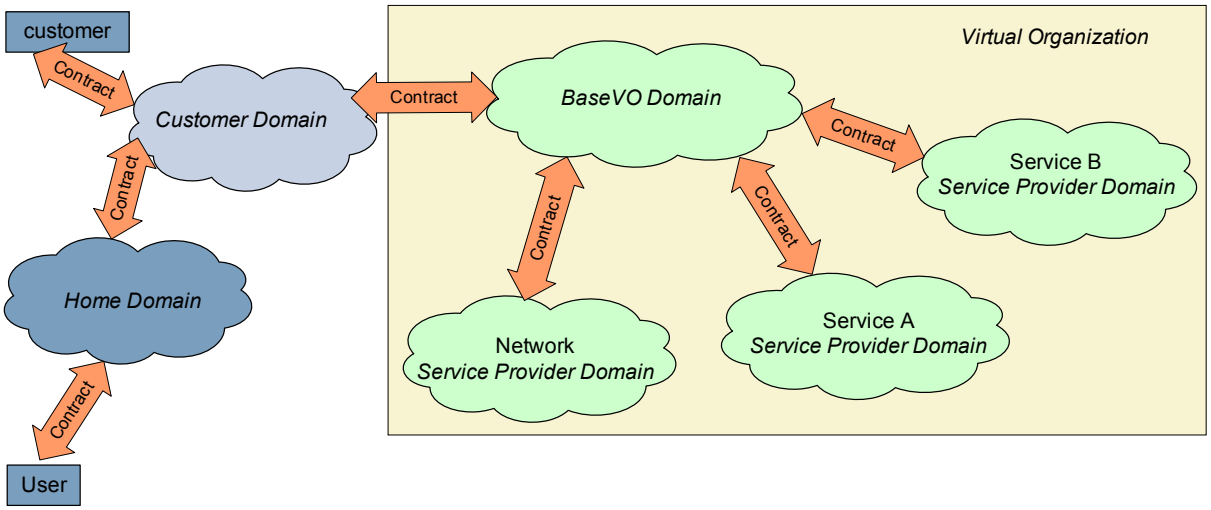

Figure 1: Akogrimo example of trust establishment

3.1.2. Personalization: User Profiles and Identities

The user consists of a set of characteristics that are required to be at the Home Domain. Some of these attributes are considered of a generic nature and some of them are specific or relevant for the Home Domain. All attributes at the Home Domain represent the user profile.

In the normal life, the user identity which is usually presented depends on the type of organization. For instance, the user profile that an employee presents at work -with attributes like social security number, Health insurance number among others- is different to the profile that a user presents, for example, at a travel agency – with attributes like interesting countries, type of trip…etc.

It is desired in Akogrimo that the user can choose the attributes he wants to share within a Virtual Organization. Depending on several factors like the content, the purpose and the nature of the VO, he will want to customize the set of attributes that form his user profile. Some of the attributes could already be stored at the Home Domain and some others could be specific. In Akogrimo, the personalization will be done by the user, at the Base VO registration in a very straightforward way. He will be able to select the attributes that are already at the Home Domain and include other ones at that phase.

In this case, it is considered that the user can have several identity profiles or digital identities. It is a matter for the user if he wants to use the same profile/identity for several VOs.

3.1.3. Privacy: Anonymity and Pseudonymity

Privacy is something very important for users when doing transactions in different administrative domains. There are several reasons why the user wants to be anonymous. Anonymity of a user is the maximum grade of privacy. They may depend basically on the type of activity and the content of that activity.

The Akogrimo user would like to have the most grade of privacy depending on the nature and the services and domains that belong to the VO. In a Grid environment, several tasks or jobs may require the work of different services based on a user need. When they are invoked, the services must know that they are really working for the same user. Anonymity –or an anonymous identifier- can not be used, since there will be no possibility of linkage and the services could not

relate their work. This is the case, for instance, in the e-Health scenario when the data manager needs to know at which data store an identifier belongs to.

Although global anonymity is not possible within the Grid concept of Akogrimo, the most secure possible combination has been achieved. This can be done with the use of pseudonymous identifiers. A pseudonym is a privacy-preserving name identifier used to identify a user in an environment. Akogrimo proposes the creation of multiple pseudonymous identifiers for a user, given multiple BaseVOs related to the user. That is, a pseudonymous identifier represents uniquely, but in a privacy preserving fashon, the identity of the user at the VO. Outside the VO, the identifier does not mean anything. Only at the Home Domain can the real identity of the user be linked with that identifier.

At the creation of the BaseVO, the new pseudonymous identifiers are created. This identifier is related to a certain trusted domain for the VO. That is something like "uniqename@domain". The domain of the identifier is the domain of the Customer who sets up the VO. This domain needs to have trust relationships with the VO and with the user's Home Domain and therefore it is the one who is involved in generating the new identifier and map it to the identifier received by the user's Home Domain.

In this sense, the private use of the pseudonymous identifiers implies that only services at the same VO can link the identity of a user. More over, the same service used for two different VOs can not link the profile of the user at one VO with the profile of the user at the other VO. There will be two different identities.

3.1.4. Single Sign On with multiple identities

Although the user has multiple identities, for different VOs and domains, the ability of Single Sign On is still possible in a very smart and secure way. The user receives the IDToken D4.2.2 [D4.2.2] –just one- which is independent of all different identities the user is related to, when doing the first login. After that, it will be up to the user to select the pseudonymous identifier chosen for that VO service. This can be done by himself or maybe with the help of an identity selector that automatically selects the identifier.

3.2. Mobility Concepts

One of the main objectives of Akogrimo is to allow mobile users to not only use the Akogrimo network, but to allow the effective use of those users devices and resources as part of the whole Akogrimo network. Mobility presents several challenges; a breakdown of the types of mobility involved follows.

3.2.1. Terminal Mobility

Terminal mobility, as the name implies, relates to the mobility of the device (or terminal) the user utilizes for accessing the network. In order to have effective terminal mobility, the communications of a given user must not be disrupted by movements of that user which might provoke network topology changes.

Topology changes can occur when users move from one access network to a new access network. Without terminal mobility support, the change from one access network to the other will cause the terminal to lose its connection with the old access network, acquire a new connection in the new access network and a new IP address. In practice, this causes network connections to be stopped. They then have to be restarted, with a new IP address, which causes problems for most of the software that uses the network.

A solution for this problem is the Mobile IPv6 protocol. Mobile IPv6 is further discussed in section 6.1.1.

3.2.2. User Mobility

User mobility relates to the capability of the user to access personalised services independently of the terminal device he or she is using. It is provided by a user-oriented security and authentication framework. The user has to perform his/her registration in the network – and in the Grid infrastructure – before using the network services. This registration process associates the user with the terminal.

3.2.3. Session Mobility

Session mobility enables the transfer of application sessions between different devices without interruption. This is achieved with the SIP protocol. SIP can be used both by the user, and by the Grid infrastructure, to redirect communications (e.g. image display) to different devices, retaining the user association mentioned above.

3.3. Context and Device Capabilities

In the literature many definitions and examples of context information are found. Dey [Dey01] reviewed previous context definitions, and provided the following general definition of context: "Context is any information that can be used to characterise the situation of an entity".

Context awareness means that devices, applications and systems have information about the circumstances under which they are operating and can adapt their behaviour to be optimal for the current situation. This increases system usability tremendous by reducing the demands on the end-user and minimizing the need for user attention.

The vision of Akogrimo involves context-aware mobile Grid services, allowing the user to solve his/her tasks while devices and technology fade away into the background. Akogrimo focuses on context that describes the situation of a mobile user. While much prior Grid research has focused on batch-mode supercomputing applications, Akogrimo introduces the mobile Grid, where interactive services involving mobile and nomadic users are of key importance. The lives of humans are much more varied and dynamic than those of computational resources, so when humans are introduced as resources in the Grid, there is a need to keep track of their context. By knowing this context, the system may easier choose the right people to participate in a workflow, and better adapt service behaviour to the situation of those people. Context has many facets:

- Presence: Is the user logged on? Is he idle or busy?
- Physical properties: User location, local time, body temperature etc.
- Device context: What terminals and other I/O units are available to the user and what are the hardware and software capabilities of those devices?
- Local services: What services are found in the proximity of the user?

The definition of context is open-ended; the set of data that should be monitored will depend on the application domain. Akogrimo shall therefore provide a context infrastructure which allows relevant parties to keep track of user context in terms of (SIP-) presence, local services in user's proximity, user location and device capabilities for user terminals.

Obviously, it is not desirable that every application implements a context engine to gather and process specific context information; rather a generic context infrastructure that supports applications by gathering, processing and managing basic context information is an essential component of the Akogrimo infrastructure.

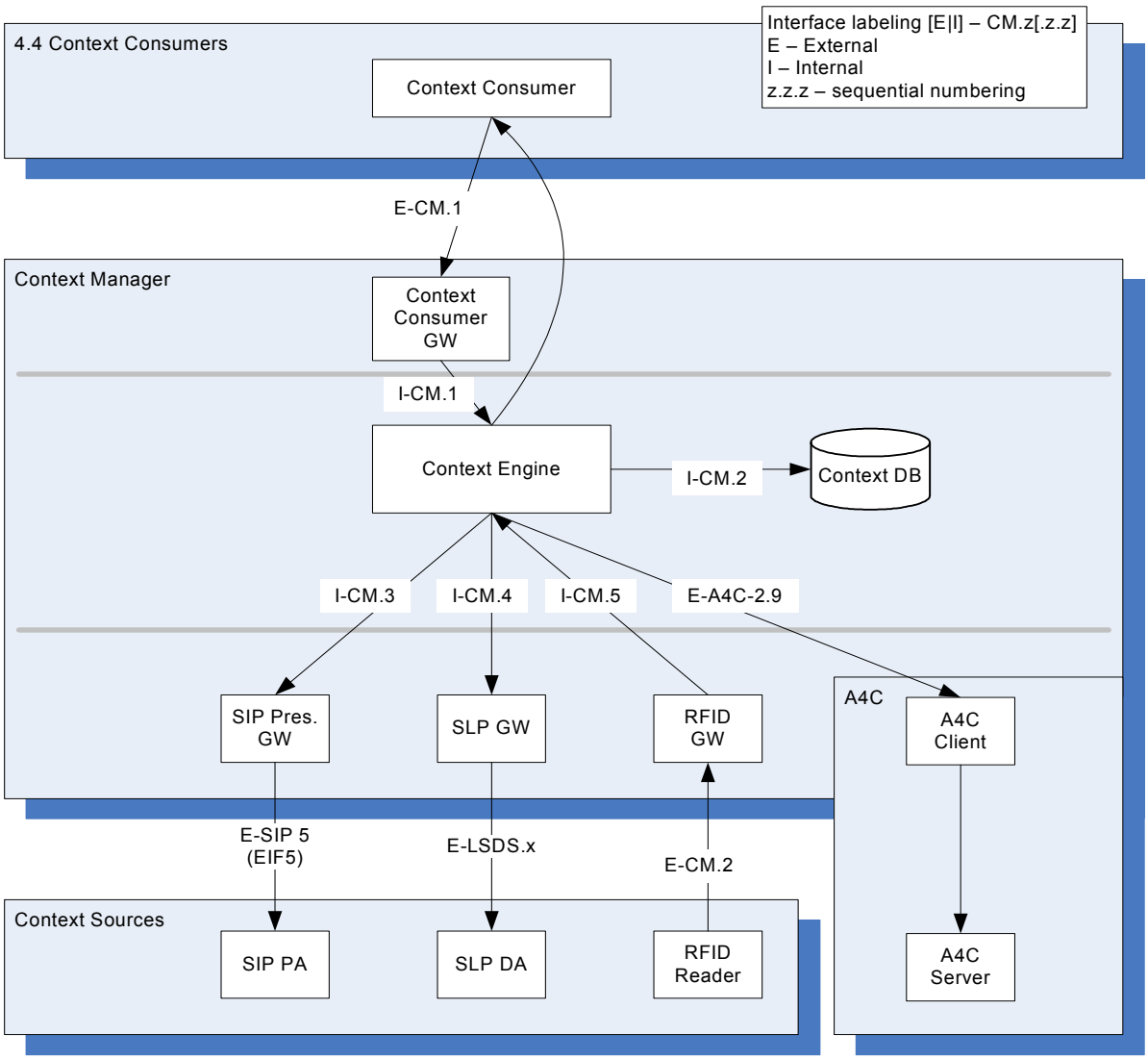

Figure 2: Structure of Akogrimo Context Awareness system

Context data, obtained from different context sources, are distributed to the interested parties (context consumers) through a mediator entity, the Context Manager. This element filters relevant data and converts it to a uniform format prior to delivering it to the interested destination. Additionally it has to solve the possible inconsistencies and infer high level context data from basic context information.

Context consumers can either execute queries towards the Context Manager to get specific context information or subscribe for notifications about changes in certain context data (using a suitable semantic language, e.g. RDF, OWL). The Context Manager is responsible for distributing context data to context consumers accordingly.

To handle domain specific context inference, the Context Manager should offer interested parties interfacing with extension modules. One example is the mapping of location coordinates to a map of a building, such that the context will also specify which room a person is in, and what physical facilities.

For more comprehensive discussion of context and device capabilities, see D2.2.4 [D2.2.4] and D4.2.1 [D4.2.1] and D4.4.2 [D4.2.2].

3.4. Policies

Policy is a very general term, capturing the separation of the information used to come to a decision from the decision making itself. In this context, policy is describing "how use available resources" and not merely things like "permit/deny access to available resource" (authorization or access control problem). In our context, available resources are virtualized and accessed (directly or indirectly) through a Grid service interface. Therefore, policies can provide the necessary flexibility for the resource management process, which occurs at several levels of abstraction.

Within distributed computing frameworks, policy has to take into account the behaviour of the individual devices that make up the collective entity, along with the behaviour of the collectivised applications using the framework. Policy is therefore required at each level to support this behavioural management. Within Akogrimo this challenge is made richer and more complex with the introduction of mobility and the management of events at the network level which directly affect the availability and virtualization of resources as seen in the higher application layer.

Within Akogrimo participants exist as either service providers or service consumers. The representation of these entities is achieved via the use of agents, which are central to the management of devices within the framework. These agents act as a filter and also an integration layer between the behaviour of a device manifesting a user or service and the central Akogrimo framework. Thus the design of the agents and the potential interactions on offer within the central core of an Akogrimo VO can be seen as a first point of application of policy design and enforcement. This is an example of centralised control within Akogrimo, and is seen in policies such as the membership policy of the Base VO.

Control can also be achieved by altering specific service behaviour in the central Akogrimo architecture. This is achieved by applying policy defined in other core services within Akogrimo, one example being the Service Level Agreements (SLA) mechanism. SLA's in Akogrimo express and encourage specific behaviour of devices at a local level (that is local to the device) when collaborating as a node of a larger application. For example SLA's in relation to bandwidth could help satisfy a higher level central policy in a framework where connection speeds between devices has to be kept at a certain level. The use of SLA will also at the same time encourage local device managers to comply with central rules in order to avoid penalty clauses in the SLA.

More details on the various policies within Akogrimo and the Policy Framework being developed are provided in section 5.9 of Akogrimo Deliverable D3.1.1.

3.5. Business Process and Workflow

The Akogrimo architecture is intended to support business process designs that both support and take advantage of dynamic, mobile services and users.

It is important to distinguish between the terms "business process", "workflow", "choreography" and "orchestration". By a "business process" we mean a high-level description of a process in terms that are meaningful at a business level (as opposed to a computational or engineering level). Such a description should be abstracted from any particular implementation of the process, and should be more concerned with requirements and goals than specific execution approaches. At the opposite extreme, a "workflow" is a precise definition that is (or can be easily converted into)

an executable form. The basic steps of a workflow are normally service invocations; though even here the workflow itself is not concerned with how the service is implemented (though it does have to depend on and make assumptions about the service's behaviour).

"Choreography" refers to a form of process control where there is no single centralised controller, but where the details of execution of a process are delegated to individual execution sites. A choreography may describe how the different parts interact, but will almost certainly not describe or control how each part performs (only that it produces the expected interactions at the expected times). In "orchestration" on the other hand, the behaviour of a process is under control of a single central agency which controls the actions of each part and mediates in (or at least knows about) all interactions between them.

To some extent, the distinction between choreography and orchestration is a matter of detail. For example, an ActiveBPEL engine running a workflow script "by itself" is acting as an orchestrator; but even so, it does not (and cannot) define or control the implementations of the services that are used in the script. In the other direction, the script may be being executed as part of a larger choreography that links other workflows. In Akogrimo, the need for a "big picture" view capable of assessing and handling context changes lead towards an orchestration solution, while the dynamic nature of most of the applications tends to support devolution of responsibility to choreographed services (ie they know what they need to achieve and just get on and deliver it)

Akogrimo intends to cater for the mobility of participants (both users/clients and services) in a business process. One consequence of this is that the business processing components must "track" users and services as they change location while retaining their identity, but must also support the ability of the process to adapt to changes in context of such mobile agents, for examples, changes in their capabilities, discovery of alternative services, and responding to situations where an agent becomes disconnected. Of course, some of this adaptation can be delegated to the services (particularly the "how" to adapt in order to still deliver in the face of a context change) but ultimately any change that may require a change in the overall business process must be handled at the orchestration level (the "what" to adapt in order to satisfy the possibly changed business objectives). For example, the availability of a high definition screen in an eHealth scenario may mean the presentation of some medical data is changed (graphs instead of text), or it may mean that the pattern of interaction with the user is changed (patient can be treated on the spot rather than being immediately rushed to hospital). One immediate consequence of this is that the business process enactment sub-system needs to have access to the context information associated with all its users/clients and services.

The final requirement generated by Akogrimo's mobile and dynamic nature is the need to build on-the-fly secure Virtual Organizations, where the data can be shared among the dynamically changing members but prevented from falling into the hands of outsiders. The eHealth scenario clearly demands a high level of security. The security is implemented using security tokens underwritten by the Akogrimo A4C servers. The VO security stops at the User and Service Agents, which act as proxies for the user and service; the security of the links beyond these agents is the responsibility of the respective user/service provider. Within the VO, security is the responsibility of the VO manager, which can dynamically alter the internal security to reflect the state and requirements of the workflows.

3.6. Service Level Description and Service Specification

3.6.1. Network Layer services

The Akogrimo project aims for an integration of the worlds of network and Grids, which presents some difficulties. The network layer typically uses a variety of protocols; which have to perform such that they meet the strong time-constraints. This performance comes at the cost of interoperability. When there is a need for interaction between different systems that don't have a common language, additional effort creating some "translation layer" or similar is required. In the Grid world, however, interoperability is the main advantage. By making extensive use of Web Service based technologies, a common base is provided that makes it inherently easier to support different interactions between systems, should the need arise.

In the second phase, Akogrimo's goal is to achieve a more complete integration of network and Grids, making the network and its resources available as native Grid services. For the first phase, however, only a partial integration will be achieved. Although the Akogrimo architecture is structured following a layered model, the necessary support from network layer to higher layers requires cross-layer interfaces for important services, such as SIP and QoS. These cross-layer interfaces will be provided using the same Web Services protocols typically used by Grids.

Taking into account the scenarios specified in D2.3.1, we can see that multimedia calls are important for Akogrimo. Initiating and managing multimedia calls is supported by the SIP protocol. On the other hand, for those calls to be practical, it is necessary that the network can provide the required bandwidth at all times.

An interface for SIP 3PCC (Third Party Call Control) will be provided. This will be useful for workflows where Grid components need to initiate SIP calls between two other parties

The QoS Broker, which is the component which manages network quality of service, will have an interface designed specifically for interaction with the EMS. This interface is based on Web Services and OGSA standards. As a consequence, network services become part of the workflow, and are defined both according to the user profile (and subscribed services) and according to its current operations (e.g. emergency life support overrides contractual user capabilities). Network QoS is further detailed in chapter 6.1.3 and 6.1.4 as well as Akogrimo deliverable D4.1.1 – Network Layer Architecture.

For implementation and scalability reasons, the network supports well defined QoS bundles, strongly influenced by the existing models for mobile technologies (UMTS). Each of the three defined bundles is designed for a specific usage profile, audio, video and data. A QoS Bundle is comprised of several well defined services, which the user may choose from when using the Akogrimo network. Table 1 presents these bundles.

In The following, a brief classification of traffic classes is given:

Signalling: This traffic is of the highest priority and time-critical, but has a low bandwidth requirement.

Interactive real-time: Time-critical traffic typically for interactive multimedia applications which are sensitive to delays and out-of-order packets.

Priority: Not time-critical, but important, such as multimedia streaming, or some Grid application data exchange. Higher priority than Data Transfer, but lower bandwidth typically.

Data Transfer: Not time-critical but may be loss-sensitive.

Best Effort: As the name implies this service offers best effort. Its efficacy is highly dependant on network conditions. This is basically what Internet provides.

Type of Service	Bundle 1 Mixed audio + data [kbyte/s]	Bundle 2 High data + video [kbyte/s]	Bundle 3 Mostly voice [kbyte/s]
Interactive	10	20	10
Data	100	1000	1
Priority	1	200	1
Signalling	1	1	0
Best Effort	250	0	250

Table 1: QoS Bundles

3.6.2. Network Middleware services

On top of the basic network functionality is placed network middleware service provisioning. Here, especially network session based Signalling such as SIP-based sessions, but also AAAC based sessions are grouped to the lower network related services.

The SIP and QoS cross-layer interfaces allow interoperability between the Grid and the network layer. This interoperability is somewhat limited as of now, but in the second phase of the project, a more consistent integration of those layers will be achieved.

The network middleware layer provides a set of basic infrastructure functions to the higher layers, including cross-layer A4C, presence and context management, and semantic service discovery. The corresponding traffic is mainly signalling sessions between components in the core network (or wired part of the access network). The QoS cross-layer interfaces offered by the network layer do not apply to core components, that are assumed to be well connected at all times. However, there are some network middleware sessions involving the mobile terminal, e.g. SIP presence publishing that can benefit from the QoS interface offered by the network layer.

