SEVENTH FRAMEWORK PROGRAMME SEVENTH FRAMEWORK PROGRAMME: PRIORITY 7.1B LARGE SCALE INTEGRATING PROJECT (IP)	IP project number 247950 Project duration: February 2010 – February 2014 Project coordinator: Joe Gorman Project Coordinator Organisation: SINTEF, Norway Strategic Objective: 7.1.b website: www.universaal.org UNIVERSAAL Universal Open Architecture and Platform for Ambient Assisted Living

Document Type:		
"Deliverable:" Item Appearing in "List of Deliverables in DoW with delivery date shown in bold "Supplementary Report" As "Deliverable", but delivery date *not* shown in bold. These documents are formally internal to the consortium, but can be delivered on request.		Project Deliverable, with independent sub-parts. *Each sub-part forms a coherent whole in its own right, and has been edited and reviewed independently. The sub-parts are integrated in this document, to form the deliverable as a whole.*
		Project Deliverable (single document, no sub-parts).
	X	Sub-part of a Project Deliverable.

Document Identification			
Deliverable ID:	**D2.1-A**	Part title:	Part I – Task 2.1 Report
Release number/date:		Release 1 / 07-04-2011	
Checked and released by:		Sergio Guillén/ITACA	

Key Information from "Description of Work" (from the Contract)	
Deliverable Description	*Software that enable easy setup and deployment of the universAAL execution environment. Provides a uniform way for AAL services to access the hardware layer*
Dissemination Level	PU=Public
Deliverable Type	P = Prototype
Original due date (month number/date)	Month 11 / 31-Dec-2010 (changed in the first project review)

Authorship& Reviewer Information	
Editor (person/ partner):	Saied Tazari / Fh-IGD
Partners contributing	CNR-ISTI, ENT, Fh-IGD, FZI, IBM, ProSyst, TSB, TUW / USIEG
Reviewed by (person/ partner)	Salvatore Flavio Pileggi/ITACA

Release History

Release number	Date issued	Milestone*	eRoom version	Release description /changes made
1	13.01.2011	PCOS approved	V1	Initial version containing the structure an outline of content
1	22.02.2011	Intermediate approved		Intermediate approved for D2.1 on Software and wiki pages
1	23.03.2011		V2	Finished until end of section 2
1	24.03.2011		V3	Added chapter 3
1	26.03.2011		V4	Improvements + Section 4.1 added
1	26.03.2011		V5	Section 4.3 added
1	27.03.2011	External Proposed	V6	4.2 completed
1	31.03.2011	External revised	V7	Minor improvements
1	05.04.2011	External approved	V8	External approved
1	05.04.2011	Released	V9	Technical manager release

* The project uses a multi-stage internal review and release process, with defined milestones. Milestone names include abbreviations/terms as follows:

- PCOS = ”Planned Content and Structure” (describes planned contents of different sections)
- Intermediate: Document is approximately 50% complete – review checkpoint
- External For release to commission and reviewers;
- proposed: Document authors submit for internal review
- revised: Document authors produce new version in response to internal reviewer comments
- approved: Internal project reviewers accept the document
- released: Project Technical Manager/Coordinator release to Commission Services

Table of Contents

Table of Figures

List of Tables

Executive summary

This is a subpart of the deliverable D2.1-A that reports about the work done to deliver the actual prototype. Hence, it provides information that should help to justify the effort reported in the context of Task 2.1.