On top of this network session related part Grid infrastructure related elements are added to the overall cross-layer service bundles. Here, the Grid Service Discovery System (GrSDS), provided by the Akogrimo middleware layer, is essential for interoperability and cross-layer interactions. Service providers, of e.g. the cross-layer network layer services, will register their services with the Grid Service Discovery System. The GrSDS represents a "static" discovery registry with semantics capabilities where the static description of services (i.e. WSDL description and corresponding SLA template) is stored. When an entity needs a certain service, e.g. a service that virtualizes network layer functionality, it will query the GrSDS for a list of relevant services, select and invoke the most appropriate. However, dynamic service attributes, e.g. processor load, number of instances running, queued jobs, network load etc., are not managed by the GrSDS. Details on the network middleware layer are found in Section 6.2, as well as D4.2.1 [D4.2.1] and D4.4.2 [D4.2.2].

3.6.3. Network Middleware services

In order to finally come from Grid resource to user-centric service, in the Akogrimo architecture the Grid functionality of the whole infrastructure is provided by the "Grid infrastructure" level. This level consists of the Web Services Resource Framework (WSRF) level and the OGSA services layers, being placed in the middle of the Akogrimo architecture participating as the "Grid glue" to the functionality of the Akogrimo system.

All Grid resources, both logical and physical, are modelled as services on the basis of Web Service implementations. However, for the purposes of the resources modelling in the Grid the Web Service interfaces must frequently allow for the manipulation of state, that is, data values that persist across and evolve as a result of Web Service interactions. To achieve this, the Web Services Resource Framework (WSRF) defines a family of specifications for accessing stateful resources using Web Services. Note that communication services are also described using this framework. The Web Services layer, together with the WSRF, provides a base infrastructure for the OGSA architected services layer providing the overall Grid management functionality.

The basic motivation is to provide OGSA specific services implemented through the framework of the WSRF. The basic functionality that must be supported from the Grid Infrastructure Services layer, as identified by the consortium and compliant with the OGSA draft specification, can be categorized in the following:

Execution Management Services (EMS): This category of services comprises all the functionality that is concerned with the problems of instantiating and managing tasks, such as assigning jobs to resources, creating an execution plan, balancing the workload, optimizing the performance, and replicating jobs to provide fault tolerance. Conceptually, these resources can be represented by the composed bundles ad indicated above.

Data Management: This category comprises all the functionality that is concerned with the access to and movement of large data sets, as well as data sharing, replicating and archiving of data.

Monitoring: This category comprises the services that are focusing on monitoring and managing of the Web Services within the layer.

Service Level Agreement (SLA): Services related to the enforcement of the SLA contractual terms that especially influence the execution of jobs within the layer.

Policy management: This category of services comprises the functionality concerned with the management of rules and the policies which apply in the execution of services within the Akogrimo architecture.

Security: This category comprises the services that are concerned with the security issues of the specific layer. It comprises the services that will deal with the confidentiality of the communications and the authorization for execution within the system.

The WSRF solves the problem of statefulness in the following way: it keeps the Web Service and the state information completely separate. Each resource has a unique key, so whenever we want a stateful interaction with a Web Service we simply have to instruct the Web Service to use a particular resource. The motivation for these new specifications is that while Web Service implementations typically do not maintain state information during their interactions, their interfaces must frequently allow for the manipulation of state, that is, data values that persist across and evolve as a result of Web Service interactions.

Example of e-Health stateful resource

A pairing of a Web Service with a resource is called a WS-Resource. The address of a particular WS-Resource is called an endpoint reference (in WS-Addressing terminology). In figure 3 we

present an example of how to build stateful resources for the e-Health application. The execution of specific services (like analyzing medical data for a patient) is managed through the EMS component (which is part of an OGSA enabled architecture). The e-Health Web Service keeps the state of each patient and interacts with the resources (which in this case is their medical data like heart pulse monitoring) and returns the result of the operation to the EMS (or alternatively to any other component) for post processing.

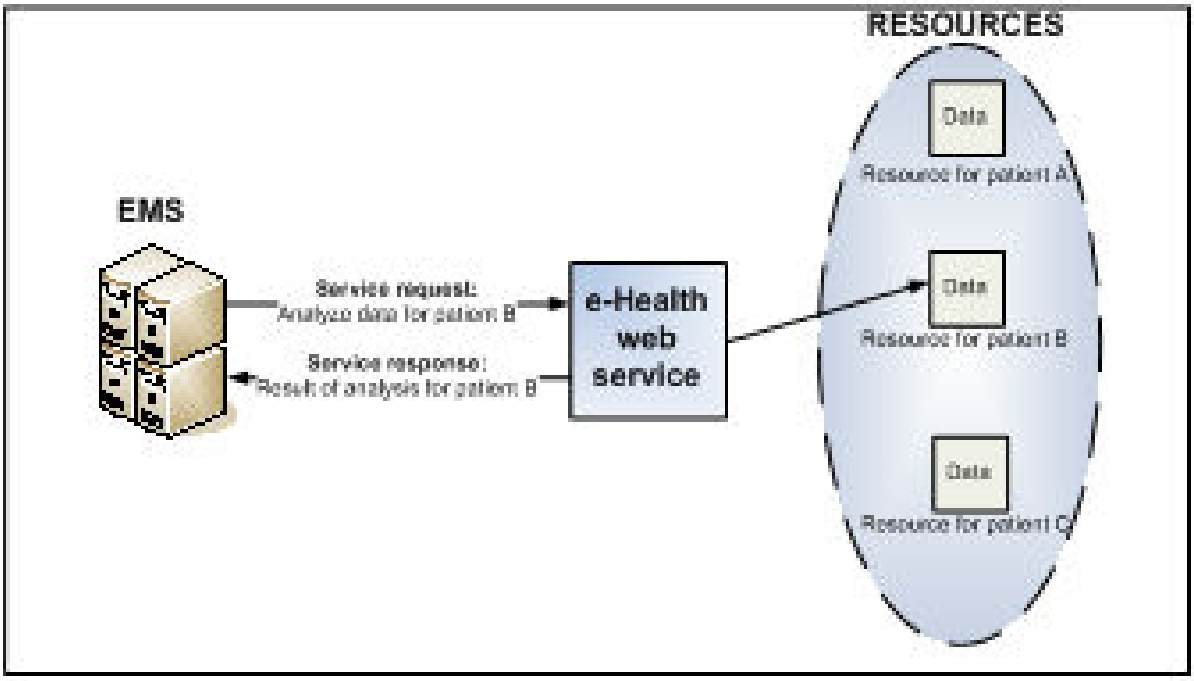

Figure 3: Example of stateful resources for the e-Health application

3.6.4. Grid Application Support services

The upper layer (Grid Application Support Services layer- GASS) leverages on the above infrastructure in order to provide capabilities closer to the viewpoint of the applications. This means that GASS layer provide a set of services that are directly visible on top of Akogrimo and they are grouped in the following categories:

Virtual Organization management: this group includes a set of services that allow the creation of Virtual Organization where different entities can share their resources (virtualized as services). The goal is to guarantee a controlled resource sharing in order to enable effective business relationships between virtual organization participants.

High Level Service Level Agreement: this category of services supports the Virtual Organization creation by providing functionalities to manage the definition of a business agreement between service providers and their customers.

Business Process Enactment: this group includes services that provide orchestration of the services coming from different providers and hosted in different administrative domains. The execution of workflow will be aware of user context. This is of particular interest when we deal with mobile user because on the basis of context changes it may be necessary to take corrective actions during the workflow execution.

The interactions between these services creates an environment where the applications are executed and, in particular, services from different providers are orchestrated, to implement the

logic of the front-end application. The main issue is to support mobility and network awareness exploiting in a transparent way the lower layers of the architecture.

In fact, in order to make effective these high level capabilities, the GASS layer uses functionalities coming from the lower layers but which are not actually handled by the GASS layer directly (i.e. using specific protocols or having a low level knowledge of the technology providing the particular functionality). Every function is provided by a service that virtualizes the resource in charge of that function. Of course, the term service is intended in the sense explained above when discussing the WS-Resource concept.

We can say that the lower level functionalities are available at the GASS level just like any other WS-Resource. The details are transparent to GASS layer itself, which deals with them like any other WS-Resource exploiting the low level Grid management functions provided by the "Grid glue" infrastructure. (See Figure 4 below.)

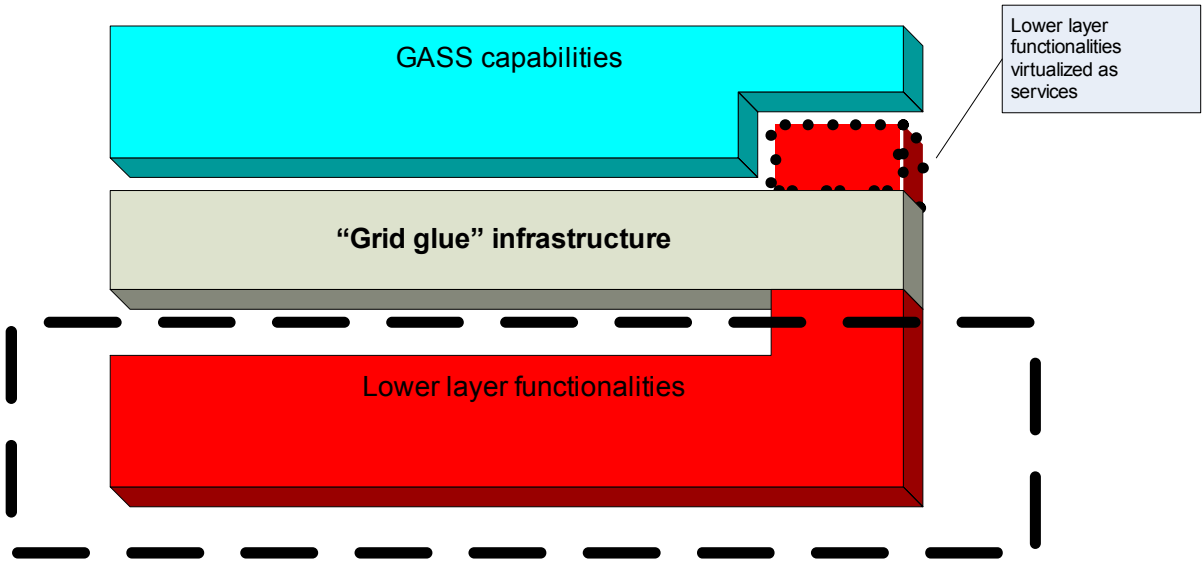

Figure 4: Integration between GASS and lower layers

4. Derived Requirements and Technological Context

4.1. Requirements from the Scenarios

The requirements from the scenarios influencing the architecture are the following:

- Provision of seamless mobility across technologies and operators participating in a VO: As a consequence of this, a multi-media (UDP) session should continue without interruption in the case of a handover between network technologies. In case of a handover between operators a slight interruption might not be avoidable.
- Support of disconnected operations:

  In case a member of a VO is temporarily disconnected from the network (and thus from the VO), after automatically reconnecting (using another operator/network technology/terminal) he can continue working in the VO. The period of disconnection can be in the rage of seconds until hours (reconnecting the system after leaving the plane following a transatlantic flight)
- Support of mobile dynamic VOs:

  This means that the management of the VO presumes that the members of the VO dynamically change their point of attachment, the connected operator and /or the terminal in any combination, dynamicity and without any notification. The services and workflow accordingly adapt the current stage of execution according to the specific context in a pervasive environment

4.2. Requirements from the Business Modelling

The requirements arising from the business modelling are the following:

Akogrimo is seen as telecom operator oriented service platform which allows the dynamic composition of service bundles offered to customers, realized in a *mobile dynamic virtual organization* concept. This requires basically that each member of a VO providing a fragment of a service to a composed service (an application, from a user point of view) can offer this service directly to the user of an intermediate company who bundles a service or who sells this service directly to the user.

This leady to two basic processes: BtoB and BtoC

The Akogrimo platform supports both of these processes in a way that each actor can contribute his part based on a reliable agreement between seller and vendor.

4.3. Requirements derived from the BAC

Some requirements or suggestions from the Akogrimo Business Advisory Committee (BAC) related to Akogrimo in general, are the following:

- **Accounting/Billing**

The Akogrimo platform should be flexible regarding accounting mechanisms. This means the platform should not force a specific way of accounting, for example, to force accounting by time or packages.

- **Open Platform**

  Another requirement is the accessibility of the platform. This issue is already satisfied by Akogrimo (e.g. providing 3rd parties access to accounting, positioning APIs).

  An open platform will enable the communication among different distributed environments and with different management systems. This requirement will enable solutions from several providers to be used over the same Grid infrastructure.

- **Computational Resources Virtualization**: Due to the virtualization of the computational resource some companies could improve their Data Centre resources and provisioning for the different services offered to end users.

  A better use of CPU resources through creation of virtual servers and improved storage capacity thanks to a virtualization layer that allow the storage space distribution and management in a more efficient way, could reduce resource over-provisioning

  The resource virtualization is perhaps the feature that IT professionals associate most with Grid technology. This kind of virtualization software is currently been tested inside the companies to evaluate cost and resource improvements for its infrastructures.

  This technology enables companies to become more dynamic, adjusting its resources to the business necessities in an easy way.

- **Sales forces language**: this is a very important issue, translating technologies to business benefits, that is, delivering arguments that sales forces that can present to the clients, in order to sell it better.

- **BAC suggestions regarding White Papers**

  - the business relevance is explained well
  - focus slightly more on the realization of technological challenges in the scenarios, or at least refer to further documents where the details can be found
  - add Grid technology features relevant to the network operators and visa versa to the invention list

The requirements derived from the BAC at architecture level are the following:

- **Security**: from the access point of view (Authentication and Authorization of the users), this is an important requirement for all commercial applications, especially at the Grid level (in order to avoid incorrectly using the available Grid services).

- **Service Level Agreement (SLA)**: another important requirement is the fulfillment of quality of service, the conditions agreed between consumer and Service Providers, and between Service Providers themselves and the Grid infrastructure.

  In an environment where the final service depends on the composition of several services, to assure the QoS of the final services requires assuring the QoS in the Grid infrastructure.

- **Trusted Third Parties (TTP)**: The role of TTP should be considered. Which roles are possible (e.g. network operators may prefer to open their platforms to TTP that can do the accounting for e.g. aggregated services)

The requirements derived from the BAC for the transition phase to a Mobile Grid platform:

- **Migration Path**: Outline the path on how to migrate from existing telecommunication and Grid infrastructures to the Akogrimo Mobile Grid platform. Which components of the Akogrimo platform can be introduced independent from the whole platform to support an incremental migration?
- **Existing APIs**: Explain how the Akogrimo platform deals with existing APIs of the network provider/operators. Which APIs do the network provider have to open in order to support Mobile Grid platforms? Which APIs does Akogrimo offers, to be provided by the network and future Grid operator to its customers?

4.4. Selected Technologies

4.4.1. Profile and Identity

The evolution of the Grid community and the federated identity community is already mentioned in section 3.1 of this document. The main concepts of the identity model in Akogrimo have been also explained in that section.

In order to develop the identity management of Akogrimo, there is a clear influence from the de facto standard in the federated identity world, SAML. Also, Akogrimo has a good relationship with the Daidalos identity project and there is a strong similarity in both models in terms of the single sign on (SSO) concept and in terms of the basic components- network environment, services and users.

However, there are also significant differences in the Akogrimo model as it is not able to just apply one model with no other considerations. Akogrimo contains much functionality and many peculiarities that have not beed taken into account by other identity models, basically due to Akogrimo's integration of the network and service layer and the appearance of many different administrative domains.

Other models were also studied, like the Liberty Alliance, Shibboleth and GridShib. The first one is not sufficient due to the restrictions implied by the project, in the sense of design, with too many licences to check and insufficient implementation information available unless you buy a license from a commercial company. The Shibboleth and specially the GridShib projects are very interesting, with some results that go in the same direction as Akogrimo but they do not fit particularly well in Akogrimo due to restrictions such as the implication of a trust relationship between all domains that are involved and the release of the Grid roles by the user.

4.4.2. Mobility and IMS

IP Multimedia Subsystem (IMS) is a next generation networking architecture originally defined by the 3GPP for 3G mobile phone systems in UMTS networks. While originally designed for mobile networks, it has support for fixed networks as of Release 7.

IMS is independent of the access network used, supports different network technologies, provides roaming and user mobility. IMS was also designed to allow the offering of any kind of IP-based service, and is in fact strongly oriented towards multimedia services.

The IMS core network is composed of different functions. A function is not a node or hardware box; two different functions can be combined in one node. Likewise, one function can be split across several nodes, or be present multiple times in the network. The most relevant functions are:

- HSS: Home Subscriber Server. Holds the master database holding user information, performs authentication and authorisation of users, and provides information about the physical location of the users.
- P-CSCF: Proxy-Call/Session Control Function. A SIP proxy that is the first point of contact for the user terminal. The P-CSCF enables the coordination between events in the application layer and resource management in the IP bearer layer, acting as a Policy Decision Point
- I-CSCF: Interrogating-CSCF. Is a SIP proxy located at the edges of the administrative domain which acts as the contact point for all IMS connections destined to a subscriber of a network operator or to a roaming subscriber currently located in that operator's service area.
- S-CSCF: Serving-CSCF. It's a SIP server that handles the session states in the network, managing ongoing sessions and providing accounting mechanisms.
- GGSN: Gateway GPRS support node. Connects the radio network and the IP network in the UMTS scenario and maps the radio layer QoS to the IP layer QoS. Other scenarios require other nodes with similar capabilities.

The Akogrimo project follows an all-IP approach, with IPv6 acting as a convergence layer between the core network and any access technologies an operator might want to use.

The main Akogrimo components from a network point of view are:

- A4C Server: acts as a Policy Information Point (PIP). Its functionality resembles the combination of the HSS and I-CSCF.
- QoS Broker: acts as a Policy Decision Point (PDP). Some of its functionality resembles the combination of P-CSCF and S-CSCF.
- Access Router: acts as a Policy Enforcement Points(PEP). Part of its functionality is close to the GGSN.

Multimedia service provisioning is supported by a SIP Server, and different services could be provided by different entities should the need arise.

In conclusion, Akogrimo's all-IP approach defines a universal architecture for heterogeneous environments capable of supporting any kind of service. By having separate PDP and service provisioning servers, the Akogrimo network provides the ability of supporting any application signalling protocol, thus not being tied to a single protocol (SIP) as is the case of IMS. This approach allows a simpler and more flexible architecture which is suitable for any kind of service provisioning, whereas IMS is optimized for multimedia services.

4.4.3. Context and Device Capabilities

This section summarises technological choices related to context and device capabilities. A detailed discussion and rationale for choices made is found in D2.2.4.

a) The Context Manager has been implemented in Java using Java RMI Gateways towards context consumers and heterogeneous context sources.

b) The Context Manager offers a Web Service (SOAP) interface where context consumers can subscribe to context changes of a specified user. Results are returned over a so called Web Service call-back interface, i.e. the Context Consumer must offer a Web Service interface that Context Manager may call to return results. In phase 2, this interface should be realised with WS BaseNotification.

c) A static data model for context data should be used. The context data model can then be hard coded as e.g. SQL tables.

d) Mysql has been selected for the context database implementation.

e) Akogrimo uses RFID technology (passive) for positioning of users.

f) UAProf has been chosen for describing device capabilities.

g) Local service discovery approach is centralized, directory-based. The implementation is based on Service Locator Protocol (SLP)

h) For Akogrimo, SIP SIMPLE has been selected for supporting Presence, this due to the use of SIP as a main technology in the project.

For more comprehensive discussion of context and relevant technologies, see D2.2.4 [D2.2.4] and D4.2.1 [D4.2.1] and D4.4.2 [D4.2.2].

4.4.4. Business Process and Workflow

This section outlines some of the technological choices in Business Processing and Workflow in Akogrimo. The concepts of Business Processing, workflow and orchestration are already defined in section 3.5.

In Akogrimo we are interested in handling a Business Process based on Grid services. Mobility and Context handling (3.2 and 3.3) are other big challenges for Akogrimo and in particular for the orchestration module. This section explains how these considerations guided the overall structure of the Akogrimo orchestration module and its subdivision into four sub-components (Workflow Manager, Enactment Engine, Workflow Registry, Monitoring daemon). Each sub-component (and the interaction between them) have been designed having in mind what is already available in the market and the special needs of Akogrimo.

What is available nowadays are Business Process modelling and execution tools that target Web Services and do not foresee the possibility of dynamically changing workflow execution.

The big challenges of Akogrimo in the area of Business Process and Workflows is thus to handle Grid services and Mobility, including context changes.

4.4.4.1. Language for Business Processing - BPEL

Defining a new language and engine to handle Business Processing is out of the scope of Akogrimo. For this reason we have decided to adapt an already existing language and an already existing engine.

After having done a survey of the available languages and related engines we have decided to use BPEL to model Akogrimo Business Processes and in particular to use ActiveBPEL as the Enactment Engine. BPEL is an XML language for describing business process behaviour based on Web Services. There is currently an effort to make BPEL work also with WS-Addressing, which will be very important from the Akogrimo point of view because it would solve a problem with interfacing with Grid Services. This makes BPEL more promising then other languages. The BPEL notation includes flow control, variables, concurrent execution, input and output, transaction scoping/compensation, and error handling. All those features are necessary to design Akogrimo workflows.

Another point in favour of BPEL is its wide diffusion and knowledge of it among people who design Business Processes. It will be easier for a Business Process Designer to use an familiar

language (that also includes availability of a lot of good documentation and tutorials) rather than a language which may be more advanced, but could give implementation problems.

Future work on mobility may well require consideration of choreography, which would allow greater autonomy, but focussing on orchestration at this stage allows useful, early experience to be gained.

In choosing a suitable existing workflow engine, it is essential for Akogrimo to use open source tools. For this reason and for reasons of availability we have chosen to use the ActiveBPEL engine, although there are restrictions on the timing of instantiation which have to worked around,. This is an open source implementation of a BPEL engine, written in Java. It reads BPEL process definitions and creates representations of BPEL processes.

4.4.4.2. Component structure

As will be explained in chapter 6.4 in more detail, the Akogrimo orchestration module consists of four components.