Apart from the standard Section 1, which defines the scope and the role of this part in the context of the whole D2.1, this report consists of the following important information:

- Since the whole development work in WP2 is divided among seven expert groups, the scope of D2.1 is defined to be equivalent to the scope of the following expert groups: Middleware, Local Device Discovery and Integration (LDDI), and Security. Therefore, the requirements to be covered by D2.1 are equivalent to the requirements covered by those expert groups.
- The consolidation method used for the reuse of software inherited from the input projects towards the alpha releases of the first set of software artefacts in D2.1 has been based on the general consolidation method described in Part I of D1.3: gathering info (here, about reusable software modules) in a harmonized way (here, in a certain table with predefined columns), while categorizing them (here, assigning modules to one of the seven expert groups); each expert group could then extract the related design decisions of each of the input projects based on the related bunch of software modules and compare them with its own design decisions so that, at the end, the software modules of the one input project with the nearest design decisions could be selected as the initial software to be further developed based on the group's own design decisions and the experience from all of the input projects.
- The last section summarizes the result of the above consolidation process for the three expert groups in the scope of D2.1 by providing info about the imported software artefacts, the first development tasks done during this importing phase, the corresponding development agenda in universAAL, and the time and resource planning for the implementation of the agenda.

Important Note: In this document, there are many references to stuff accessible under the URL http://depot.universaal.org/. In order to access them, you might need to create an own account by clicking a provided link and filling in the form that will open. The rules are: to access compiled code in the maven repository (mostly JARs), there is no need to have any account; for accessing the wiki pages of the expert groups, you only need to have an account and read access will be granted to you automatically; to access the SVN repository of an expert group and its associated issue tracking, you must request to be added as a guest using a provided link.

1 About this Document

1.1 Relationship to other sub-parts of this deliverable

The actual content of D2.1 consists of certain software artefacts and a set of related wiki pages that should help to better understand the software. The expected audience of the above is, however, people from R&D who might want to use the universAAL platform for developing AAL applications, or further improve the platform software itself, or even just to understand how the work done in universAAL might be related to their own R&D work. Therefore, we decided to provide this subpart to satisfy the specific needs for the approval of project efforts in the official reviews organized by the European Commission. It reports about the work done to deliver the actual prototype by providing information that should help to justify the effort reported in the context of Task 2.1.

1.2 Structure of this document

The document consists of three further sections:

Section 2 Defines the scope of this deliverable in the context of the runtime support for the development of AAL applications and in comparison to D2.2.

Section 3 Explains the consolidation method used in WP2 for the reuse of software inherited from the input projects.

Section 4 Provides a complete list of software artefacts in the scope of this deliverable by providing an ID card, reporting the status, and describing the future plans.

1.3 Relationship to other versions of this Part

This is the initial version. Three other versions will follow in project months 18, 27 and 36. We expect that most of the changes in the future versions will be in the last Section of this document, where the status of the developments to date and the plans until M36 are reported.

2 Development goals

2.1 Scope of the developments

2.1.1 The scope of the whole WP2

As known from D1.3-B, AAL Spaces represent smart environments that provide access to AAL services. Using this abstraction, universAAL has agreed that the scope of the AAL platform **for WP2** can be summarized as shown in Figure 1:

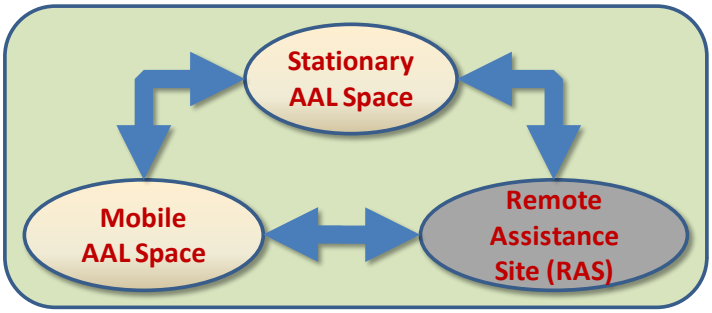


Figure 1: The scope of the AAL platform for WP2

- We differentiate between two types of AAL Spaces: stationary (e.g., a smart home) and mobile (e.g., a car).
- As known, an AAL Space has been abstracted as a dynamic ensemble of networked nodes. This means that we can abstract from the actual number of nodes. Hence, the case of a mobile user, no matter how many networking-enabled nodes he/she is carrying or wearing, can also be categorized as a special case of a mobile AAL Space, namely a near-body AAL Space.
- WP2 must certainly provide runtime support for AAL Spaces, no matter if stationary or mobile. However, apart from the AAL Spaces, there are also organizations whose business is to provide remote assistance to AAL Spaces and their owners/inhabitants. Since the provision of remote assistance is a kind of AAL Service (see D1.3-B, Part II), its internal logic is not interesting for an AAL platform (symbolized with the gray background colour) but such a platform must facilitate the availability and accessibility of them to the end users. Hence, the runtime support by WP2 must also include the interactions between AAL Spaces and remote assistance sites (the arrows in Figure 1).