One of the reasons why we have decided to separate the Workflow Manager from the Enactment Engine is to have the possibility, in future development, to change the Enactment Engine without affecting the Workflow Manager and the rest of the orchestration module.

At the moment we can identify the Enactment Engine with the ActiveBPEL engine. In the future we can decide to change this engine or simply to work with more then one engine. In this way our architecture is flexible enough to be not tied to a single product.

4.4.4.3. Monitoring for Context

After the choice of the language and of the engine, the major issue remaining to be solved is context handling. As mentioned above one of the components of the orchestration module is the Monitoring Daemon.

This component has been designed to take care of Context changes (subscribing to them and receiving notifications (4.4.3)) and to alert the Workflow Manager whenever one of the Context notifications could affect the Workflow execution. The Workflow Manager will then interact with the Enactment Engine communicating decisions about executing another branch of the workflow , stopping the workflow, suspending the workflow, etc (depending on the new Context).

4.4.4.4. Workflow Registry

It is also important to mention another sub-component of the orchestration module, the Workflow Registry. This has been introduced for several reasons, one of them related to the Context handling. In fact, context does not only mean context changes (that affect the workflow during its execution) but it also means information about the business process to be executed (semantic information stored in the workflow).

The Workflow Registry contains workflow templates (BPEL scripts) that could be chosen based on semantic queries to the registry. This is part of the service discovery mechanism as described. The service discovery mechanism leads us to another important issue of Akogrimo: mobility and thus dynamic change in services availability (3.6).

The Business Process enactment, that is the workflow execution, should take into account that the available services are not known a priori but that they should be discovered just before the workflow execution.

For this reason each Workflow template (in the Workflow Registry) is associated with another file (specified by BPEL). This file contains an abstract reference to the services that should be invoked during the workflow execution. Before workflow instantiation and executionn the Workflow Manager will parse this file and feeds it with the information about real services as provided by the Grid Service Discovery module.

4.4.5. Grid Infrastructure

The Grid infrastructure architecture has been based on the OGSA framework since the requirements that it imposed for Grid related functionality match the Akogrimo requirements.

The key benefits of implementing the OGSA based architecture are among others:

- It provides the Web Service framework which is a simple mechanism by which applications to become services accessible by anyone, anywhere, and from any device.
- It allows for interoperability in complex and mobile environments such as those the Akogrimo project will focus on.
- It enables dynamic location and invocation of services through service registries.
- It enables collaboration with existing applications that are modelled as services to provide aggregated Web Services.
- It handles service-based application connectivity facilitating intra-enterprise and inter-enterprise communication with respect to predetermined Service Level Agreement terms
- It allows for self-manageability, ease of use, vertical QoS and security

For this reason the 2 distinguished and widespread implementations of the OGSA framework through WSRF have been adopted in the Akogrimo project. These are:

4.4.5.1. The Globus Toolkit

The open source software Globus Toolkit, developed by *The Globus Alliance*, is a fundamental enabling technology for the Grid, leting people share computing power, databases, and other tools securely online across corporate, institutional, and geographic boundaries without sacrificing local autonomy. The toolkit includes software services and libraries for resource monitoring, discovery, and management, plus security and file management. In addition to being a central part of science and engineering projects the Globus Toolkit is a substrate on which leading IT companies are building significant commercial Grid products.

The toolkit includes software for security, information infrastructure, resource management, data management, communication, fault detection, and portability. It is packaged as a set of components that can be used either independently or together to develop applications. Every organization has unique modes of operation, and collaboration between multiple organizations is hindered by incompatibility of resources such as data archives, computers, and networks. The Globus Toolkit was conceived to remove obstacles that prevent seamless collaboration. Its core services, interfaces and protocols allow users to access remote resources as if they were located within their own machine room while simultaneously preserving local control over who can use resources and when.

This toolkit, first and foremost, includes a number of high-level services that we can use to build Grid applications. These services meet most of the abstract requirements set forth in OGSA. In other words, the Globus Toolkit includes a resource monitoring and discovery service, a job

submission infrastructure, a security infrastructure, and data management services. The Globus Toolkit is a realization of the OGSA requirements and is almost a *de facto* standard for the Grid. Indeed, the Globus Toolkit 4 includes a complete implementation of the WSRF specification.

4.4.5.2. The WSRF.NET

WSRF.NET is an implementation of the WSRF specifications on the Microsoft .NET platform. WSRF.NET consists of a set of libraries and tools that allows Web Services to be transformed into WSRF-compliant Web Services. WSRF.NET uses the IIS/ASP.NET architecture for Web Services; that is WSRF.NET services are Web Services. To use WSRF.NET, a service author first creates a Web Service using VS.NET just as they would any other Web Service. Then the service logic is annotated with attributes that the WSRF.NET tools recognize. WSRF.NET tools transform the author's compiled Web Service into a WSRF-compliant Web Service consistent with the meta–data given by the service author in the attributes.

5. Architectural Overview

5.1. Architectural Goals and Constraints

The integration of modern, mobility aware networks with Grid concepts looks at a first glance artificial. This section will provide a motivation why this combination is beneficial from a technical viewpoint. The economic dimension, which is at least equally important, is addressed in the existing and forthcoming deliverables of WP2.1, WP3.2 and WP6.3 in particular D3.2.1, the Akogrimo consolidated Value Chain and D3.2.2 The Akogrimo Business Modelling Framework. However this integration also brings additional challenges to be solved. Akogrimo, following the fundamental Grid concepts, is accordingly embracing the paradigm of resource sharing in the context of Virtual Organizations. As a major differentiator from other projects enabling the commercial exploitation of Grids, Akogrimo is considering highly dynamic Virtual Organisations (VO) that need to adapt to changing contexts, to take into account the location of a service and feature the concept of a user. This character of Akogrimo's VO concept in turn requires the vertical, i.e. cross-layer integration and interoperation of 'lower layer' functionalities such as Mobility management and user-related AAA functions with Grid middleware.

5.1.1. Opportunities and Challenges

Many Grid solutions as of today are targeting rather static Virtual Organizations where the participating organizations are infrequently changing or the resources to be shared are provided by one single company. For this kind of settings the motivation for using Grids is driven by considerations around reduction of Total Cost of Ownership (TCO) often in conjunction with reorganisation of hardware infrastructure from large mainframe systems towards cluster based solutions.

We believe that the chosen Akogrimo approach to rely for basic properties of the overall system on the developments outside the standard Grid world is providing new opportunities in particular in the following areas:

- User and identity management
- Cross organisational accounting
- Integration with multimedia collaboration tools
- Device and Context Awareness

The following table elaborates on these issues:

Area	Specific Challenges and Opportunities
User and Identity Management	The security and identity models in the Grid domain are not designed for a large number of users (for example the UNICORE security model). For typical deployments of Grids the number of users is several hundred maybe some thousand users. Furthermore as rather static Virtual Organisation has been considered a centralised user and identity model has been chosen. For Akogrimo the more advanced systems from the network domain, designed for several million users, will enable different solutions in particular federated identity models which are seen as important

Area	Specific Challenges and Opportunities
	property of a real dynamic Virtual Organisation.
Cross Organisational Accounting	The realisation of an accounting system spanning several companies is not yet sufficiently addressed in Grid solutions. Either accounting is completely handled out of band of the Grid solution or a central management is realised. For the kind of dynamic Virtual Organisations considering mobility new solutions are needed. Solutions from the network domain enabling this kind of cross organisational accounting for network services might be expanded to cover accounting for Grids in a similar way.
Device and Context Awareness	For existing Grid solution the network is seen as a simple transport layer. For supporting mobile participants applications and services must be aware of the current device capabilities and the context (e.g. bandwidth, network provider, network type, "home" or "foreign", and possibly the geographic location etc.). Without the provision of context it is hard to see how mobile participants could be supported appropriately. The way how context data is provided to the Grid middleware is again a key difference to existing solution providing portal based access to the Grid resources. If a component within a Grid suddenly disappears this is typically considered to be a failure situation and recovery mechanisms for overcoming this failure are started (e.g. replacing the service with another one). In a mobile environment this must not necessarily be a failure condition. A device (maybe even driven by a user decision) might have changed the underlying access network or entered a tunnel. An architecture for Mobile Grids must support adaptation to such kind of changing conditions
Integration with Multimedia Collaboration Tools	The integration of the user in the Grid middleware enabling workflows spanning not only compute or data services is becoming possible as Grids are using the same identity and accounting model as the multimedia communications. So in contrast to the approach taken in AccessGrid the security and identity model from Grids is not pushed down to be used for collaborative applications such as Videoconferencing but the other way around.

5.1.2. The concept of Cross-Layer Cooperation

As mentioned above, Akogrimo's VOs are context-aware. Some related context changes will be handled **intra-layer**, e.g. within the Grid Infrastructure Layer an SLA management system may replace a non-performing service with another one. In addition – and this is Akogrimo-specific – also (mobile) users and user devices as potential members of a Virtual Organisation including their changing contexts are considered. These context properties such as "Location" and "Mobility" or even SIP-based presence attributes need to be addressed by what we call **cross-layer** or **inter-layer** integration. As an example consider the move of a session from a

Workstation to a Mobile Device (Network Infrastructure Layer). The changing context (different bandwidth, screen size, …) is notified through the SIMPLE protocol to a Context Manager in the Network Middleware Layer, see section 6.2.1 which is sending a notification based on the Web Service Notification family of specifications to several subscribers. One of the subscribers is the Business Process Enactment Engine (Application Support Layer) that can put the further execution of the workflow on pause. Additionally the Operational Virtual Organisation Manager Service can start a process of finding from the workflow repository an appropriate workflow to be executed in this case (Network Middleware) and/or can add a new service provider and monitoring infrastructure (Grid Infrastructure Layer).

Another – key cross-layer issue in Akogrimo is the vertically integration of AAA providing the needed parameters in only one round-trip spannig from the related access network and core network protocols and mechanisms directly to the SOAP-based Authentication and Authorization based ones. Furthermore AAA is expected to be an instrument for realising cross organisational accounting across different layers from network through Grid middleware up to the application layer.

5.2. The Generic Akogrimo Architecture

The generic Akogrimo architecture has been initially introduced in Deliverable D311 and follows a formal layered approach where Grid Infrastructure Services are laying on top of Network middleware services which in turn are placed on top of the network services.

These overall layers are somehow controlled by a vertical application s support layer as shown in the following figure.

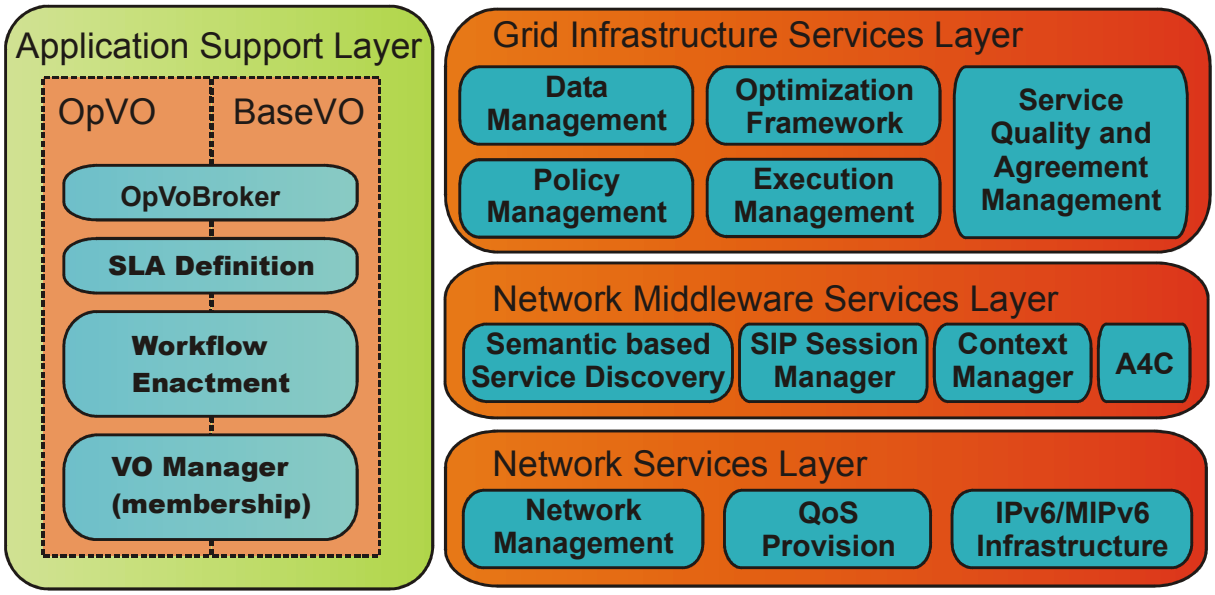

Figure 5 Akogrimo Basic Components

5.3. Service Structure

5.3.1. Steps of an Application to Become Part of the Akogrimo Platform

A prerequisite to registering a service within an Akogrimo BVO is that the service provider is a member of the BVO. Members that are registered in the role Service Provider have the right to publish services into the GrSDS of the BVO. Information about BVO members and their roles inside OpVOs is stored in the Participant Registry. The Participant Registry maintains dynamic profiles that can differ in their content depending on the specific OpVO. Basic static user profiles are stored in the A4C server. In order to integrate a service into the Akogrimo platform a service provider has to take care of two parts:

- Service Description
- Service Interactions

The description is used in the finding phase. Service interactions occur in the binding and execution phase.

5.3.1.1. Service Description

The service description is used by the GrSDS to find suitable services for a customer. With the term service, in Akogrimo we mean a Web Service or WS-Resource. Their functional capabilities (services interfaces) are going to be described using standard technologies. Furthermore, in Akogrimo we need to describe non-functional capabilities, e.g. semantic information, QoS, access rights, etc.. The following description standards are going to be used:

- OWL-S mark-up of Web Services will facilitate the automation of Web Service tasks, including automated Web Service discovery, execution, composition and interoperation
- WSDL
- WS-Agreement for defining the SLA template and contract
- WS-Policy

If the service is represented by a workflow, a workflow template has to be published into the workflow registry. The workflows will be described using:

- WS-BPEL

5.3.1.2. Service Interactions

Service related interactions occur between the OpVO and the service that belong to the hosting environment of the service provider. In order to interact with the OpVO, the hosting environment and the service have to support the following:

Interaction	Technology / Protocol	Description	Available Component (for use in the Hosting Environment)
SLA Negotiation	WSLA, WS-Agreement	The SLA Negotiator of the hosting environment talks to the SLA Negotiator of the OpVO. Once a agreement has been reached, the contract is stored in the SLA contract repository via the SLA-Access service.	SLA Negotiator
Policy Management	WS-Policy	WS-Policy is used to make statements about the policies that apply to the service or hosting environment.	Policy Manager
Registering for context information	WS-Notification for subscribing	The service can contact the Context Manager to subscribe to context information of a specific user.	
Receiving context information	WS-Notification	The service has to implement context sensitivity if desired.	
Authorisation	DIAMETER, SAML Assertions	The service has to find out if a service request should be authorized.	A4C Client
Configuring Network QoS	QoS Broker Web-service interface	The service provider has to understand the service interface of the QoS Broker.	
Receiving network metering information	WS-Notification	The service provider may use information about the network utilisation.	
Reporting SLA-Violations to the OpVO	WS-Notification	The service provider has to report SLA violations to the OpVO Manager. The service provider may use Akogrimo components for local monitoring and management.	Monitoring, SLA-Controller, SLA-Decisor, EMS
Sending Accounting	DIAMETER	The service provider sends information about service	A4C-Client

information		usage to the A4C server. This includes information about SLA violations.	
Data Transfer	GridFTP, WS-Attachment	The service provider may use the Data Manager component if needed.	Data Manager, GridFTP Server component of GT4

Table 2: Service Interactions

6. Akogrimo Sub-Systems and their relations and interfaces

6.1. Network Services Layer Architecture

The next figure shows a diagram depicting a network-oriented view of the Akogrimo network. Components which do not belong to the Network Services Layer are shown in dimmed colours.

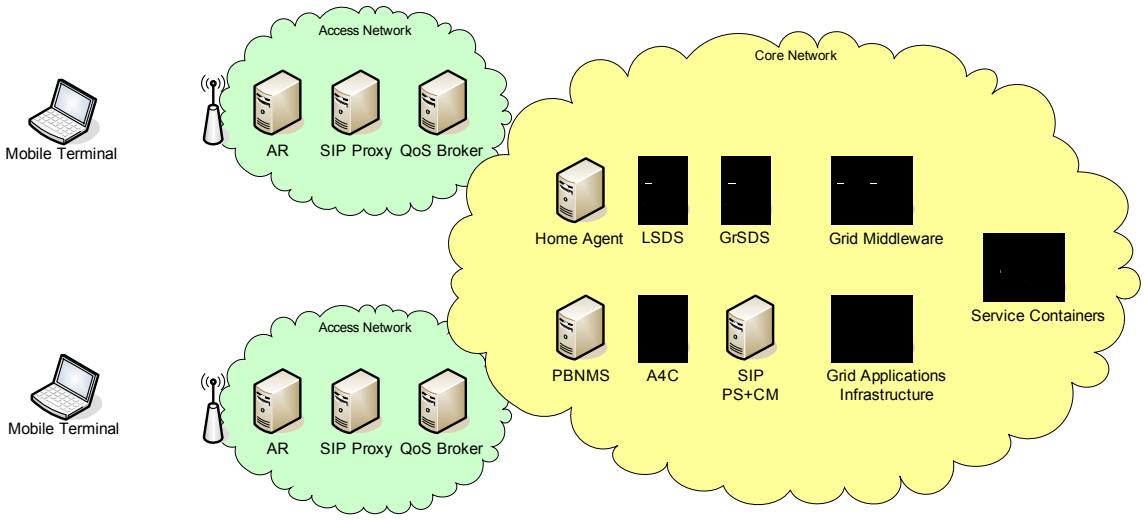

Figure 6: Network Layer Diagram

The core network is the central part in the network. It connects all access networks and controls the user's identity, security, authentication, authorization, accounting, auditing, charging, multimedia services and mobility. It is also responsible for providing users with access to Grid services.

The access network allows the connection of terminals using different access technologies and communication with the core network. The core network has several access networks. Several user terminals may connect to a given access network. Akogrimo will focus on Ethernet and Wireless LAN access technologies, since the infrastructure required for other technologies (e.g. UMTS) is very expensive. However the use of IPv6 as a convergence layer guarantees that, should the need arise, those other technologies can in the future be integrated without any problems in the Akogrimo network.

The access network is comprised of at least one Access Router and one QoS Broker. The SIP Proxy is part of the SIP Server which is located in the core network; although for scalability reasons the proxy module may be separated and distributed among the various access networks (which is how it is depicted in figure 6).

6.1.1. IPv6/MIPv6 infrastructure

Due to the rapid evolution and constant growth of the Internet, there has been a need to redesign IPv4 in order to accommodate present and future demands. To this end, IPv6 was developed as a replacement of IPv4. It brings along a lot of improvements to IPv4; among others, extended addressing capabilities, header simplification, extensibility, etc… However, for both IPv4 and IPv6, the IP address topology is designed in a way that determined addresses belong to determined networks or sub-networks, and thus, the routing of packets is performed

according to this structure. If a node disconnects from its point of attachment to the network and connects to the network at any other place, a reconfiguration of a new IP address, netmask and routers is necessary. Mobile IPv6 extends IPv6 so that mobile terminals are able to change their point of attachment to the network with minimal disruption. When a mobile terminal roams across different networks, Mobile IPv6 deploys a mechanism, restricted to the network layer that makes this change of position in the network transparent to higher layers. Therefore, already established connections (for example a TCP connections) are not dropped when a node changes its position from one network to another and needs to reconfigure its network characteristics. In fact, these connections are not even aware of this change. Since Mobile IPv6's scope is restricted to the network layer, it does not impose any requirements on the lower layers. This means that a mobile terminal is able to roam across networks independently of the access technology. For example, a hand-over may take place between two Ethernet accesses or between an Ethernet and a WLAN access without any concern.

The main idea of how mobility has been implemented in MIPv6 revolves around the use of not only one but two addresses for a node to be addressable at any point. Theses addresses are:

- Home Address (HoA): As any other regular node, a mobile terminal has a permanent address, which is valid inside his home network. This address always remains the same, independently of in which network the mobile terminal is located, i.e. whether at home or roaming. This is the only address that the transport or higher layers are aware of.
- Care-of Address (CoA): For as long as the mobile terminal is located in a network other than its home network (i.e. a foreign network) it makes use of the CoA, which is a suitable address for the foreign network in which the mobile terminal is located at that moment. The relation between the HoA and the CoA of a mobile terminal is called binding.

The new entities that the Mobile IPv6 protocol adds to IPv6 are described as follows:

- Mobile terminal (MT): A node with the ability to change its point of attachment to the network keeping connections alive.
- Home agent (HA): A node that resides at the MT's home network, which is in charge of forwarding traffic directed to the MT while it is away from home so that it looks as though the MT is virtually at his home network. To this end, the HA must be aware of the MT's current binding.
- Correspondent node (CN): Any peer node which a MT is communicating with. A CN does not necessarily have to implement mobility i.e., a CN may be either mobile or stationary.

When the MT is away from his home network the HA will act on behalf of it forwarding all the packets addressed to the MT at its HoA to its current CoA by means of a tunnel. The CN is not aware that the MT is roaming unless it supports Mobile IPv6. If this is the case, the CN will realize that its traffic is being tunnelled by the HA and may perform a route optimization by directly communicating with the MT by means of the MT's CoA instead of using the tunnel provided by the HA. This last should be the standard mechanism because it notably increases the performance. In order for the MT to communicate its position to the HA and all the CNs it is communicating with, a procedure is launched in which a packet called Binding Update (BU) is sent with the address currently used and a packet called Binding Acknowledgement (BA) is received by the MT as a confirmation. Regarding security, these packets should be protected, since several attacks can be applied if this is not done (man-in-the-middle, impersonation, DoS and others). The standard way to protect these packets is IPSec although there are other alternatives, such as using cryptographically protected CoAs.

6.1.2. Service Description and Provisioning

Network Service Provisioning in Akogrimo is intended to provide network functionality to Akogrimo higher layers. This means that all relevant user-network interactions (that comprises QoS, mobility, security and session management topics) must be available to the Grid layers in order to enable Grid-based services.

The goal of Akogrimo is to achieve integration of network and Grids; however for the first phase of the project this integration will be limited to a selected number of important services, like QoS and SIP. Since Grid technologies are based on open standards such as SOAP, WSDL, XML, etc., it was decided that the network layer should make the effort of developing those selected services as Web Services, in order to allow a more complete and homogeneous integration with the Grid infrastructure or applications.

An end-user application that uses network intensively, like for example a video-conference application, needs to have network quality of service in order not to be disrupted by changingl network conditions. To that end, an application needs to be able to make QoS requests on behalf of the user so as to setup the video-conference session. Likewise, in a scenario where a workflow suddenly requires the establishment of an audio link between a patient and his doctor, it is useful to have quality of service for that audio conference. The QoS Broker Web Service will allow any Grid application to use Web Service standards to make the QoS reservation.

The SIP protocol will play an important part in Akogrimo. The Session Initiation Protocol (SIP) is an application-layer protocol that provides mechanisms to establish, modify and terminate sessions. In conjunction with other IETF protocols it can provide advanced support for multimedia sessions, but it is not restricted to multimedia. These protocols are SDP, Session Description Protocol for describing multimedia sessions; RTP, Real-time Transport Protocol for transporting real time traffic and RTSP, Real Time Streaming Protocol for controlling delivery of streaming data. It can also provide session mobility. Also, if a workflow needs to establish a call between two other distinct parties, it will use SIP's 3PCC (Third Party Call Control) capabilities and play the role of session controller.

For the second phase of the project, it is intended to further advance on the network-Grid integration, providing not only cross-layer interfaces, but more importantly, to transparently expose network services and resources as Grid services, making the network an integral part of the Grid.

6.1.3. Network Management

Network management is done primarily with the A4C, QoS Broker and the Access Routers. The A4C Server acts as a Policy Information Point, which provides necessary information about users to the QoS Broker. The QoS Broker, in its turn, acts as a Policy Decision Point. The information it gathers from the A4C Server is used to make informed decisions. Those decisions are then communicated to the access routers, which are in charge of enforcing those decisions, thus acting as Policy Enforcement Points. The access routers also allow homogeneous communications with mobile terminals, independently of the access technology they use.

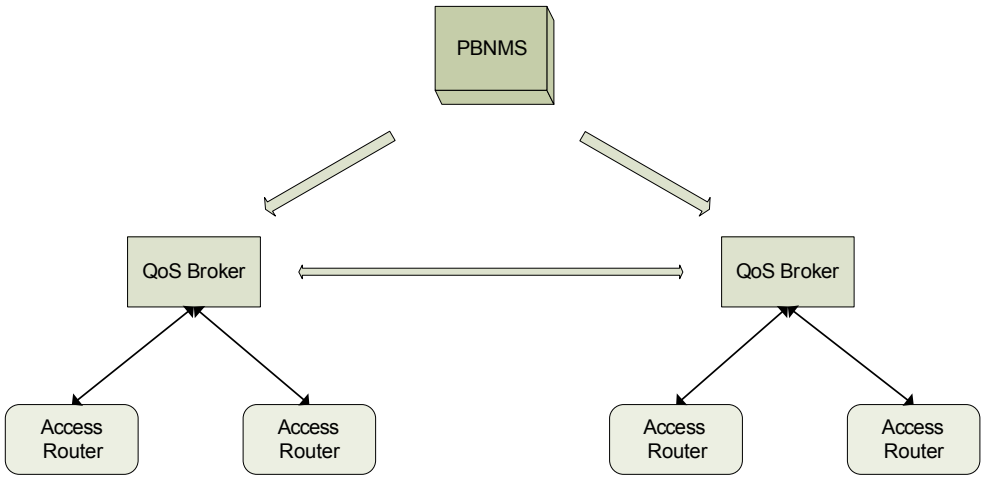

Figure 7: Management components hierarchy (A4C Server not depicted)

Akogrimo will also have Policy Based Network Management (PBNM) as the basis of network management (although it was not considered in the first phase of the project). PBNM is an alternative for the management of telecommunications networks that offers a way of overcoming many of the limitations of existing human resource-intensive network management techniques. In heterogeneous and distributed environments like Akogrimo, network configuration to guarantee required end-to-end QoS, or to assure certain rules for admission or congestion control could be a very complex problem to solve without automatic mechanisms for configuration and control.

The PBNM allows easy creation of policies through an intuitive interface and then applies those policies to all the machines affected. It will also respond to policies solicitations from entities which require them.

6.1.4. Network Resource Management

In Akogrimo we consider that the core network is sufficient for all traffic from all users. This is done because ISP's typically have low usage rates of their core networks; the real issues are on the access networks which have more limited bandwidth available, and more so with wireless access networks.

The main entity in charge of managing network resources is the QoS Broker. The QoS Broker knows at all times who is using the network and what bandwidth is used. It receives policies from the PBNMS and user profiles from the A4C Server. QoS requests are approved or not depending on the network status, user profile and global network policies. Multiple QoS Brokers exchange information about the respective access networks they are administering.

Akogrimo will have a hybrid IntServ/DiffServ QoS model. IntServ is used in the access networks since it supports fine-grained QoS, but is not scalable, which poses a problem for networks of significant dimension. The access routers support different reservation methods and translate them to COPS messages that are sent to the QoS Broker, which then decides whether or not to allow the reservation requests according to user profile and network conditions.

The core network uses DiffServ for aggregating network flows with the same QoS requirements. While DiffServ does not allow fine-grained QoS parameters setup, it is the optimal solution for big networks which require great scalability. The access network's IntServ signalling is translated by the AR to DiffServ signalling, which is appropriate for the core network. Any network flow in the core network will be aggregated with other similar flows, following the DiffServ model.

This setup allows fine-grained QoS control at the access networks, along with scalability due to the use of DiffServ at the core network, effectively combining the strengths of both IntServ and DiffServ.

6.2. Network Middleware Services Layer Architecture

Figure 2 gives an overview of the Network Middleware Services Layer Architecture including its main components, interfaces, interface protocols and labels. Details on the network middleware layer architecture can be found in deliverable D4.2.2 Final Integrated Services Design and Implementation Report.

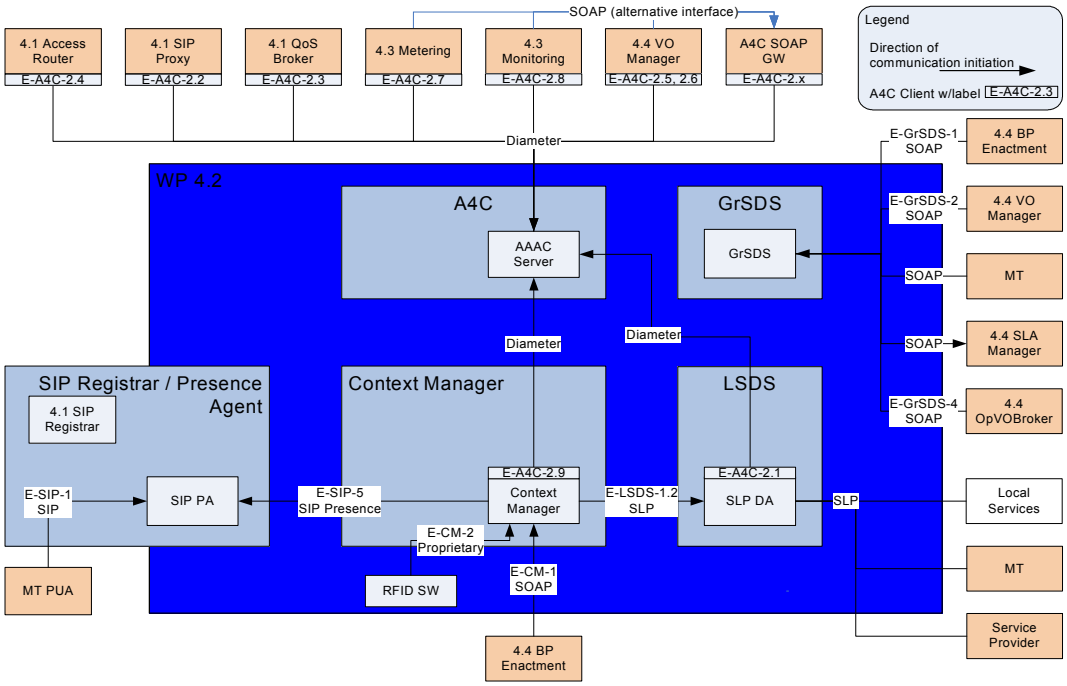

Figure 8: Network middleware components and interfaces

6.2.1. Context Management

It is not desirable that every application implements a context engine to gather and process specific context information; rather a generic context platform/infrastructure that supports applications by gathering, processing and managing context information is needed. It is not realistic that such a context infrastructure can provide every representation of context information needed for any domain specific purpose. The approach is therefore that the context system gathers basic context data, processes and refines it according to common Akogrimo context requirements. In the Akogrimo architecture, the component that handles context is denoted Context manger. The Context Manager gathers raw context data from multiple heterogeneous context data sources. The Context Manager refines raw context data to provide standard formats (e.g. geographical coordinates, temperatures, etc.), performs processing according to common Akogrimo requirements and stores the context data in a context database.

Context consumers subscribe for notifications about changes in certain context data. The Context Manager is responsible for distributing context data to context consumers accordingly.

Interfaces

In the following a brief description of external interfaces is given.

The Context Manager gathers context data using the following interfaces:

- To collect user presence data (including terminal capabilities) the Context Manager interacts with a SIP Presence Agent using SIP SIMPLE (E-SIP 5). - Protocol: SIP (SIMPLE)
- The Context Manager collects information about services available in the nearby area of a user (but not web/Grid services) from a SLP Directory Agent (DA) (E-LSDS.x). - Protocol: Service Locator Protocol (SLP)
- The Context Manager gets the location/position of users from RFID readers placed at specific locations (E-CM.2). - Protocol: Java RMI (proprietary).

Additionally, the Context Manager has an interface to the A4C system:

- The Context Manger interacts with the A4C system. (E-A4C-2.9) to obtain static user data (SIP address, RFID, …) and to authorise context consumers (not implemented in Phase 1). Protocol: Java API

Finally, the Context Manager provides a Web Service interface (E-CM.1) to interact with context consumers:

- Context consumers use this interface to subscribe/unsubscribe to context information. Notifications about context changes are returned as over a so called Web Service call-back interface, i.e. the Context Consumer must offer a Web Service interface that Context Manager may call to return results. - Protocol: SOAP

6.2.2. Service Discovery

The Service Discovery System (SDS) has functionality that allows a service requestor to find an appropriate service provided by a service provider. A service provider must describe the service capabilities (Service Description) and publish the service through the Service Discovery System.

Two systems for service discovery are used; discovery of "local services" (SLP based) and discovery of Web Services/Grid services (UDDI based). The system in charge of discovering local resources (services) is called the Local Service Discovery System (LSDS), whereas Grid Service Discovery System (GrSDS) is the name for the module dealing with proper Grid Services. Implementation of GrSDS will be mainly concentrated on next development phase.

In the following, a brief description of Akogrimo service discovery systems is presented. Details are found in D4.2.1 [D4.2.1] and D4.2.2 [D4.2.2].

6.2.2.1. Grid Service Discovery System (GrSDS)

The GrSDS represents a "static" discovery registry with semantics capabilities where the static description of services (i.e. WSDL description) is stored. The two basic processes involving GrSDS are the publication and the search of Grid Services (WS-Resources). The publication of Grid services consists mainly on storing descriptions of services in the same way as WSDL document are stored in a standard UDDI registry. Furthermore, the publication of Grid Services implies also the process of storing information to the corresponding SLA Template stored physically in the SLA Repository. Regarding the process of Grid Service search, two different components can look up for services within the GrSDS in the process of acquiring the service. Those components are the WorkFlowManager (WFManager) and the Operational Virtual Organisation Broker (OpVOBroker).

Some important features like semantics have not been addressed yet at this stage, since the design and implementation phase will be mainly concentrated on next development phase.

The implementation of this component will be based on the well-known and established UDDI specifications. UDDI can be considered as the start point especially regarding the registry part of the GrSDS component. In addition a more fitted publication component has to be developed over UDDI in order to be able to store Grid Services (not just stateless Web Services). Moreover, semantic functionality should also be added, especially in the WS-Resource search process.

Interfaces

As illustrated in Figure 2, the GrSDS has the following interfaces towards other components:

- Interface to search and fetch a Grid Service (E-GrSDS-1)
  Protocol: SOAP
- Interface to Publish/Deregister Service (E-GrSDS-2)
  Protocol: WSDL/SOAP
- Interface to interact (E-GrSDS-3) with the SLA-Access to retrieve the SLA from published SLA-Template and associate them to the interface of service published in GrSDS
  Protocol: SOAP
- Interface to look for services to carry out a certain WF (E-GrSDS-4)
  Protocol: SOAP

6.2.2.2. Local Service Discovery System (LSDS)

LSDS is based on the **Service Location Protocol (SLP)** set forth by Internet Engineering Task Force (IETF). SLP allows network clients to discover available services on a network. Services are located by means of URL and some additional name-value pairs, called attributes. The definition of services are specified by templates defined in [RFC2609].

In SLP three different entities can be observed:

- User Agents (UA): Are the entities who perform the query.
- Service Agents (SA): Are the entities that represent services
- Directory Agents (DA): Are entities that keep information about the available services in the network.

SLP may work in two different modes depending on the size of the network being managed. In Akogrimo, a centralized modality is used in which a DA is introduced which a UA queries in order to look for a service. SLP would need to be extended in the case of the centralized mode in order to fully support mobility of services.

SLP is able to:

- Look for all services of a determined type
- Query by attributes by making use of LDAP's query language.
- Request the attributes of a services

Interfaces

- Interface that allows the service requestor (CM) to search for Local Services for user Context using LDAP query language (E-LSDS-1.2)
  Protocol: SLP

- Interface to A4C used to authorize user or SP (E-A4C-2.1)
  Protocol: Diameter

6.2.3. A4C

The A4C infrastructure provides the necessary functionality for authenticating users, authorizing access to services, accounting and charging for service usage, as well as auditing within Akogrimo. It provides an interface to other components which allows them to access and store key information about users and their usage of services. This data is stored within a database that is maintained by the A4C server. Important components that access A4C include Base VO manager, SIP server, Context Manager, access router, and QoS broker.

The A4C infrastructure consists of the following modules:

- A4C server: it performs and manages the A4C tasks and communicates with other servers within and across domains.
- A4C client: it supports the communication and message exchange with the A4C Server and is needed within each network component requiring A4C support.
- SAML authority: It is responsible for the generation and verification of SAML IDTokens and SAML Assertions.

Details on A4C modules and interfaces are found in D4.2.2.

Interfaces

Akogrimo network components requiring A4C functions use the A4C client library to communicate with the A4C server via the Diameter protocol. Grid components can either include the A4C client library or the library can be used to create an A4C SOAP gateway, which provides a SOAP interface for Grid components. The SAML authority assists the A4C server in the authentication process and provides the IDToken management and verification.

- The A4C server interface (E-A4C-1) comprises a single management interface (E-A4C-1.1) used for A4C server configuration and maintenance.
- The A4C client interface (E-A4C-2) comprises several sub-interfaces (E-A4C-2.x):
    a. Authentication and Authorization API
    b. Accounting API
    c. Auditing API

6.3. Grid Infrastructure Services Layer Architecture

6.3.1. Execution Management Services

6.3.1.1. Objectives

The Execution Management Services (EMS) comprise the central controller of the business service execution. The aim of EMS is to:

- Decide where a task can execute. The locations at which it can execute must satisfy the QoS parameters retrieved from the SLA-Access Service. These parameters include CPU and binary

type, and available Disk space. Furthermore, the EMS must check the existing policy restrictions, which may further limit the candidate set of execution locations.

- Prepare the task to execute. Setup includes reservation of CPU, memory and bundles resources, deployment and configuration of binaries, staging data.
- Manage (monitor, restart, move, etc.) the task. The EMS is also responsible for notifying the service consumer when the task is complete.

6.3.1.2. Functionality

The Execution Management service is developed on the basis of the WSRF specification. It is a persistent service that directs the accomplishment of the requirements specified in the contracts of the Akogrimo clients. In more details, it coordinates the operation of the various modules responsible for the execution, monitoring and SLA enforcement of the business services. The basic operation of the EMS is as follows:

The Workflow Engine uses this operation to invoke the Execution Management Service. The EMS contacts the SLA-access service in order to retrieve the quality of service parameters that are defined in the contract with the specified id (slaID) by invoking the getQosParameters operation of the SLA-access service. After retrieving the quality of service parameters, the EMS uses this info in order to find the most appropriate host for the execution of the E-Health service that the client wants and performs a reservation by creating a resource for this service in the selected host and saving the endpoint reference that corresponds to this resource in a database. The EMS sets the time period for which the E-Health service resource will be available and also performs a subscription to this E-Health service resource in order to receive notification messages from it. Afterwards, it contacts the SLA-Controller service and creates a corresponding resource. The EMS activates this SLA-Controller resource and also invokes the Monitoring service by calling the startMonitoring operation at the time specified in the contact.

To achieve this, the EMS is responsible for a set of actions concerning the resource management through a set of sub-components like:

Job Manager

The Job Manager (JM) encapsulates all aspects of executing a job, or a set of jobs, from start to finish. A set of job instances (a complex job) can be structured (e.g., a workflow or dependence graph) or unstructured (e.g., an array of non-interacting jobs). It can schedule for them and it can collect agreements and reservations.

Candidate Set Generator

The Candidate Set Generator (CSG) is in charge of finding the computational resources where the job can be executed. The match-making supports finding of the resources that correspond to the requirements expressed by the service consumer. Examples of static resources are: operating system, number of processors, available software available (libraries, binaries, etc), memory, disk space, bandwidth, etc… Dynamic resources are resources such as: free disk space, available CPU and file transfer rate.

Execution Planning Service

The Execution Planning Service (EPS) is a high level scheduler that creates mappings called "schedules" between jobs and resources – the resources that have been selected by the CSG. An EPS attempts to optimize some objective function such as execution time, cost, reliability, etc (i.e. answering "where should the job execute?"), in order to improve system performance,

provide Quality of Service and meet the Service Level Agreements. For this reason the EPS implements a job priority system.

Advanced Reservation Service

The Advanced Reservation Service **(**ARS) is in charge of reserving resources for a specific period of time or permanently, depending on the type of reservation. Different types of reservation are possible:

- Computational resource reservation
- Storage resource reservation
- Network resource reservation

The computational reservation guarantees that the specified resource is available at the time that a job is executed. The service consumer (SC) can reserve a certain amount of disk space, a specific percentage of CPU, etc. While the network resource reservation involves the network bundle services provided through the QoS Broker of the networking layer.

6.3.1.3. Interfaces and interactions with other components

WP4.3 Subcomponent	Interfaces with Subcomponent, WPs	Interfaces / protocols
EMS	*Data Management* To handle data related jobs (consume / produce data).	SOAP
EMS	*Monitoring* To update the status of the execution and for the manageability of services.	SOAP
EMS	*Metering* To receive updated metering information of the execution.	SOAP
EMS	*SLA Enforcement* To receive the SLA parameters that have to be met.	SOAP
EMS	*SLA High Level Services (WP4.4)* To check service availability, To confirm reservations with SLA-Negotiator.	SOAP
EMS	*Policy Manager* To retrieve the policies that will be applied in the execution procedure.	SOAP
EMS	*BP Enactment (WP4.4)* To initiate an execution and retrieve the results of it (if any).	SOAP

EMS	*QoS Broker (WP4.1)* To use the network services of WP4.1. To request network resources availability and reservation.	SOAP
EMS	*Service Discovery (WP4.2)* To discover the services that need to be used in the various execution phases.	UDDI

Table 3: Interfaces for the Execution Management component

6.3.1.4. Involved technologies

The technologies involved for the set of the EMS services are mainly satisfied by the Globus Toolkit. This includes a set of service components collectively referred to as the Grid Resource Allocation and Management (GRAM). GRAM simplifies the use of remote systems by providing a single standard interface for requesting and using remote system resources for the execution of "jobs". In the context of GT4 the involved technology is the Web Service GRAM (WS-GRAM). Concerning the Candidate Set Generator functionality, various technologies such as Condor-G can be potentially used. The technologies needed for the implementation of the Execution Management Service are: WSRF and WS-Notification specifications, Java WS-Core 4.0.1 included in the GT4 toolkit and Java JDK 1.5.0.

6.3.2. Data Management Services

6.3.2.1. Objectives

The Data Management component is in charge of storing data, transferring data from one location to another and in general of keeping tracks about data stored.

The Data Management is able to handle large amount of data. It is able to handle different kind of data formats like text, binary, etc. The Data Management module has been developed has a Web Service and it is not directly linked to other components of the layer. This means that it can be used as a stand alone service.

6.3.2.2. Functionality

The Data Management component is implemented as a Web Service and it is based on the RFT (Reliable File Transfer) component of the Globus Toolkit 4.

It exposes the following methods:

- *upload*
- *retrieve*
- *transfer*

The methods listed above reflect the Data Management behaviour. The first two methods, *upload* and *retrieve*, identify the data management module as a data manager itself. The third method, *transfer*, let the data manager acts as a third party instrument for data transfer.

The idea behind the upload and retrieve methods is that data should be first uploaded to the Data Manager. Once uploaded data is identified by a logical name that correspond to a physical name. Please note that for the first implementation data replication is not foreseen; the

implementation will be enhanced when data replication is inserted into the module features. Once the data are uploaded to the Data manager, data can be retrieved in any time.

The data transfer permits moving data from one location to another. It is implicit that, in order to do this, some software has to be installed on the machines that want to use this third party feature. Each one of the methods above needs some parameters in order to work. **¡Error! No se encuentra el origen de la referencia.**The following table provides a complete synopsis for each method.