As known from D1.3-B, the process of consolidating the architectures of the input projects into the first version of the universAAL reference architecture for AAL (see Section 4.3.2 in Part I of D1.3-B) has resulted in the creation of seven expert groups that should cover the basic requirements from the universAAL runtime support for the development of AAL applications and services:

1. Middleware: deals with the issues of distribution and heterogeneity of nodes and facilitates the integration of software components and the communication between them.
2. Local Device Discovery & Integration (LDDI) – formerly known as "hardware abstraction layer": deals with binding special-purpose nodes that are not AAL-aware but can be used in the realization of AAL use cases.
3. Service Infrastructure Expert Group (SIEG): deals with abstracting shareable functionality as service and providing for service brokerage, chaining, composition, and orchestration.
4. Context: deals with all aspects of representing and sharing context and supporting context-awareness and personalization.

5. User Interaction (UI): deals with explicit interaction between human users and AAL spaces.
6. Remote Interoperability (RI): deals with the communication between AAL spaces and the world outside them.
7. Security: deals with the topics, such as trust, access control and privacy-awareness.

Mapping these seven groups to the scene illustrated by Figure 1, we can conclude about the scope of the developments in WP2 the following way:

- The Middleware, LDDI, SIEG, Context, and UI expert groups focus mainly on interoperability within an AAL Space, no matter if mobile or stationary.
- The RI group deals with the arrows in Figure 1.
- The security group considers the whole scene without RAS internals.
- The internals of RAS is business-specific and not WP2 relevant.

2.1.2 The scope of Task 2.1

Having limited the scope of the developments in WP2 to the sum of the scopes of the seven expert groups, it should be sufficient to map those groups to the tasks within WP2 for defining the scope of each of those tasks. In other words, we can define the scope of Task 2.1 by determining the concrete expert groups that work in the context of Task 2.1.

With the argumentation in the Section 1 of the Intro part of this deliverable that defines the content of D1.2 by considering both the original definitions in the DoW and the current terminology and status of work in the project, it should be clear that the Middleware, LDDI, and Security expert groups fall into the scope of Task 2.1 (cf. Figure 2). Hence, the scope of Task 2.1 can be derived by concatenating the description of the scopes of those expert groups:

- Middleware: http://depot.universaal.org/wiki/middleware:overview
- LDDI: http://depot.universaal.org/wiki/lddi:overview
- Security: http://depot.universaal.org/wiki/security:overview