6.3.2.3. Interfaces and interactions with other components

WP4.3 Subcomponent	Interfaces with Subcomponent, WPs	Interfaces / protocols
EMS	*Reliable File Transfer* To transfer a file, as requested, from one location to another.	EMS request for files transfer (SOAP)
EMS	*Replica Location Service* To discover the physical location of the replicated data file. To register replicated files in the Replica Location Service.	EMS discover the best replica based on the information provided by the RLS (SOAP)
EMS	*Data Replication Service* To add, register and replicate data files, or delete replicas.	Ask to replicate a file (SOAP)
EMS	*Metadata Element Service* To add and retrieve metadata related to a replicated file, or query a metadata catalogue to find out information related to replicated files.	Reserve disk space
Policy Manager	*Storage Element Service* To upload a file to, or retrieve it from a Storage Element. To move a file within a specific Storage Element, or delete it from it.	Negotiate local data access (SOAP)

Table 4: Interfaces for the Data Management component

6.3.2.4. Involved technologies

The Data Management module is based on the following technologies: GT4 , in particular Reliable File Transfer, Java 1.4.2 and Tomcat 5.0.28

6.3.3. Monitoring Services

6.3.3.1. Objectives

The Monitoring module of the Akogrimo architecture has the function of collecting data regarding to a service execution from different sources and sending this data to the interested modules.

In a first stage, the modules sending data to the Monitoring one are:

- QoS Broker. This module sends to Monitoring a change in the Network Bundle.
- Metering. This module sends to Monitoring the CPU Load and the Number of Instantiations associated to a certain service.

The module receiving this information from the Monitoring is the SLA Controller, which needs it to fix a service violation.

6.3.3.2. Functionality

The Monitoring component is modelled like a WS-Resource, implemented following the WSRF and WS-Notification specifications and using the GT4 toolkit.

For the Monitoring module begin to work it exposes the method startMonitoring, that will be called by the EMS module. The next step is the Monitoring receiving the measured data from the QoS Broker (Network Bundle parameter) and Metering (CPU Load and Number of Service Invocations parameters). For this purpose, the Monitoring module exposes two methods: setMeteringData and setQoSBrokerData, invoked, respectively, by the Metering and QoS Broker modules to sending this information related to the service performance. With the received information, the Monitoring component makes some calculations and sends to the SLA Controller the values of the SLA parameters.

6.3.3.3. Interfaces and interactions with other components

WP4.3 Subcomponent	Interfaces with Subcomponent, WPs	Interfaces / protocols
Monitoring	EMS (Job Manager), WP4.3	SOAP
Monitoring	EMS (Candidate Set Generator), WP4.3	SOAP
Monitoring	SLA (SLA-Controller), WP4.3	SOAP
Monitoring	Metering, WP4.3	SOAP
Monitoring	Metering, WP4.1	SOAP
Monitoring	Accounting, WP4.2	SOAP

Table 5: Interfaces for the Monitoring Component

6.3.3.4. Involved technologies

The following technologies are needed for the implementation of the Monitoring module: WSRF and WS-Notification specifications, Java WS-Core 4.0.1 included in the GT4 toolkit and Java JDK 1.5.0.

6.3.4. Service Level Agreement Enforcement

6.3.4.1. Objectives

The SLA Enforcement is split in two main services; SLA Controller and the SLA Decisor. The SLA Enforcement services are in charge of

- Receiving the QoS measurements sent by monitoring
- Checking throughout the execution phase the fulfilment of the SLA contract agreed between a service provider and a service client
- Notifying of new or existing violations

The SLA Enforcement system has a supervisor role and it is responsible for the verification of the contract conditions (QoS thresholds) of all running services, alerting about any abnormal situation and taking quick and effective decisions.

6.3.4.2. Functionality

This service is in charge of controlling the status of a service while it's running. It is created by the EMS component when a new instance of a service is created. Afterwards, it receives from the Monitoring component continuous measurements of the QoS parameters according to these ones defined in the contract related with the service execution. Two modules have been identified in the design of SLA Management architecture:

1) SLA-Controller

This module is in charge of ensuring that all the agreements reached in a contract are respected. It receives and processes all measurements performed by Monitoring subsystem and therefore it must be deployed alongside the Monitoring subsystem in order to avoid network communication overheads. It checks whether the measurements are within the thresholds established in SLA contract for QoS metrics. This module also applies the domain filter extracted from Policy Manager before sending eventual notifications to SLA-Decisor (in cases when the QoS is not fulfilled).

2) SLA-Decisor

This receives and manages notifications from the SLA-Controller. In turn, it requests the Policy Manager to retrieve the suitable policy associated according to the execution context. This policy contains the actions that must be undertaken (to show informational messages, to increase processes priorities, to apply discounts, to destroy the service, to re-instantiate the service, etc). Depending on the seriousness of the violation, context of service and the overall status of the system, this event can be solved at domain level or at VO level. In the first case, it may be only necessary to notify the EMS which will apply corrective actions, whereas in the second case, it may be necessary to update higher layers (Workflow manager or other subsystem) to recover from the situation by applying global corrective actions. It is the responsibility of the designer to decide which performed actions will be managed directly by the Policy Manager or the SLA-Decisor. After selecting the best policy, the Policy Manager can ask directly for a concrete action

on another component, or on the other hand it can communicate the corrective action to the SLA-Decisor.

6.3.4.3. Interfaces and interactions with other components

WP4.3 Subcomponent	Interfaces with Subcomponent, WPs	Interfaces / (protocols)
EMS	SLA-Controller	To create an "agent" to control the status of the new service created / SOAP
SLA-Controller	Policy Manager	To obtain the mapping defined between the High and Low level parameters from a service / SOAP
SLA-Controller	SLA-Decisor	To create a subscription of SLA-Controller into the SLA-Decisor for posterior communications between these two modules/ SOAP
Monitoring	SLA-Controller	To notify SLA-Controller about the value of the QoS dimensions related to the service / SOAP
SLA-Controller	Monitoring	To ask directly Monitoring about the value of the QoS dimensions related to the service / SOAP
SLA-Controller	SLA-Decisor	To inform to SLA-Decisor about a violation produced on the QoS fulfilment of the service/ SOAP
SLA-Decisor	Policy Manager	To ask for the best policy to apply in case of violation / SOAP
SLA-Decisor	EMS / Workflow Manager (according to associated policy)	To perform action when something has gone wrong during the service execution / SOAP

Table 6: Interfaces for the SLA enforcement component

6.3.4.4. Involved technologies

SLA Controller and SLA Decisor are developed in Microsoft .NET platform and therefore they use the .NET framework 1.1 with WS Enhancements (WSE 2.0 SP3)

SLA Controller is a transient Grid service that makes use of WSRF.NET implementation of the University of Virginia. The communication between both modules (SLA Controller and SLA Decisor) is based on WS-Notification specification.

In terms of SLA Contract definition, the WS-Agreement specification is considered. It defines the structure of agreements and their templates and it can be extended and complemented by other terms. However, the analysis of approaches gathered in WSLA (IBM specification) has been taken into account to define the contract template.

6.3.5. Metering Services

6.3.5.1. Objectives

The main objective of the metering component is to measure the usage of resources that belong in the Akogrimo Grid infrastructure and to communicate them to the AAA sub-system, so that an aggregated accounting record for the usage of resources can be made for the customers of the Akogrimo platform.

User applications have different resource requirements depending on computations performed and algorithms used in the specific application services that are used in Akogrimo. Some applications can be CPU intensive while others can be I/O intensive, or a combination of both. For example, in CPU intensive applications it may be sufficient to charge only for CPU time whilst offering free I/O operations. This scheme is not sufficient for I/O intensive applications. Finally, wall clock metering of resource and services consumption which is applicable as a general measurement of utilization of resources which is applicable in cases of having idle resources (case of advance reservation which might have been unused).

The Metering service provides the functionality required for the measurement and logging of the resources that the system dedicates to the execution of the business services (E-Health) services that are requested by the customers. This information is necessary for the accounting system's operations.

6.3.5.2. Functionality

The Metering service shall initially record for each host the following performance parameters:

- CPU utilization
- Disk utilization

The Metering service is invoked by the Execution Management service when the execution of a requested by a client E-health service starts. The Metering service periodically records the parameters listed above and invokes the *setMeteringData* provided by the Monitoring service in order to notify it about the values of these parameters. The notification message passed to the Monitoring service through the *setMeteringData* operation is an XML-like string that includes the IP address of the host and all the necessary information about the parameters that are being monitored.

6.3.5.3. Interfaces and interactions with other components

WP4.3 Subcomponent	Interfaces with Subcomponent, WPs	Interfaces / protocols
Metering	EMS/WP4.3	SOAP
Metering	Monitoring/WP4.3	SOAP
Metering	AAA/WP4.2	SOAP

Table 7: Interfaces for the Metering component

6.3.5.4. Involved technologies

The technologies needed for the implementation of the Metering service are WSRF and WS-Notification specifications, Java WS-Core 4.0.1 included in the GT4 toolkit and Java JDK 1.5.0

6.3.6. Policy Management Services

6.3.6.1. Objectives

Within the Akogrimo system different kinds of communications with different kinds of terminals take place. Many participants are involved in these communications and there are rules necessary in order to manage them. These rules represent policies, fixed on the base of participant roles and participant services offered. In this context, the policy describes how to use the available resources and has nothing to do with access permissions to the available resources. Agents can change their own behaviour on the basis of the policies received.

Policies are a means of specifying and influencing management behaviour within a distributed system, without coding the behaviour into the manager agents. Furthermore, in a large distributed system, such as Akogrimo, there are multiple human administrators specifying policies that are stored on distributed policy servers. Policy conflicts can arise due to omissions, errors or conflicting requirements of the administrators specifying the policies, but the existence of Policy Management Services within the system's architecture helps alleviate this problem.

6.3.6.2. Functionality

The Policy Manager architecture supports the following functionality:

- The DB management functionality allows addition/removal DB policies information. This functionality provides an interface used to manage policies stored in DB.
- The Policy Selection functionality searches in DB for the policies that are involved in the context described by the request being performed by PR.
- The Uplink Request functionality retrieves the Global policies from GPM. The request is also based on the request being performed by PR.
- The Policy Intersector collects all the policies returned from the policy selection process (local and global) and merges them in a single policy response.
- The Query Service exposes an interface used by PR to submit query to PM System.

6.3.6.3. Interfaces and interactions with other components

WP4.3 Subcomponent	Interfaces with Subcomponent, WPs	Interfaces / protocols
LPM	GPM	SOAP over HTTP
GPM	VO Manager	SOAP over HTTP
LPM	Policy Requestor (e.g. EMS, SLA-Controller…)	SOAP over HTTP

Table 8: Interfaces for the Policy Manager

6.3.6.4. Involved technologies

The interactions between elements collected inside the policy manager architecture involve Web Services and repository to maintain data. So, the main technologies that could be involved are:

- Standard driver to connect databases (ODBC, JDBC…).
- Transport technologies (HTTP, SOAP over HTTP, SOAP over UDP...).
- Policy Description Language (XML, LDAP filter…).
- GUI framework to define Policy Editor

6.4. Application Support Services Layer Architecture

The GASS layer provides a set of subsystems that support the provision of applications to the final users and the collaboration between the entities involved in the provision process. We can identify three relevant roles that can be played by these entities:

- Service Provider: this entity provides services (in this context everything is virtualized as a service, then a service can be a physical resource, software, a device,…)
- Service Customer: this entity buys service use in order to aggregate them to build a complex application or even just directly use them.
- Service User: this entity actually will use, or interact with, the services. It can be the service customer itself but this is not mandatory.

In the GASS layer, the business relationship between these entities is managed through the Base Virtual Organization and Operative Virtual Organization concepts.

Definitions:

- ***BVO***: this is a "static" VO that lives while at least one SP belongs to it. It can be thought as the marketplace where SP and Customers meet together to sell/buy services.
- ***OpVO***: this is a "dynamic" environment that lives while a service instance is running. It can be thought as the environment that allows the execution of the application and its invocations by the user

The following figure explains the relation between BVO and OpVO

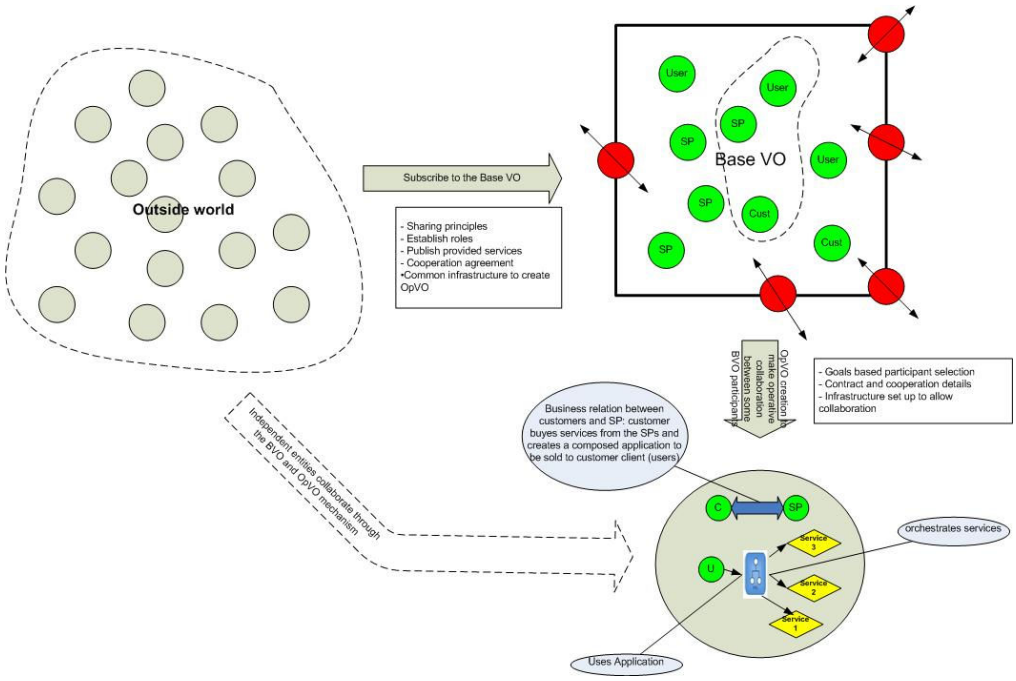

Figure 9: From BVO to OpVO

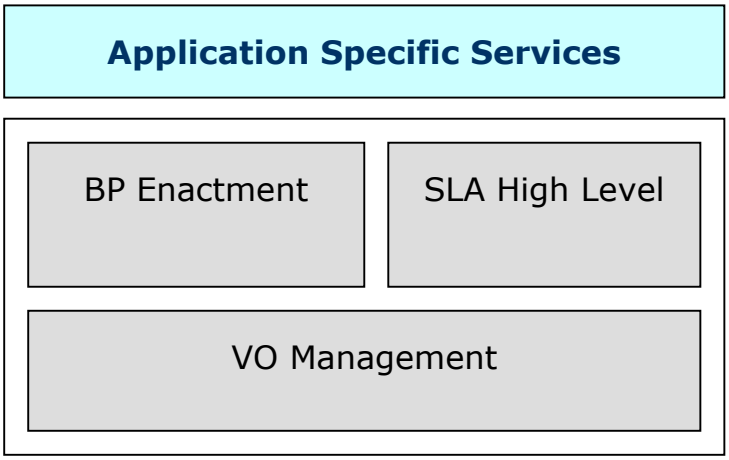

Figure 10: WP4.4 Main Components

In the figure above, three main management areas have been identified: VO Management (VOM), BP Enactment (BPE) and SLA High Level (SLAHL). These layers provide core functionalities supporting the execution of applications foreseen in Akogrimo environment.

The Application Specific Services represent functionalities specialised for developing the application software for several devices, enabling the interactions with Akogrimo VO. Cross communication among components belonging to this layer is needed in order to access their functionalities, e.g. communication between components within VO Management (VOM) and SLA High Level (SLAHL) in order to enlist a new service provider into a VO.

In the following sections we are going to describe each subsystem with respect to the objectives, functionalities, relations and involved technologies.

6.4.1. VO Management

6.4.1.1. Objectives

The VO management subsystem has the objective of providing capabilities to manage BVO and OpVO, which are key concepts in Akogrimo because they make possible the existence of Mobile Dynamic Virtual Organizations (MDVO). This subsystem can be considered a core component of Akogrimo insofar as every business relationship in Akogrimo lives inside a MDVO.

Having in mind this overall objective, the VO management subsystem has the goal of allowing the following two set of basilar operations respectively against the BVO and the OpVO:

- **BVO operations**:
  - Subscription and Registration
  - Log-in and Log-out
  - Publishing
  - Acquire a service
  - Leaving the BVO
- **OpVO operations**:
  - Create the OpVO
  - Join/Leaving the OpVO
  - Using a service
  - Destroy the OpVO

6.4.1.2. Components and functionalities

In order to address its objectives, the VO management subsystem includes a set of modules that interact between them providing the above capabilities.

User Agent: User agent is a service that acts on behalf of the VO participant inside an OpVO. It is a general mechanism which can check/grant access rights at OpVO level. Actually the same mechanism can be applied for the BVO as well. The User Agent is a WS-Resource created for each specific user and it will provide the following functionalities:

- Authenticate user to access the BVO/OpVO
- Check/Grant rights to invoke services inside the BVO/OpVO
- Interacts with services inside the BVO/OpVO on behalf of the external entity

Service Agent: A service agent has similar behaviour to the User Agent. It acts on behalf of the OpVO/BVO participant who is outside the OpVO/BVO itself:

- Checks/grants rights to access services outside the OpVO
- Interacts with external services

BVO Manager: The VO Manager essentially covers functionalities for registration of users and services. Furthermore, it is the high level coordination entity of the BVO. It will be accessible through the Web Service paradigm. The BVO manager provides functionalities to:

- Subscribe to the BVO, that is registering a new participant to the BVO providing him/her with the rights to access the BVO.
- Authenticate registered BVO participants to allow them using BVO services.
- Manage the publications of new "business services"[1] by Service Providers that have the rights to ask for publication.
- Un-register participants that unsubscribe from the BVO.
- Start the service acquiring and OpVO creation processes
- Destroy an OpVO

OpVO Manager: it is the high level coordination entity of the OpVO. For each OpVO we will have a dedicated instance of the OpVO manager that will manage:

- Creation of the OpVO interacting with the OpVO broker
- Generation/Validation of token for accessing the OpVO
- Create User Agents to access the OpVO
- Setting the UserAgent with the rights to access OpVO services
- Information about the status of the OpVO (under creation, created, active, working, to be destroyed,…)

OpVO Broker: it is involved in the OpVO creation phase. It will interact with other services to choose the best services to be involved in the OpVO.

- For each service involved in the OpVO, it searches, on the basis of the service requirements, for a possible SP to be candidate for providing the service.
- Start negotiation with the identified SP to agree on a contract for using an instance of this service
- Create the Service Agents that will act inside the OpVO on behalf of the negotiated services. The components inside the OpVO (e.g. the workflow engine see 6.4.2) will call the Service Agent instead of the real service.

Participant Registry: Stores information about the BVO participant. For each participant the Participant Registry will have a dedicated WS-Resource in order to store information about:

- OpVOs joined by the participant
- Tokens to access the related OpVO
- User Agent references associated to the participant
- Participant profiles related to a specific OpVO.

Figure 11 shows these components and how they are related between them.

[1] With the term business service we mean a service provided by a Service Provider that offers business functionalities (it is not part of Akogrimo infrastructure). For example, a service of the eHealth application is a "business service" (e.g. ECG data generator)

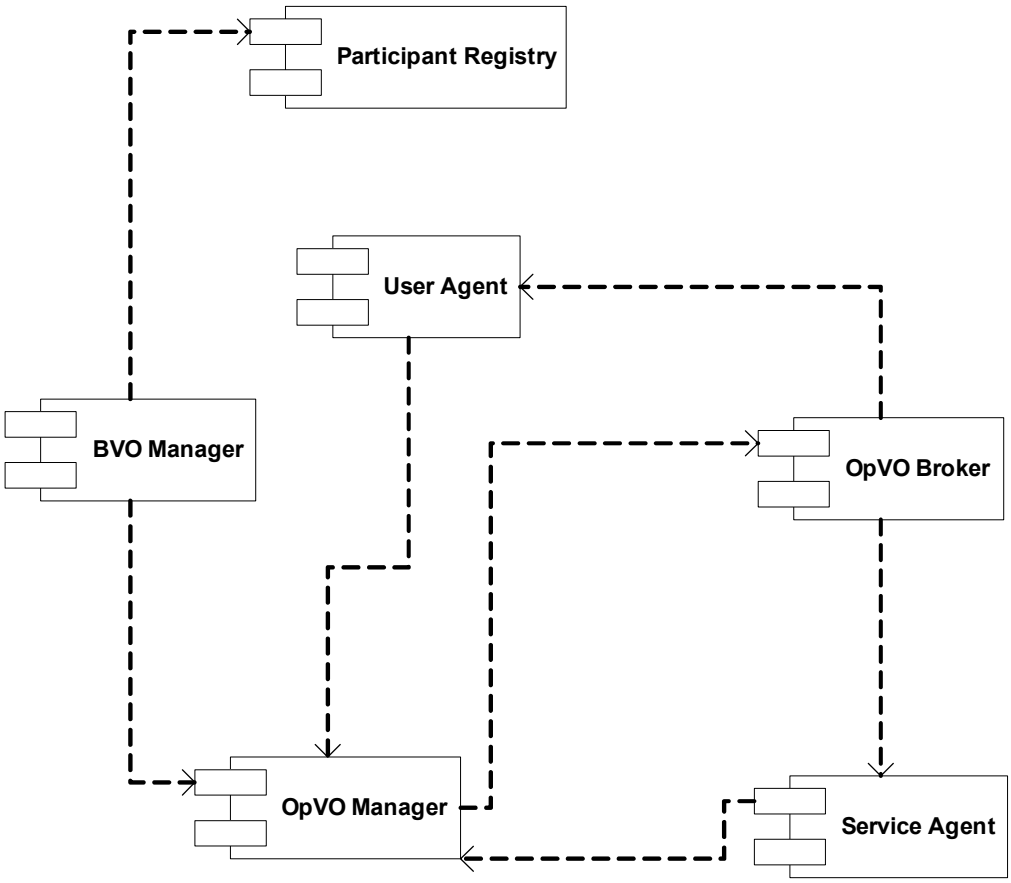

Figure 11: VO management modules Overview

In Figure 11, we have described the static vision of the VO management subsystem, underlying the dependences between the components using standard UML notation.

In section 6.4.1.3 we are going to sketch the interfaces and the interactions between these modules and external ones.

6.4.1.3. Interfaces and interactions with other components

In the following Table 9 a brief description of external interfaces is given. In particular details about invocation of interfaces exposed by subsystems outside the Grid Application Support Service layer are provided. When there is an invocation from a component that is not part of WP4.4 to the module in the first column, this interaction will be listed, as well, specifying that it is from <outside subsystem name> to <module name>.

Invocation to / Module	Subsystem Name	Part of	Purpose of invocation	Method of connection
User Agent	From **External** (to UA)	Outside user application	The UA exposes a generic "invokeApplication" method that allows specification of the method and service to be invoked inside the OpVO. The UA will forward the invocation to the right service	SOAP over http. The UA is a WS-Resource
Service Agent (SA)	**External Business Service**	Outside business services	The SA invokes business services (WS-Resource or Web Service) hosted in external administrative domains	SOAP over http. The SA invokes WS-Resource and WS
OpVOBroker	**Grid Service Discovery Server (GrSDS)**	WP4.2	Search a service Note: Each service (simple and complex) should be exposed as service with its interface	SOAP over http
BVO Manager	**A4C**	WP4.2	To confirm user authentication Retrieve user profile	Diameter
	From External (to BVO Manager)	Outside	• Log-in to the BVO: the BVO Manager authenticate the BVO participant • Retrieve available application for the specific participant • Publish a service for a SP	SOAP over http.

Table 9: Interfaces and interactions of VO management

6.4.1.4. Involved Technologies

Technology/ Specification	DESCRIPTION
XML, SOAP, WSDL	Because of every module will be Web Service the basilar technologies to manage WS will be involved in the implementation, in particular, tools that allow automate processing of these specification will be used (e.g. Axis)
WSRF	The most of modules of VO management subsystem will be implemented as WS-Resource compliant with the WSRF specification (in any case they will be Web Service) Depending on the platform on which these modules will run they will be implemented using two available frameworks: • WSRF.NET: it is the implementation in C# language and on Microsoft .NET framework of WSRF specification running on Microsoft OS. • GT4core: it is the implementation in Java programming language of the WSRF specification. It is installable both on Microsoft (but it doesn't use .NET framework) and Linux OS.

Technology/ Specification	DESCRIPTION
XML native database	Most of the information will be stored in XML format (e.g. Participant Registry). Iin order to simply the management of this info XML native databases will be used, in particular, XIndice
WS-Security	In order to pass security information (token, cryptography) between Web Services, the WS-Security specification will be used and a tool that allows processing of the SOAP headers (e.g. Microsoft Web Service Enhancement)

Table 10: Involved Technologies

6.4.2. Business Process Enactment (BPE)

6.4.2.1. Objectives

A business process represents a set of one or more linked procedures or activities that collectively realize a business objective goal, normally within the context of an organizational structure defining functional roles and relationships. The business process is available at application layer and the final consumer wants to buy and use it. Behind the business process there is a workflow that represents the automation of the business process. The workflow coordinates and manages component services or entities involved in the automation of business process. The view of business process is a high level vision (application layer), but at a low level we need to map it into a service[2]. In this vision we need to deploy the service inside the registry service, using templates that identify the service features. By service features we mean at least QoS and the providers that are able to provide it. From the accounting and charging view point, features about resource consumption and SLA violation management should also be inserted.

Inside an Akogrimo MDVO, an application is the result of an orchestration of existing services that a SP sells to a MDVO customer: note the service sold by a SP can be the result of an orchestration in its turn.

The objective of the BPE subsystem is to implement this behaviour in the frame of the BVO concept and, in particular, in the OpVO. In fact the OpVO is the place where a business process will be executed in order to address the requirements of a customer. Inside an OpVO, through the BPE subsystem, different services will be orchestrated and the result will be the provision of the complex functionalities of the business process. For example, if we refer to the eHealth scenario, several services (ECG Data Analyzer, ECG Data Generator, ECG Data Visualizer,…) will be orchestrated by the BPE subsystem in order to implement the eHealth crisis management scenario.

The above is the position of the BPE subsystem with respect to the overall Akogrimo scope and in order to achieve it we have identified the following set of capabilities that should be provided by the BPE subsystem:

- Selection of a business process template on the basis of the business objective goals
- Selection of services to be involved in the business process

[2] We would like to underline that at VO level we manage services (Web Services or WS-Resource). The Application Support services layer will interact only with services

- Automation of the business process in a workflow
- Instantiation of the workflow
- Workflow execution and management of its lifecycle
- Workflow adaptation depending on the changing context of the mobile/nomadic parties involved in the business process or failures of the services involved in the business process.
- Workflow instance destruction.

6.4.2.2. Components and functionalities

In order to address its objectives, the BPE subsystem includes a set of modules that interact between them providing the above capabilities.

- **The Workflow Manager:** This component is the actual central unit for processing all the necessary high-level decision-making in the workflow enactment process. The Workflow Manager is the decision-making component of the BPE. It is a WS-Resource associated to each workflow to be executed and provide functionalities such as:
  - Collecting services to be involved during the BP execution
  - Submitting, managing, steering and inspecting a workflow's execution
  - Checking workflow state and notifying to the interested parties.
  - Stopping/Destroying the workflow instance
- **The BP Engine:** This component is directly responsible for the workflow instantiation and execution, and for managing the invocation to the external services (i.e. the Service Agents) by submitting all the necessary requests through the most common Web Service communication protocols, namely SOAP over http. It provides functionalities such as:
  - Instantiate and start workflow
  - Stop and destroy workflow
  - Inspect workflow status
  - Expose the workflow as a Web Service if the workflow activities require it to receive input data during the execution.
- **The Workflow Registry:** This component is just a repository of the published workflow templates that will be stored in some relational database along with annotations necessary for semantically identifying workflows. It will be available as a Web Service and will provide typical functionalities of a registry virtualized through the WS interface:
  - Store and get workflow templates
  - Update workflow template
  - Remove workflow template
- **The Monitoring Daemon:** This is the component in charge of monitoring and tracing all events related either to context changes of an entity involved in the workflow enactment process, or to the SLA failures deriving from the inability of a service of fulfilling the initially agreed Service Level Contract. The WorkFlow manager will be informed of these events and it can then take corrective actions in order to guarantee the correct execution of the workflow. To achieve its goals the Monitoring Daemon offers the following functionalities exposed through methods of a Web Service interface:

- Subscribe to the entity to be monitored (e.g. context manager or SLA violations).
- Unsubscribe from monitored entities.
- Provide events to be managed.
- Provide information about the number of unmanaged events.

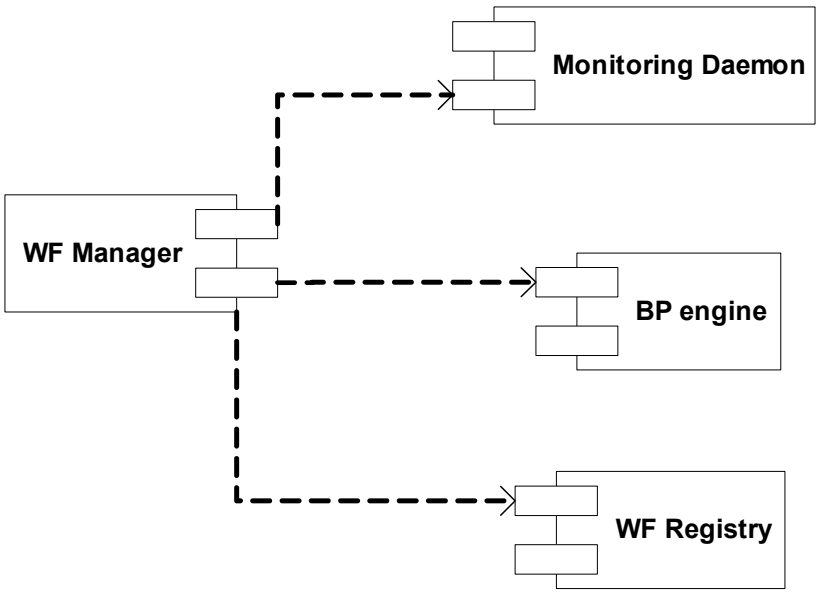

Figure 12: BPE subsystem modules Overview

In Figure 12, we have described the static vision of the VO management subsystem, underlying the dependences between the components using standard UML notation.

In section 6.4.2.3 we are going to sketch the interfaces and the interactions between these modules and the external ones.

6.4.2.3. Interfaces and interactions with other components

In the following **Table 11** a brief description of external interfaces is given. In particular details about the invocation of interfaces exposed by subsystems outside the Grid Application Support Service layer are provided. When there is an invocation from a component that is not part of WP4.4 to the module in the first column, this interaction will be listed, as well, specifying that it is from <outside subsystem name> to <module name>.

Invocation to / Module	Subsystem Name	Part of	Purpose of invocation	Method of connection
Monitoring Daemon	**Entity to be monitored**	Outside business services	Subscribe to the entity to be monitored in order to receive notification about events that need an adaptation of the business process execution. The subscription follows the publish and subscribe mechanism available for Web Service	SOAP over http

Invocation to / Module	Subsystem Name	Part of	Purpose of invocation	Method of connection
	Context manager	WP4.2	This is a particular example of an entity to be monitored; in this case the entity is the context manager. Receive notification about devices and services available in the location of the mobile user.	SOAP over http
Monitoring Daemon	**SLA controller**	WP4.3	This is a particular example of entity to be monitored; in this case the entity is the SLA controller. Receive notification about violation of the SLA related to the services involved in the business process execution	SOAP over http

Table 11: Interfaces and interactions of BPE subsystem modules

6.4.2.4. Involved Technologies

Technology/ Specification	DESCRIPTION
BPEL4WS	In Akogrimo, we are going to orchestrate Web Services. We are going to use the Business Process Execution Language for Web Service to describe workflows that will involve the invocation of several Web Services during the workflow execution.
OWL-S	The workflow template needs to be annotated with semantic information in order to allow the WorkFlow manager to perform advanced search against the WF registry. These semantic annotations will be described using languages such as OWL-S.
BP engine	We are going to design the business process using the BPEL language. There are several business process engines available that interpret this language and execute the related business process. We have chosen to use an open source BP engine: ActiveBPEL.
Relational database	In order to store Workflow templates we are going to use relation database, in particular, MySQL has been selected
WS-Notification	We will use the implementation of this specification to manage the subscription and notification mechanism between Web Service. GT4 core and WSRF.NET frameworks provide implementation of this specification in Java ad C# respectively. The second is based on .NET framework and then it can be installed just on MS OS.
XML, SOAP, WSDL, WSRF	For this technology/specification the same consideration of VO management subsystem are valid here as well.

Table 12: Involved Technologies

6.4.3. SLA High Level Services

6.4.3.1. Objectives

The Akogrimo MDVO requires the management of business relationships between those who provide services and those who acts as customers of this provider. In order to establish this relationship it is necessary to have a contract between the involved parties. The SLA High Level Services (SLAHLS) subsystem provides the capabilities to manage the SLA contract definition phase that will underly the identification of the QoS agreed between the SP and his/her customer. This final contract will include information such as service usage conditions (e.g. start time, end time), charging mechanism (e.g. flat, based on resource use) and violations management (e.g. apply a penalty to the SP if the service execution is not compliant with the agreed QoS).

We have to underline that the SLAHLS subsystem is part of an overall SLA management subsystem that is a set of two main groups split between WP4.3 and WP4.4 layers. Within WP4.4 context, we have the SLAHLS that is basically involved on the negotiation phase of all the services that belong in a workflow execution, as well as the establishment of a contract as a result of this negotiation, while in the WP4.3 we have the SLA Enforcement subsystem which is more focused on the execution control of the service.

In order to achieve the above goal we have identified the following set of capabilities to be provided by the SLAHLS subsystem:

- Creating and storing document templates.
- Allowing document manipulation in a transparent way.
- Performing the negotiation phase.
- Asking for resource reservation on the basis of the negotiation results.
- Dynamically creating the final contract.
- Managing the contract.

6.4.3.2. Components and functionalities

In order to address its objectives, the SLAHLS subsystem includes a set of modules that interact between them in order to address the above capabilities.

- **SLA-TemplateBuilder.** This module is not really a service that interacts with the other ones but it is a tool provided by the subsystem in order to simplify template creation. In fact the SLA-Template will be a document written using a XML based language (hopefully a standard for Web Service) and it will have a well defined XML schema. The SLA-template builder will allow this template to be created through a user-friendly GUI without having to directly manage the XML.
- **SLA-Access**. This is the access point to the SLA-documents and provides the functionalities for getting information into and out of a SLA document. The SLA-Access acts as intermediate service between the external requestor and the other modules that manage the SLA documents. In this way, the internal interactions to manipulate the SLA document (template or contract) will be transparent to the external requestors who will just be aware of the SLA-Access interface.
- **SLA-Translator**. This module is used by the SLA-Access to execute basilar operations on the XML document that represents the SLA contract/template. Actually the SLA-Translator provides low level functionalities to modify an XML document, virtualizing them in a Web

Service interface. In this way, if the SLA template schema should be modified, it will be sufficient to modify a specific method of the SLA-Translator without affecting the SLA-Access code.

- **SLA-Negotiator**. This subsystem is in charge of carring out the negotiation phase between a service consumer and a service provider to reach an agreement. SLA Negotiator allows the individuation, negotiation and reservation (by means of "Pre-reservation" and "Reservation" contracts) of the services to be used during the OpVO execution.
- **SLA-Template**. This is a document based on XML language that describes the SLA metrics, penalties and charging policies associated to a service. It should be based on a standardized language. It is provided by the SP when he asks to publish a service, in order to explain under which conditions the specific service could be sold.
- **SLA-Repository**. Repository where the SLA-Templates of all published services are stored
- **SLA-Contract**. This is an "instance" of the SLA-Template, after the negotiation phase. The SLA-template of a specific service is customized on the basis of the customer and SP agreement. It will also include information not available at the moment of the SLA-Template definition (e.g. when does the service execution have to start?).
- **SLA-ContractRepository**. Repository that stores all established (live) SLA-Contracts.

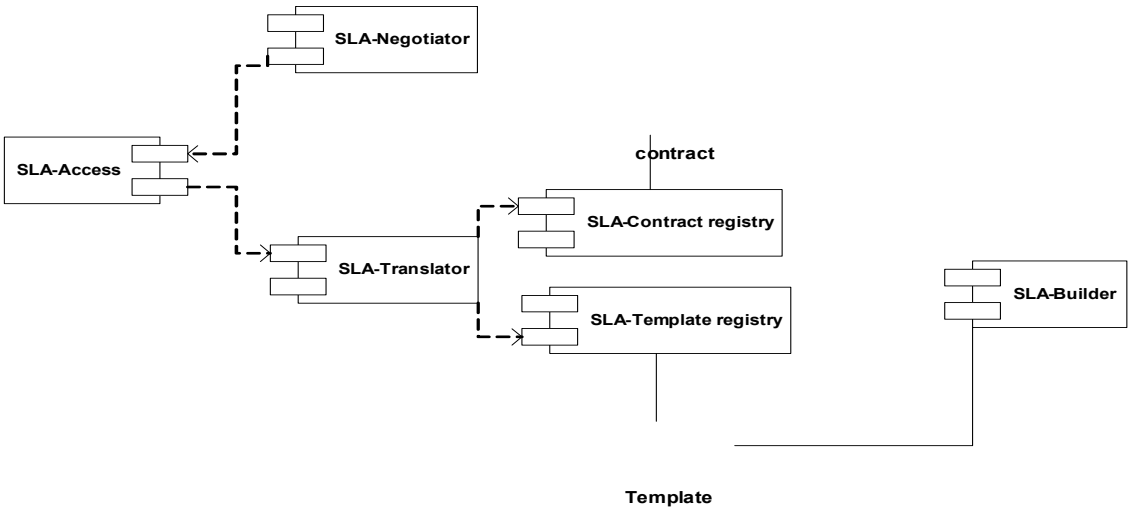

Figure 13: SLA HLS subsystem modules overview

In Figure 13, we have described the static vision of the SLAHLS subsystem, underlying the dependences between the components using standard UML notation.

In section 6.4.3.3 we are going to sketch the interfaces and the interactions between these modules and the external ones.

6.4.3.3. Interfaces and interactions with other components

In the following table a brief description of external interfaces is given. In particular details about invocation of interfaces exposed by subsystems outside the Grid Application Support Service layer are provided. When there is an invocation from a component that is not part of WP4.4 to the module in the first column, this interaction will be listed, as well, specifying that it is from <outside subsystem name> to <module name>.

Invocation to / Module	Subsystem Name	Part of	Purpose of invocation	Method of connection
SLA Access	**From EMS** (To SLA-Access)	WP4.3	The EMS invokes the SLA-Access to retrieve the QoS agreed in a specific contract for a specific service to be run.	SOAP over http.
SLA negotiator	**EMS**	WP4.3	Contacts the EMS to check the availability of a service with the specified QoS. Contacts the EMS to reserve the resource use and to find a final agreement on their use and availability	SOAP over http

Table 13: Interfaces and interactions of SLAHLS subsystem modules

6.4.3.4. Involved Technologies

Technology/ Specification	DESCRIPTION
WS-Agreement	The SLA template and contract will be written using an XML based language. In particular, it we have chosen the WS-Agreement specification that is a reference specification in the frame of Global Grid Forum
XML native databases	The most of information will be stored in XML format (i.e. contracts and template in the repository). In order to simplify the management of this info, XML native databases such as XIndice could be used
XML, SOAP, WSDL, WSRF	Most modules of the VO management subsystem will be implemented as WS-Resources compliant with the WSRF specification (in any case they will be Web Services) They will be implemented on Microsoft platform then it will be used: • WSRF.NET: it is the implementation in C# language and on Microsoft .NET framework of WSRF specification running on Microsoft OS. • C# programming language

Table 14: Involved technologies

7. USE Case Specification

7.1. Context adaptation

Use Case	Context adaptation
Description	During the execution of a workflow dealing with patient monitoring, a doctor is involved to examine an ECG diagram. First, the doctor (Alice) sees the patient's journal and the corresponding ECG diagram on her mobile terminal. To perform the diagnosis she needs to view the ECG diagram on a larger display. Therefore, she enters a room that has a big display supporting ECG visualization. When she enters the room the Context Manager notifies the Monitoring Daemon about the changed context (new physical location and nearby services at this location). The Workflow Engine receives information that a big display is available from the Monitoring Daemon, and adapts its execution by moving the ECG visualization to the big display (that is, the doctor should confirm first).
Actor	Workflow Engine, Monitoring Daemon, Mobile terminal, SIP server, Context Manager, LSDS, RFID server, A4C server
Preconditions	The patient monitoring application is implemented by a workflow template that has been instantiated and deployed. A OpVO has been created to deal with the workflow execution. The workflow template includes a high level description of what context changes that are know to be relevant for the workflow. It is know that the doctor needs a large display to perform diagnosis of the ECG diagram. The Workflow Manager has asked the Monitoring Daemon to subscribe to the Context Manager for context changes for particular user-IDs (alice@akogrimo.com). Upon receiving the subscription from the Monitoring Daemon, the Context Manager has requested the user profile from the A4C server (Akogrimo id and RFID tag id) for alice@akogrimo.com. Context Manger subscribes to SIP presence information for alice@akogrimo.com. Local services has been registered with proper attributes (physical location and services) with the SLP Directory Agent.
Postconditions	<What will be the status of the modules after this case?>
Sequence	Normal flow: 1. A RFID reader detects the new physical location of Alice. The Context Manager is notified of the new physical location. 2. The Context Manager queries the LSDS for local services at the new location of Alice. 3. The Context Manager notifies the Monitoring Daemon about the

	new context of Alice. 4. The Monitoring Daemon analyse the context notification to map to high-level context. It is seen that the context update is of relevance to the workflow. 5. The Workflow Engine is notified of the new context information from the monitoring demon. 6. The workflow copies the ECG data to the big display and invokes the service to visualize the ECG diagram. 7. The doctor performs the ECG diagnosis.

7.2. SIP integration

Use Case	**Grid Call Establishment**
Description	During the execution of a workflow dealing with patient monitoring, the EcgDataAnalyzer detects heart attack conditions in patient's ECG data. The Workflow Engine then decides to put the patient in contact with a doctor via a videoconference. The Workflow Engine contacts the SIP Grid Gateway and provides it with the IDs of the patient and the doctor. The SIP Grid Gateway uses SIP signalling to establish a multimedia session between the two parties, using the SIP Server for proxying purposes.
Actor	Workflow Engine, SIP Grid Gateway, Patient's Mobile Terminal, Doctor's Mobile Terminal, SIP Server, A4C Server.
Preconditions	The SIP Grid Gateway and the Mobile Terminals are registered in the SIP Servers Location Database, so that SIP messages can be correctly routed. The Workflow Engine will provide AkogrimoIDs to identify the patient and the doctor. A SIP URI must be present in the corresponding A4C profiles for the SIP Grid GW to be able to translate the AkogrimoIDs into SIP URIs.
Postconditions	
Sequence	Normal flow: 1. EcgDataAnalyzer analyzes data and detects conditions that suggest the patient is suffering a heart attack, and tells about this to the Workflow Engine. 2. The Workflow Engine contacts the SIP Grid Gateway and asks to establish a call between two given AkogrimoIDs. 3. The SIP Grid Gateway retrieves the A4C profiles associated with the given AkogrimoIDs to get the corresponding SIP URIs. 4. The SIP Grid Gateway uses the SIP URIs to create a REFER SIP message, which is then sent to the SIP Server.