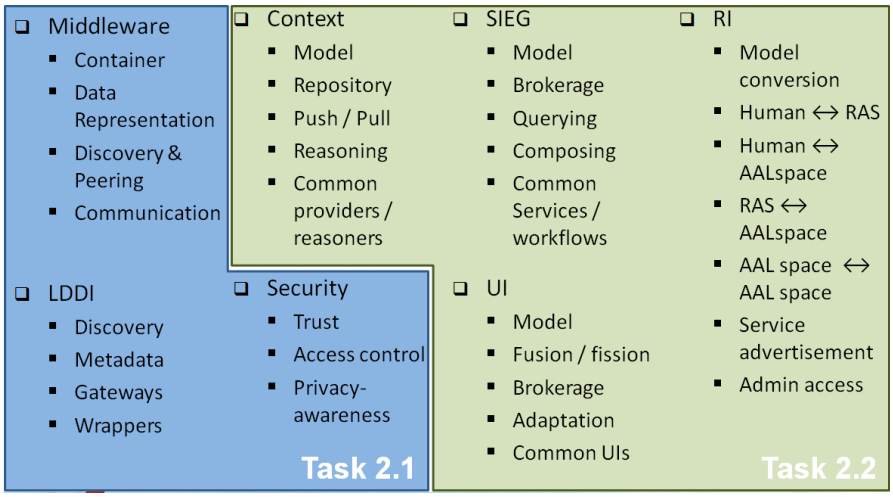


Figure 2: Scope of the development tasks in WP2

2.2 Related requirements

As the expert groups have also gathered the requirements relevant for them, the requirements to be covered in the scope of Task 2.1 can be derived by concatenating the requirements listed by the related expert groups[1]:

- Middleware
  - Container requirements: <wiki>/middleware:Container
  - Discovery & Peering requirements: <wiki>/middleware:Discovery_%26_Peering
  - Communication requirements: <wiki>/middleware:Communication
  - Data Representation requirements: <wiki>/middleware: Data_Representation_Model
- LDDI requirements: <wiki>/lddi:overview
- Security requirements: <wiki>/security:Security_and_threats_analysis_-_Generic_perspective

[1] In the following <wiki> is an abbreviation for http://depot.universaal.org/wiki.

3 Approach

Like all the other tasks in universAAL, also the platform development tasks based their work on a consolidated reuse of results from the input projects. The idea was to create an initial software repository that is mostly formed from selected software modules from the input projects and then define a development plan building on top of that repository. Referring to Table 1 from the Intro to this deliverable, the initial software repository should serve as the alpha release of the first set of software modules in universAAL, which, according to that table, was defined as the subject matter for the first version of both D2.1 and D2.2.

The starting actions for the consolidated reuse of software inherited from the input projects has been already described in D1.3-B in the following sections:

- The general consolidation approach in Section 2.2 of Part I
- The specific method for identifying common architectural building blocks based on consolidating the concrete software modules developed within the input projects described in Section 4.3.2 of Part I
- The concrete data gathered with regard to reusable software modules summarized in Section 3.2 and Appendix B of Part V

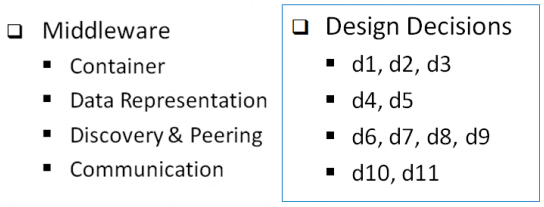
To summarize, the actions has been based on the general consolidation method, namely gathering info (here, about reusable software modules) in a harmonized way (here, in a certain table with predefined columns), while categorizing them (here, associating the modules with the universAAL requirements, architectural layers, and with the seven expert groups). That is, at the end of the above actions, each of the expert groups knew which software modules from each of the input projects were candidates for reuse by that group. A black-box understanding of these modules was also established.

The plan was that the expert groups proceed according to the theory defined in D2.3-A, namely

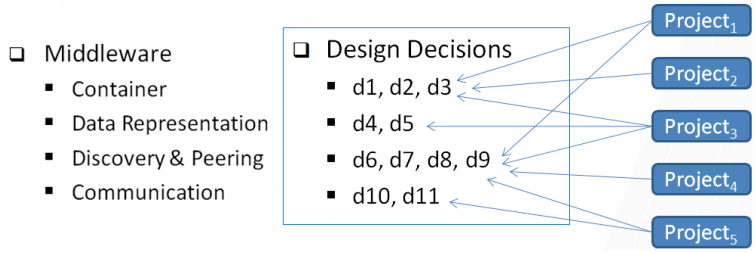
- For each software module from each input project, use the mappings to the universAAL requirements and find out which software modules from the other projects implement similar functionality. This way, identify groups of similar components from the different projects.
- For each of the identified groups of similar components, define criteria for evaluating the alternatives, evaluate them, and rank them based on the results of the evaluation.
- From each of the groups of similar components, choose the one that ranked the best among the alternatives.