	5. The SIP Server queries the Location Database to get the logical location of the patient. 6. The SIP Server routes the REFER request to the patient. 7. The patient's Mobile Terminal receives the request and contacts the doctor's Mobile Terminal. 8. The patient's Mobile Terminal informs the SIP Grid Gateway about the progress of the request through the SIP Server. 9. When the doctor accepts the incoming call, the multimedia session gets established between the patient and the doctor. 10. The patient's Mobile Terminal informs the SIP Grid Gateway about the success in the establishment (no information is given when the session ends).

Use Case	**Grid Call Transfer**
Description	During the execution of a workflow dealing with patient monitoring, a doctor is involved to examine an ECG diagram, while talking with the patient about his diagnostic. First, the doctor (Alice) sees the patient's case notes and the corresponding ECG diagram on her mobile terminal. To perform the diagnosis she needs to view the ECG diagram on a larger display. Therefore, she enters a room that has a big display supporting ECG visualization and SIP multimedia sessions. When she enters the room the Workflow Engine receives information that a big display is available, and asks the SIP Grid Gateway to move the already established multimedia session with the patient to the big display (that is, the doctor should confirm first).
Actor	Workflow Engine, SIP Grid Gateway, Patient's Mobile Terminal, Doctor's Mobile Terminal, Big Display, SIP Server, A4C Server.
Preconditions	The SIP Grid Gateway, the Mobile Terminals and the Big Display are registered in the SIP Servers Location Database, so that SIP messages can be correctly routed. The Workflow Engine will provide AkogrimoIDs to identify the patient, the doctor and the display. A SIP URI must be present in the corresponding A4C profiles for the SIP Grid GW to be able to translate the AkogrimoIDs into SIP URIs. The doctor is allowed to use the display.
Postconditions	
Sequence	Normal flow:

	1. The Workflow Engine is informed about the proximity of a big display to Alice. 2. The Workflow Engine contacts the SIP Grid Gateway and asks to transfer the call between Alice's PDA and her patient to the big display, providing the needed AkogrimoIDs. 3. The SIP Grid Gateway retrieves the A4C profiles associated with the given AkogrimoIDs to get the corresponding SIP URIs. 4. The SIP Grid Gateway uses the SIP URIs to create a nested REFER SIP message, which is then sent to the SIP Server. 5. The SIP Server queries the Location Database to get the logical location of Alice (i.e. her PDA). 6. The SIP Server routes the nested REFER request to Alice's PDA. 7. The PDA receives the request and send a REFER message to the patient to transfer the call to the big display. 8. The patient's Mobile Terminal contacts the big display and informs the PDA about the progress of the transference. 9. Alice's PDA informs the SIP Grid Gateway about the progress of the request. 10. When Alice accepts the incoming call in the big display, the multimedia session gets transferred to the big display. 11. The patient's Mobile Terminal informs Alice's PDA about the success in the transference. 12. Alice's PDA informs the SIP Grid Gateway about the success in the transference.

7.3. Data management/Execution

Use Case	**Context adaptation**
Description	During the execution of a workflow dealing with patient monitoring, the ECGDataGenerator produces data that should be stored by the Data Manager. Each data stored is identified by a unique file identifier. In another branch of the workflow the data produced by the ECGDataGenerator should be analysed. For this reason data is retrieved by the DataManager and analysed by the ECGDataAnalyser. The data that should be visualised by the doctor needs to be copied from the Data Manager to the ECGDataVisualiser.
Actor	Workflow Engine, ECGDataGenerator, ECGDataAnalyser, ECGDataVisualiser, Data Management.
Preconditions	The patient monitoring application is implemented by a workflow template that has been instantiated and deployed. An OpVO has been created to

	deal with the workflow execution. The name of the Data Manager is known a priori because we do not have service discovery in this phase. The machine that hosts the ECGDataGenerator should run a GridFTP client. The machine that hosts the ECGDataAnalyser should run a GridFTP server. The machine that hosts the ECGDataVisualiser should run a GridFTP server.
Postconditions	
Sequence	Normal flow: 1. The ECGDataGenerator produce a data file containing the information about the patient monitoring. 2. The file is uploaded and stored to the Data Manager. The Data Manager returns a unique identifier for the file. 3. … (other operations not related to this use case) 4. A data file that needs to be analysed is retrieved from the Data Manager. 5. The ECGDataAnalyser analyse the data file retrieved. 6. … (other operations not related to this use case) 7. A data file that needs to be visualised by the doctor is transferred from the Data Manager to the doctor's location. 8. The ECGDataVisualiser displays the data.

Use Case	**Service Reservation**
Description	A client buying the provision of an E – Health service, is asked by the system to provide the desired QoS parameters of the service execution. This way, a new SLA contract is made. The EMS system is responsible of making the proper reservation of resources that will allow the accomplishment of the client's demands, as described in the SLA contract.
Actor	SLA – Access, SLA – Controller, EMS, E – Health Services Factory.
Preconditions	A negotiation phase has taken place so as to determine the desired QoS of the service bought. The SLA – Access system has stored the QoS parameters values in its database. The EMS system is notified at the end of the Negotiation phase about the request of a new service being provided with the new SLA contract id.

	The names of the SLA – Access, SLA – Controller, EMS, and E – Health Services Factory are known a priori since we do not have any service discovery in this phase.
Postconditions	
Sequence	Normal flow: 1. EMS acquires the id of the new SLA contract. 2. EMS retrieves the QoS parameters from the SLA – Access. 3. EMS creates an E – Health service instance for the service provision to the new client. The machine where this instance is created is selected on the basis of the QoS request of the client: EMS selects a machine the not reserved resources of which can accomplish all the QoS requirements of the client. 4. It creates a new SLA – Controller instance that will be controlling whether the QoS described in the SLA contract is accomplished. 5. EMS stores the EPR's of the new E – Health service and the SLA – Controller in its database.
Use Case	Service Execution Management
Description	Each client buys the provision of a service for a specific time slot. The EMS is persistently controlling whether the start or the end time of a service bought has encountered so as to take the proper actions. In detail, when if the start time of a service provision has come, it activates the designated SLA – Controller instance and it notifies the Monitoring Service to start monitoring the quality of this service provision. When the end time of the service provision has arrived, EMS destroys the designated SLA – Controller instance as well as the E – Health service instance and notifies the Monitoring Service to stop monitoring this service provision.
Actor	SLA – Controller, EMS, E – Health Services Instance.
Preconditions	The proper reservations for the service provision have taken place.
Postconditions	
Sequence	Normal flow: 1. EMS activates the designated for this service SLA – Controller instance for the new service that must start being provided (according to the SLA contract associated with it). 2. EMS notifies the Monitoring service to start monitoring the achieved QoS of this new service provision. 3. EMS waits until the service provision time –slot of this service ends.

	4. EMS destroys the E – Health service instance. 5. EMS destroys the SLA – Controller instance. 6. EMS notifies the Monitoring Service to stop monitoring this service.

7.4. OP Vo creation

Use Case	OpVO creation
Description	A participant of the BVO would like to have the availability of a service that will address a specific business goal. To do that it contacts the BVO manager that will search for services inside the BVO or for workflow that address/implement the specified requirements. If the BVO is able to satisfy the request a dedicated environment has to be created in order to make available the identified service, this is the OpVO. The creation of an OpVO involves several steps. The process is started by the BVO manager that will create an instance of an OpVO manager that from now on will manage the new OpVO. The OpVO has a perimeter that will enclose: • An instance of the business process (bp) that executes the business goal required by the BVO participant. • A workflow manager that will manage the start, stop, corrective actions, destruction of the bp instance • User agents that will be the access point to the OpVO. Any request from OpVO users has to be passed to the UA that will forward it to a service inside the OpVO. • Service Agents that will be the entity that acts on behalf of external service inside the OpVO. Every invocation to an external service will be forwarded to the related SA inside the OpVO that will have the role to invoke the external service. • A monitoring daemon instance that will monitor events of interest in order to start adaptive actions during the workflow execution. The perimeter of the OpVO is considered secure because all the entity inside the OpVO will share a security token. In order to have communication from/towards the outside world, UAs and SAs will be created to act as an interface.
Actor	BVO Manager, GrSDS (to search services to be involved in the workflow execution), OpVO Manager, OpVO Broker (to start the negotiation use case), Workflow Manager, SA, UA
Preconditions	The requestor has the rights to ask for a business goal. The required business goal is achievable by the BVO (through a simple service or an existing business process template). The negotiation of identified service to be invoked during the workflow

	execution is successful.
Postconditions	A UA resource instance exists to be invoked by an external user. The workflow instance is ready to be invoked for starting the flow execution. The SA instances exist to be invoked in order to contact services provided by external entity. The management services (OpVO, WF manager, monitoring daemon) are set to manage the OpVO during the execution (take corrective actions, add new user agents, validate tokens,…)
Sequence	Normal flow of the OpVO creation process: 1. Search of a workflow template that addresses the business goal 2. Instantiation of an OpVO manager 3. Instantiation of a WF manager 4. Search for service to be involved in the bp execution 5. Finalization of the negotiation use case for each service that has to be invoked during the bp execution 6. Instantiation of a SA that will acts as a proxy for each external service involved in the bp execution 7. Instantiation of a business process that addresses the required business goal 8. Instantiation and set up of services necessary to manage the business process (Monitoring Daemon) 9. Instantiation and set up of the UA

8. Selected key Scenarios

In this chapter we present the following scenarios:

- Akogrimo Log-In
- Service Invocation and Charging
- Base VO Registration
- OpVO Creation
- Context Sensitive Services

8.1. Akogrimo Log-In

The Akogrimo Log-In scenario demonstrates four important features of the Akogrimo architecture:

- Authentication & Authorisation across organisational boundaries
- Network QoS setup
- SIP presence announcement
- Returning a Base VO reference to the user's terminal
- Returning a public customer ID to the user's terminal

Trust relationships between different network operators enable the information exchanged between A4C servers in different domain, thereby enabling cross-organisational authentication, authorisation and accounting.

Right from the start the quality of service parameters are adjusted in compliance to specified network policies.

As soon as the network connection has been established the presence of the user is communicated to a SIP server. This presence announcement makes the user available as service consumer and service provider. The system can react on the user's presence status e.g. by delivering waiting messages or resuming pending workflow actions.

The Base VO reference that is transmitted to the user's terminal during the login procedure is the user's link to their service delivery platform, the virtual organisation. The public customer ID is the identifier of the user in the domain of the customer. Customer's domain and Base VO domain have a trust relationship to enable the link between the Base VO domain and the user's domain.

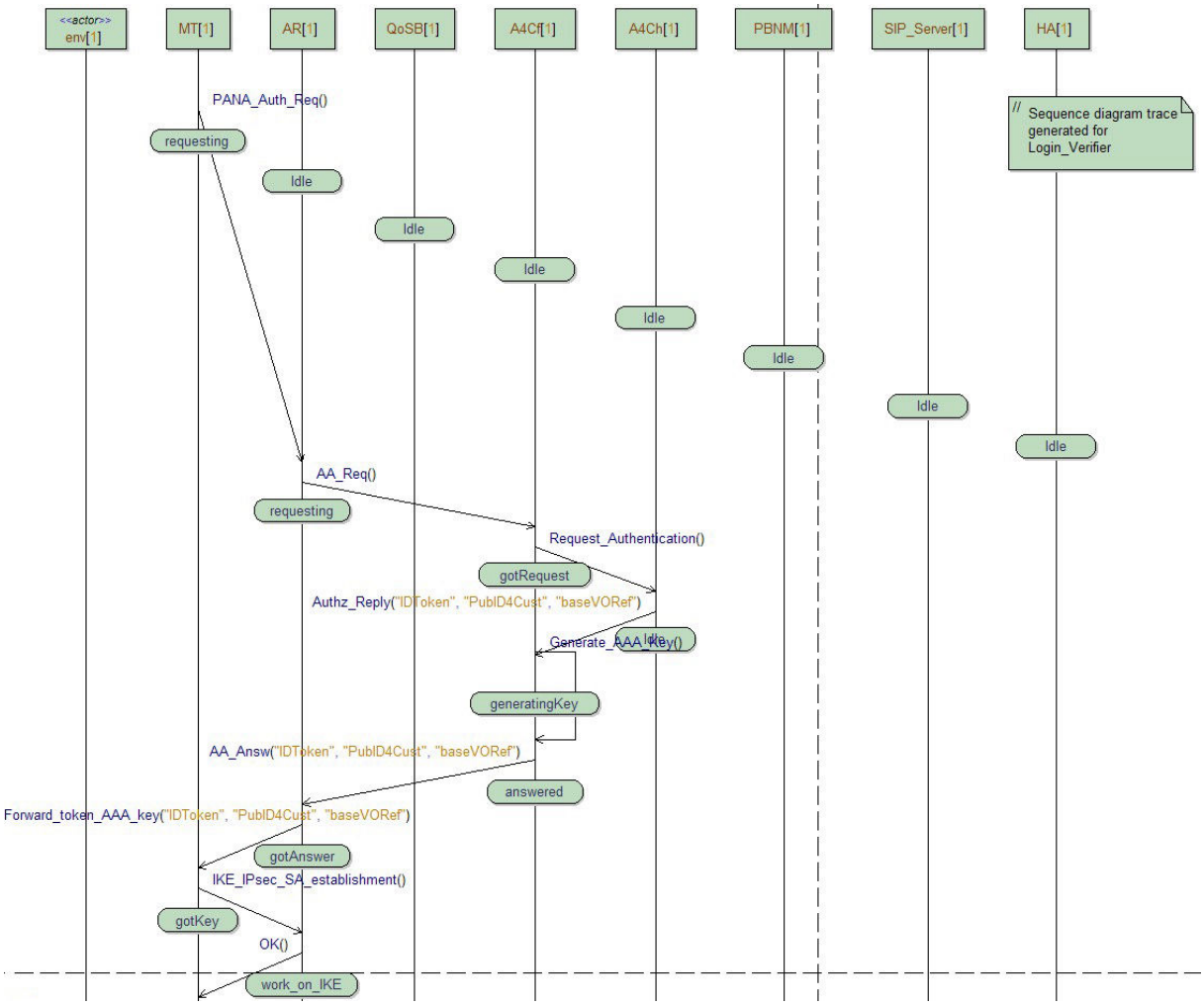

Figure 14: Login Sequence (1)

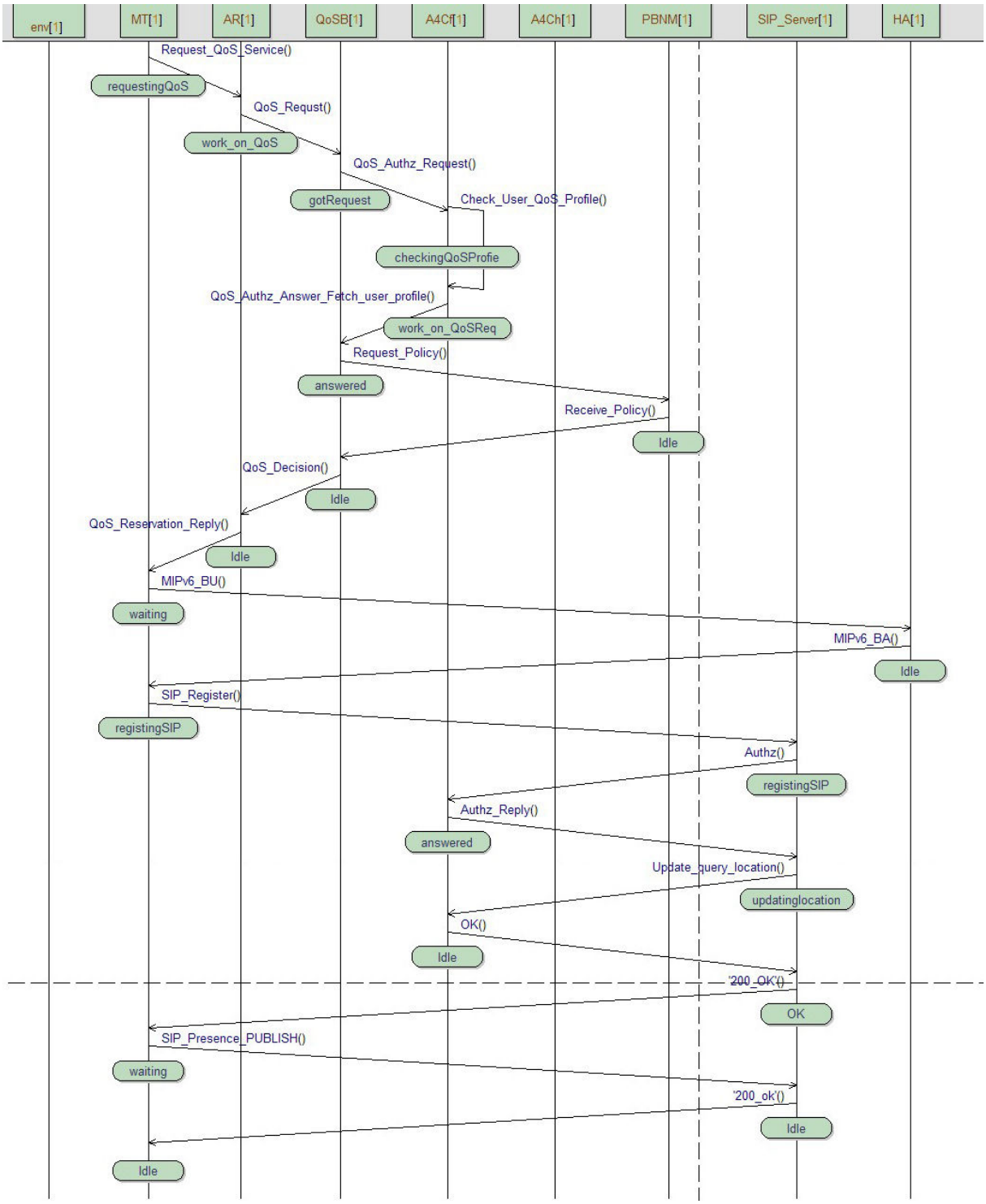

Figure 15: Login Sequence (2)

8.2. Service Invocation and Charging

The following sequence diagram demonstrates the invocation of a service by the user and the charging of the service by the A4C.

The preconditions are:

- The user is already a member of the Base VO and has an identifier (public ID) in the customer's domain.
- The user and the customer belong to different domains but a trust relationship has been established between their domains.
- The customer and the Base VO domain have enabled the trust relationships

In this scenario the following actions are shown:

- The user logs-in
- Invocation of the service through the Base VO
- Verification of the IDToken (cross-verification)
- Retrieval of the user profile
- Charging of the service (cross-charging)

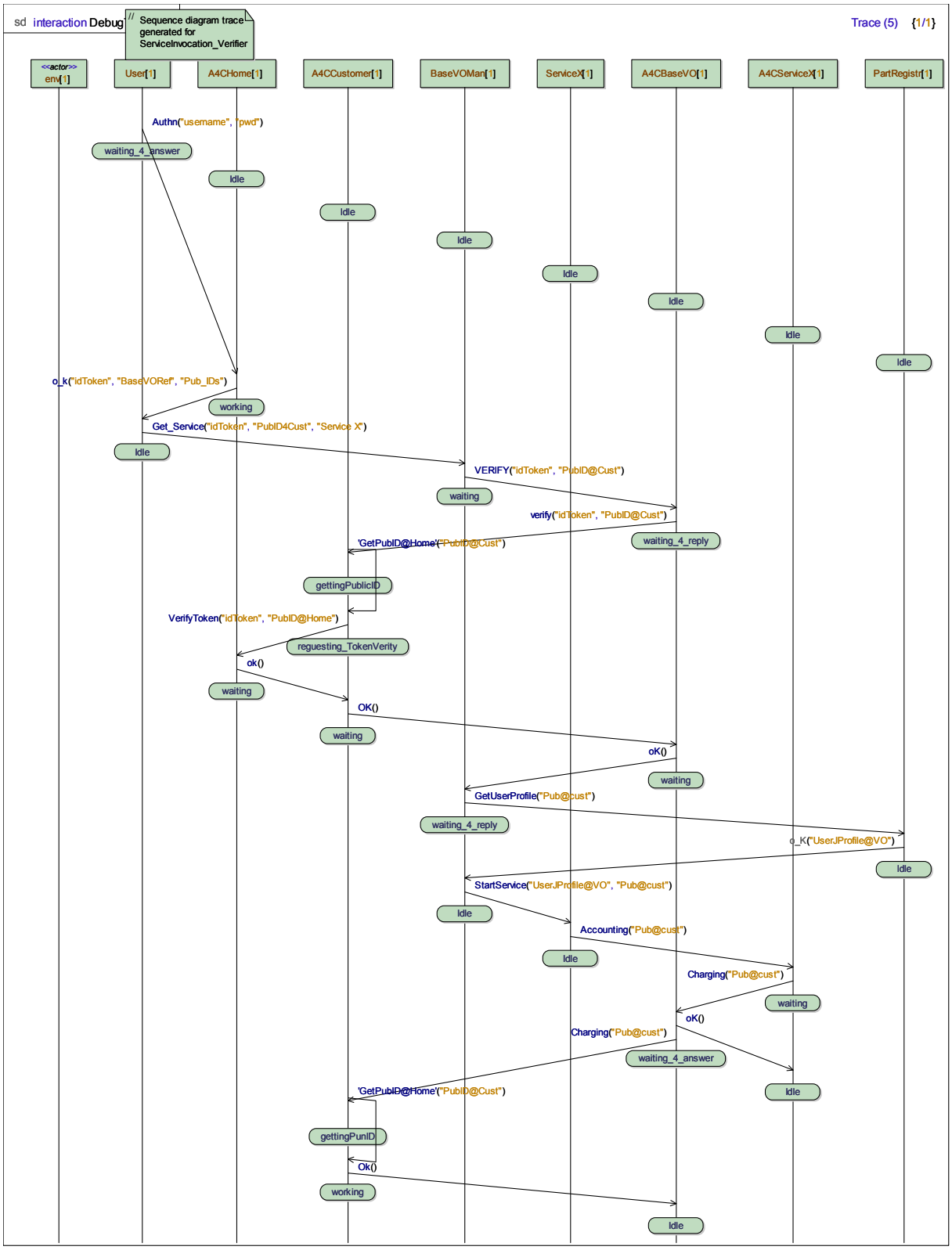

Figure 16: Service Invocation and Charging

8.3. Base VO Registration

The Base VO Registration demonstrates the registration of the user by the customer. The preconditions are:

- The user and the customer are at the same domain (they may come from different domains, but for simplicity this is not shown in the diagram)
- The customer and the Base VO domain have enabled trust relationships

The phases at the sequence diagrams are:

- Generation of the new identifier "publicID" for the user at the Base VO
- Verification of the IDTokens from the user and customer
- Provision of the specific user profile to the Base VO,
- The user profile is approved under the conditions of the Policy Manager

The post conditions are:

- The user is a member at the Base VO
- A new pair (PublicID, profile) is stored at the Participant Registry of the BaseVO

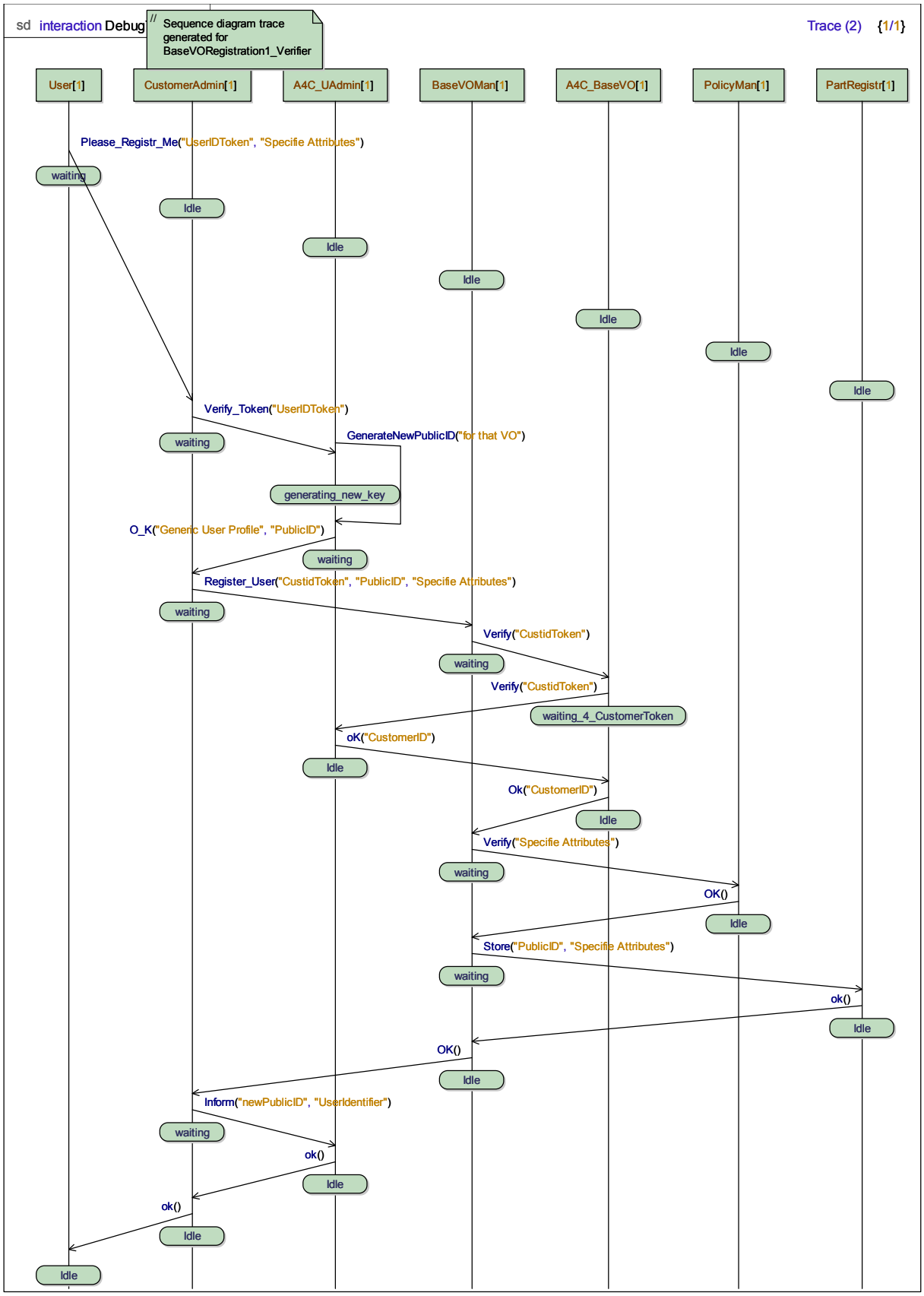

Figure 17: Base VO Registration

8.4. OpVO Creation

An OpVO is the operative environment for service delivery in Akogrimo. The creation phase involves a sophisticated search and negotiation procedure to find and bind the required services according to the customer's request. In the scenario, the requested service is implemented by a workflow. This scenario demonstrates:

- Service selection by the user
- Negotiation of the service quality
- Contract establishment
- Service deployment

The Base VO Manager offers a list of applications, e.g. patient monitoring services. The details of these applications and the quality of service can be specified by the customer. This customer created SLA is used in the service discovery to find the most suitable services in the Grid Service Discovery Server (GrSDS). The GrSDS has to be able to understand the SLA parameters and match them with parameters of registered services. In case a service is represented by a workflow, the SLA-Subsystem of the Base VO negotiates high level service parameters with the customer.

The OpVO Manager is created after the workflow service is selected. The OpVO Manager handles the actual instantiation of the OpVO. The OpVO Broker is responsible to find services that fit into the workflow description from functional and non-functional point of view. Also the involved human actors, e.g. a teacher who is part of an eLearning business process, establish a contract with the OpVO by means of their SLA component. After all services and actors that are involved in the business process have been found, the workflow is deployed into the Enactment Engine.

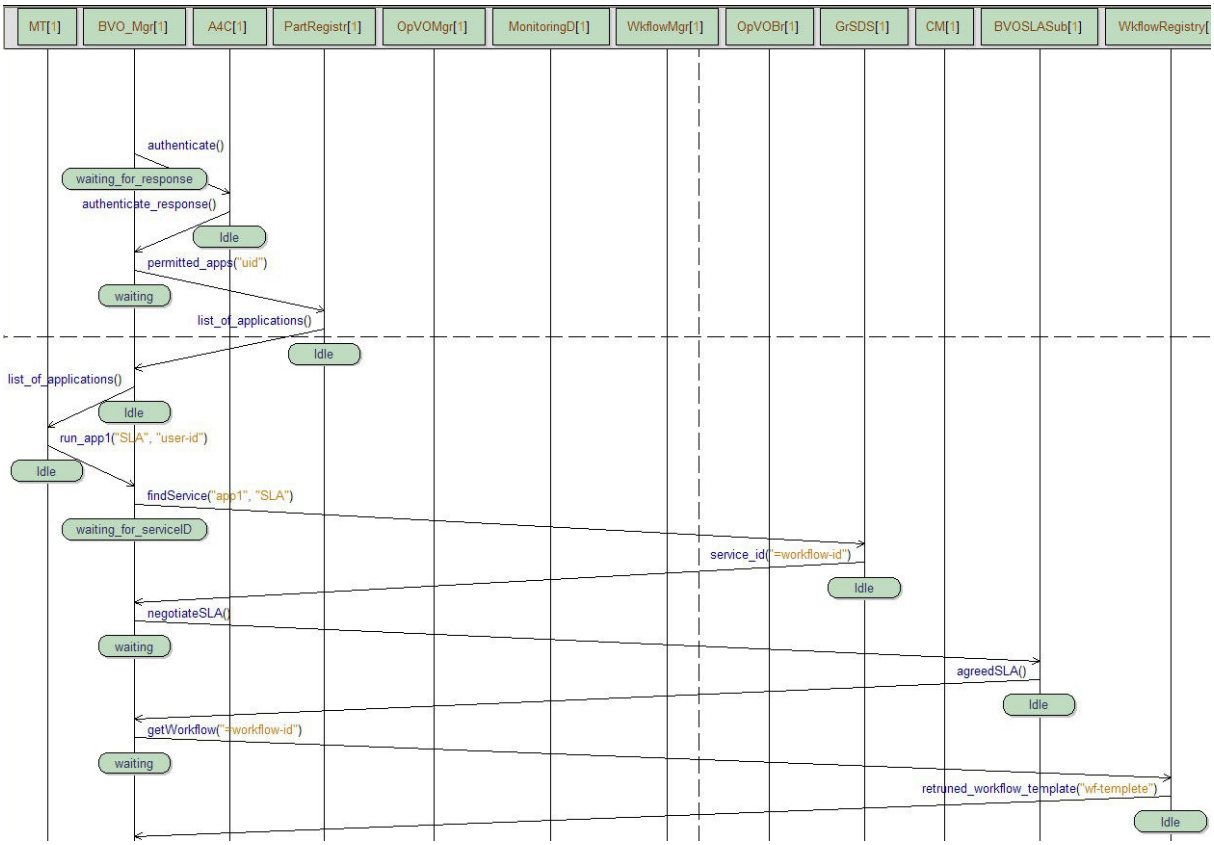

Figure 18: OpVO Creation (1)

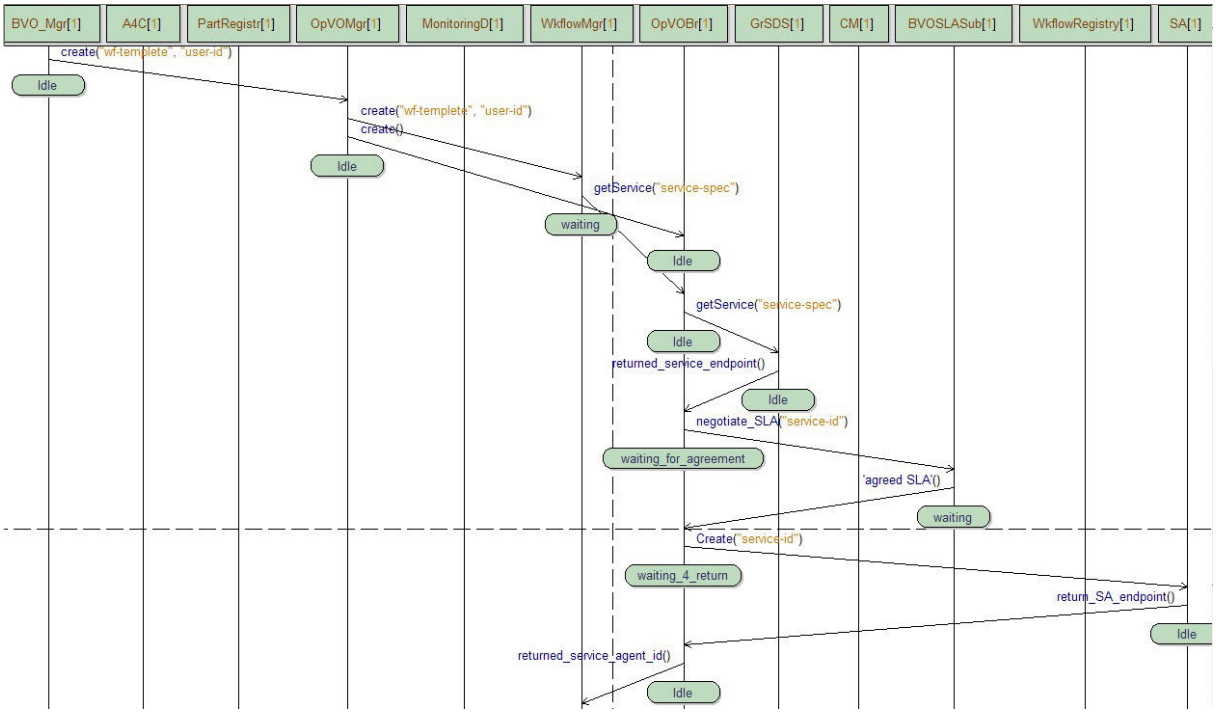

Figure 19: OpVO Creation (2)

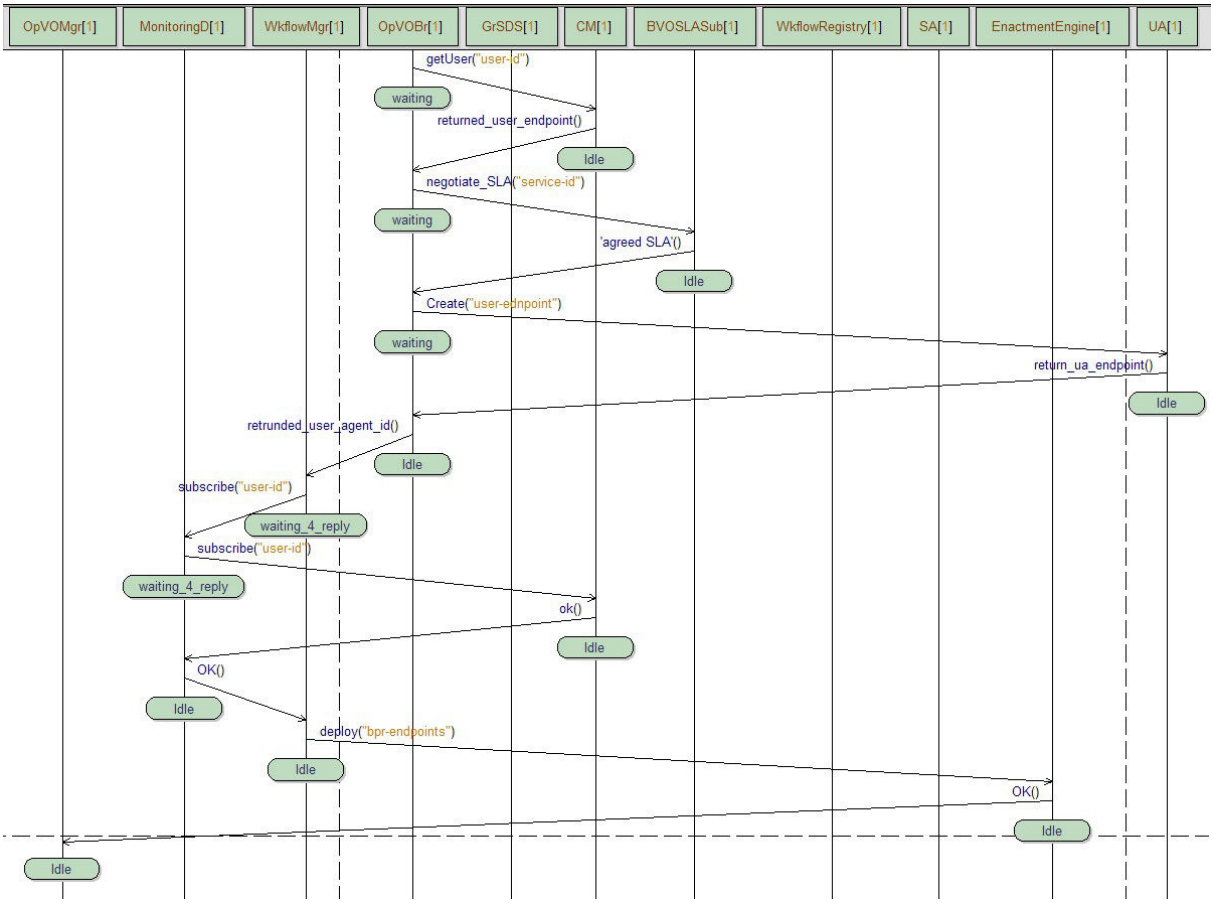

Figure 20: OpVO Creation (3)

8.5. Context Sensitive Services

One of the key features of Akogrimo is that information can not only flow from the service provider to the consumer, but also from the service consumer to the service provider, thereby enabling context aware services that are tailored to the user's needs. Adaptation of service qualities can be done at different levels. E.g. a video service might directly query the context of the user and adapt the delivered video stream to the user's device. More complex adaptations can be achieved by context sensitive workflows. The following scenario demonstrates how this can be achieved with the Akogrimo architecture.

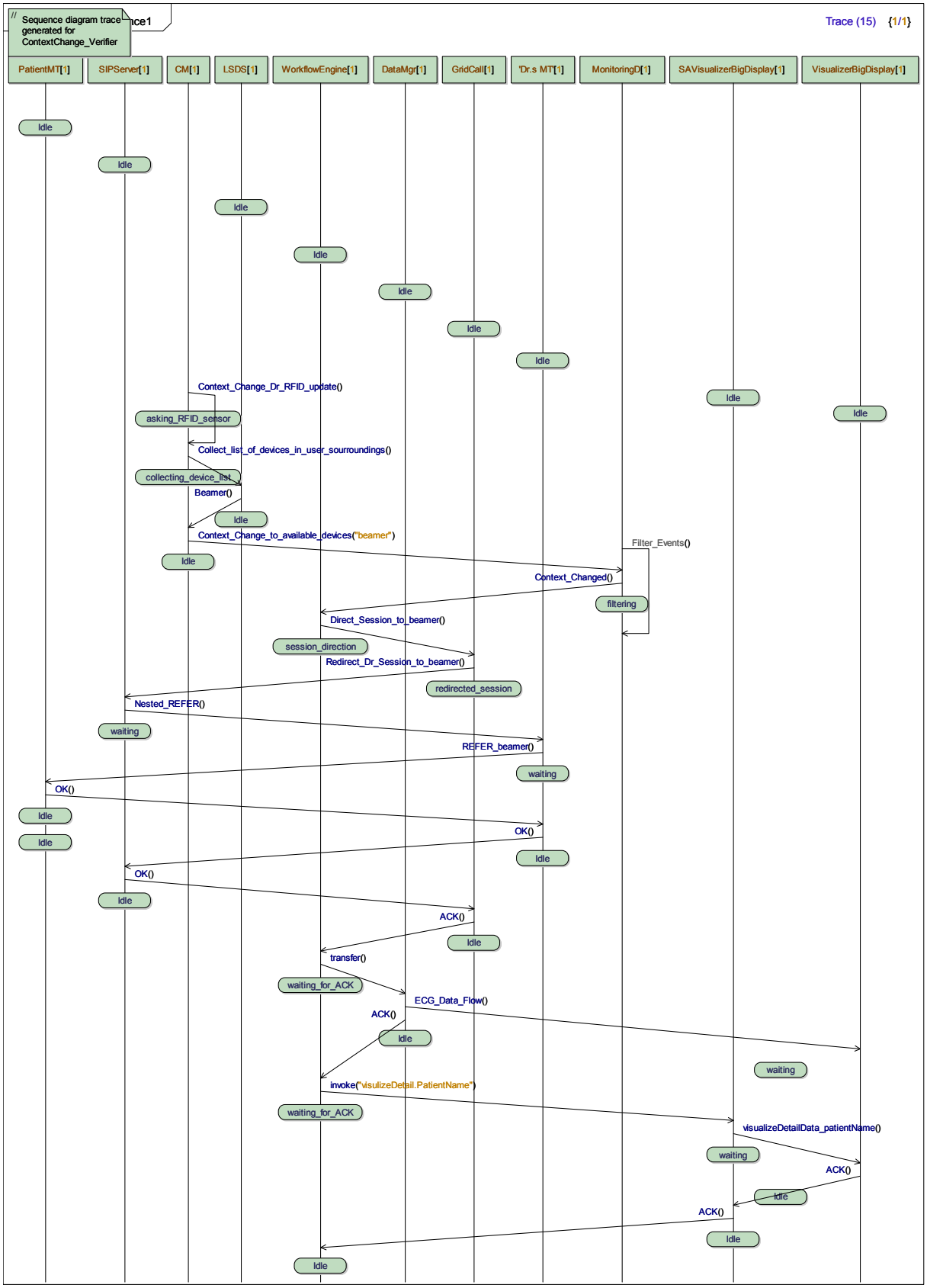

Figure 21: Context Sensitive Workflow

A context change is signalled to the Context Manager by sensors on the doctor's terminal or by sensors that are placed in the environment. The Context Manager then gathers additional

information about the user's environment, in our case, by asking the Local Service Discovery Server (LSDS) to provide a list of local services. The context information is sent to the Monitoring Daemon. The Monitoring Daemon has the ability to filter context information and can trigger appropriate actions in the workflow engine. In our case the doctor's audio/video communication with a patient is transferred to a bigger display and the Data Manager is informed to send the ECG data to a visualizing service on that big display.

9. Conclusion and Outlook

This Deliverable should be regarded as a baseline document for the whole Akogrimo project, with the assumption that the presented architecture is for the time being final, but will be enhanced in subsequent activities. The Deliverable summarises the work during the first eighteen months of the project and includes a description of generic operational scenarios.

The Akogrimo Architecture has three goals which are orthogonal: First, the Internet Protocol is regarded as convergence layer for a mobile Grid infrastructure, which relies on Internet-like AAAC concepts for commercialising the composed services. This requires a lower layer technology independent solution with the goal of providing seamless mobility compared to that of existing networks. Second, Akogrimo will manage virtualized resources of the network layer efficiently. Third, the whole Akogrimo concept will be targeted towards a commercial environment and all the required mechanisms will be added around the IETF AAA concept

This Document describes the second iteration loop of the Akogrimo Deliverable and is based on Akogrimo Deliverable D3.1.1, but with refinements represented by use cases of chapter 7 and further by the MSCs given in chapter 8. The selected key scenarios are seen by the project as key scenarios showing the interworking between the network world – as represented in chapter 6.1 and 6.2 – and the grid world – as represented in chapter 6.3 and 6.4.

10. REFERENCES

[Dey01] A. K. Dey, Understanding and Using context, Personal and Ubiquitous Computing Journal, Volume 5 (1), 2001, pp. 4-7.

[D2.2.4] J. Gallop et al: "Report on the state of the art", Akogrimo Deliverable D2.2.4, v1.1.

[D4.2.1] J. Wedvik et al: "Overall Network Middleware Requirements Report". Akogrimo Deliverable D4.2.1, v 1.1. URL: http://www.akogrimo.org/modules.php?name=UpDownload&req=getit&lid=39

[D4.2.2] P. Osland et al: "Final Integrated Services Design and Implementation Report", Akogrimo Deliverable D4.2.2, v1.0, URL: http://www.akogrimo.org/modules.php?name=UpDownload&req=getit&lid=42