In reality, however, the expert groups had many difficulties in the execution of the above procedure, especially because of overlaps and the different modularizations done by the different input projects that made the formation of groups of similar components almost impossible. Apart from that, a serious issue posed by the expert groups was that choosing the best ranked components from each of the assumed groups of similar components in each expert group might lead to the selection of a bunch of software modules from different projects that cause too much overhead of integration work that might prevent us from targeted work on covering the higher priority requirements. For this reason, the meeting of the expert group leaders decided to change the strategy and instead proceed according to the following procedure:

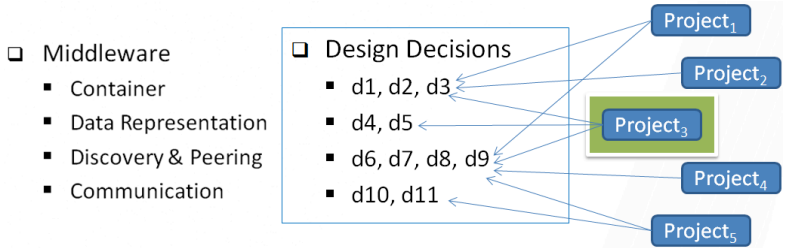
- For each of the topics covered by the expert group, make the group's own design decisions based on a comparison of the related design decisions from the input projects as well as new argumentations and ideas resulted from the group internal discussions.



- Map the design decisions by the input projects to the group's own decisions in order to find out which input projects support which of the group-specific design decisions.



- Compare the mappings to find out which input project is supporting the majority of the group-specific design decisions. All of the software modules of this project which are in the scope of the group will be selected and imported into the universAAL software repository as the alpha release of the initial set of software artefacts of the group.



- Before importing the software modules perform a minimal set of tasks determined by the group, especially in order to match agreed regulations, such as those defined by D2.3-A or usage of appropriate package names (e.g., all starting with `org.universAAL`).
- Define the further development agenda for each of the imported software modules (or new ones that have to be developed from scratch) based on (1) those group-specific design decisions that are not covered fully by the original project, (2) comparison with the related requirements, (3) any improvement ideas defined by the original project itself, and (4) experiences from the other input projects, e.g., considering any novelty of their solutions.
- Document all of the above in open wiki pages so that everybody can follow the process and understand how a group has decided to reuse which software.

This way, each expert group could guarantee that the set of imported software modules do have the needed level of consistency and coherence so that the adaptation of the different software modules to each other has to be done only for the interfaces between the expert groups, if different expert groups choose to reuse software from different input projects.

4 Status Report

In the following, the status of work done according to the above procedure is reported for each of the expert groups working within Task 2.1.

4.1 Middleware

The ID Card of the Middleware Expert Group

GForge Project Home: http://depot.universaal.org/projects/middleware/

Maven Repository: http://depot.universaal.org/maven-repo/org/universAAL/middleware

SVN Root Repository: http://depot.universaal.org/svn/middleware/

Wiki Home: http://depot.universaal.org/wiki/middleware:overview

Tracker Home: http://depot.universaal.org/projects/middleware/tracker/

Forums: http://depot.universaal.org/projects/middleware/forum/

The middleware group has finished the above procedure successfully; the quality of the wiki pages, however, needs to be further improved.

This group has decided to use the PERSONA middleware software as the universAAL alpha release of the middleware. The main reason for this decision is its total conformance with the current version of the universAAL Reference Architecture as documented in D1.3-B. The PERSONA middleware provides a unified node-level API that hides the distribution and heterogeneity of AAL-aware nodes and facilitates the integration of and the communication among the software components running on these nodes, as defined by the universAAL Reference Model and required by the universAAL Reference Architecture. Therefore, the specification of the PERSONA middleware under http://depot.universaal.org/download/docmanfileversion/1/255/PERSONA_D3-1-3_RefArch.pdf (mirrored from http://www.aal-persona.org/deliverables/D3.1.3-Final_Reference_Architecture.pdf) is an important reference for understanding the further work within universAAL. An overview of this concrete architecture can be found in Section 3.1.5 of Part V of D1.3-B. In the following, we assume that the reader of this report has at least read this latter overview, although it is recommended to read the whole specification of the PERSONA middleware, anyhow.

The middleware expert group, however, decided to split up one of the software artefacts of the PERSONA middleware for the sake of better modularization. This made it possible to separate parts of the PERSONA middleware that had overlaps with the scope of the other expert groups[2] and let those expert groups decide about their own solution for message brokering based on the provisions by the middleware group. This was an important preparation work before importing the software that had many implications on the configuration of other imported software artefacts.

[2] More concretely, each of the concrete virtual communication buses of PERSONA (the context, input / output, and service buses) is realized in universAAL by a separate software artifact so that the related expert groups (Context, UI, and SIEG) can themselves decide about the protocols on the respective buses and their brokerage strategies.

In the following, we provide more details by listing the software artefacts in the scope of this group. The artefacts are grouped by the abstract building blocks of the middleware as specified in D1.3-B. However, there is a specific development task across all building blocks: different partners in Task 2.1 have decided to work together to port the whole middleware software to Android. IBM will be leading this task.

4.1.1 The Container Building Block

Wiki home: http://depot.universaal.org/wiki/middleware:Container

According to D2.3-A, the reference implementation of the universAAL platform has been decided to be based on OSGi. On the other hand, the container functionality in PERSONA was delegated to the OSGi framework. Consequently, here we did not need to import any specific software artefact. That is, OSGi, as the default runtime environment in universAAL, is already providing most of the functionality expected from the Container building block of the middleware. For more info on the analysis of the Container building block, please refer to the wiki home.

Nevertheless, the expert group has identified a set of development tasks in order to cover all of the requirements. The strategy how to incorporate these tasks with the basic functionality provided by OSGi has not been determined yet. The tasks and the associated realization plan are summarized in Table 1.

Table 1: Development tasks in Container building block of the middleware

Task	Description	Design contrib. until M18	Development plan per partner	
			CNR-ISTI	USIEG
Fault containment for deployed bundles	A main goal is to isolate bundles that use too much CPU time or memory in order to increase the system reliability	ProSyst		M27 (3.5 PMs)
OSGi reliability improvements	Improve handling of network, device, and application failures	ProSyst		M33 (3 PMs)
Diagnostics	Specify info needed for diagnosing the origin of problems and the mechanism to share this info as context, define appropriate diagnostic rules using this contextual info, and specify actions how to react in each case	Fh-IGD, ProSyst		M24 (7.5 PMs)
Support for multi-part applications	An AAL Application might have different parts that should be deployed on different nodes; the container interface must be enhanced to allow	Fh-IGD, ProSyst	M24 (3 PMs)	

4.1.2 The Data Representation Building Block

Wiki home: http://depot.universaal.org/wiki/middleware:Data_Representation_Model

The decisions with regard to this building block were influenced by the decisions within the other expert groups (Context, SIEG, UI) that own the concrete communication buses (the context, service,

and input/output buses) on top of the basic bus model of the middleware. All the other three groups reached internally the consensus on using an ontological approach for representing knowledge and exchanging messages in order to be able to provide intelligence matchmaking mechanisms based on ontological inference. This was very well compliant with the idea of having a shared solution for data representation for which this building block is responsible. For this reason, the implementation of RDF and OWL in PERSONA and its solution for hidden handling of data serialization and de-serialization (two software artefacts in total) matched very well to the decisions made by the other three expert groups. Here, we introduce these two artefacts with their status and development plans in universAAL:

1. mvn:org.universAAL.middleware/mw.data.representation

Artefact ID Card

Group-ID: org.universAAL.middleware, Artefact-ID: mw.data.representation

SVN Repository: <middleware-svn>/trunk/data.representation

Wiki Page: <wiki>/middleware:data.representation

Maven Repository: <middleware-maven-repo>/mw.data.representation

Description: provides an implementation of RDF and OWL for data representation and ontological reasoning.

Status: Extracted from splitting a PERSONA artefact and applied the group naming conventions. The implementation is not very well compliant with the original specifications. Class hierarchies are limited to the Java inheritance mechanisms. More OWL features are needed in order to be able to perform ontology mapping.

Table 2: Tasks for further development of mw.data.representation

Task	Description	Development plan per partner
		Fh-IGD
Improve conformance	Addresses the two first issues from the status, namely compliance with the original specs and allowing multiple inheritance	M18 (1 PM)
Extend features	Add support for more OWL features, especially for ontology mapping.	M18 (1.5 PMs)
Support for an ontology repository	Provide basic mechanism for a better management of the dependencies of AAL application on used ontologies	M21 (2 PMs)

2. mvn:org.universAAL.middleware/mw.data.serialization

Artefact ID Card

Group-ID: org.universAAL.middleware, Artefact-ID: mw.data.serialization

SVN Repository: <middleware-svn>/trunk/data.serialization

Wiki Page: <wiki>/middleware:data.serialization

Maven Repository: <middleware-maven-repo>/mw.data.serialization

Description: The module for automatic message content (de-)serialization in RDF/Turtle when AAL-aware nodes exchange messages.

Status: Adapted version of a PERSONA artefact. So far no issues are known.

Development plan: There is no need for a development plan. Fh-IGD reserves about 0.5 PM for possible support activities.

4.1.3 The Discovery and Peering Building Block

Wiki home: http://depot.universaal.org/wiki/middleware: middleware:Discovery_%26_Peering

According to D1.3-B, this building block is responsible for seamless connectivity in AAL spaces; that is, its implementation should help AAL-aware nodes to find each other and acknowledge each other as peers that are allowed to exchange messages, thereby handling the low-levels of message exchange. A message at this level is as simple as a byte array or a string. The implementation in PERSONA called Abstract Connection Layer (ACL) consists of five artefacts: one defining the interfaces, three alternative implementations of these interfaces (using R-OSGi, UPnP, and Bluetooth), and one bridging mechanism across all such parallel realizations of the interfaces. The Bluetooth alternative implementation of the interfaces had proven to be very instable and must be re-implemented from scratch if needed.

The further development of the artefacts will be based on a shared development plan because all of the items in the plan will affect all of the four adopted artefacts. X summarized this shared development plan; consequently, no individual plan is provided for the artefacts realizing this building block.

Table 3: Development tasks in the Discovery & Peering building block of the middleware

Task	Description	Design contrib. until M18	Development plan per partner	
			CNR-ISTI	USIEG
Prop-based discovery	Switch from high-level discovery realizations (such as R-OSGi, UPnP, & Bluetooth) with too much overhead for unused features to their underlying discovery protocols (SLP, SSDP, & SDP), but allow the discovery itself to be restricted to nodes with certain properties.	Fh-IGD	M21 (3 PMs)	
Handshaking and security	Analyze the handshaking phase in peering and improve the mechanisms; especially move the encryption mechanisms of the PERSONA Sodapop	Fh-IGD	M24 (2	

	layer to the handshaking phase		PMs)	
Novel discovery algorithms	A research aspect for covering more complex discovery cases when devices might belong to different AAL spaces	Fh-IGD	M36 (2 PMs)	
Reliable ACL	Enhance reliability in ACL, e.g. by allowing parallel connections between two peers and handle the different message transfer failures	Fh-IGD		M18 (3.5 PMs)

3. mvn:org.universAAL.middleware/mw.acl.interfaces

Artefact ID Card

Group-ID: org.universAAL.middleware, Artefact-ID:mw.acl.interfaces

SVN Repository: <middleware-svn>/trunk/acl.interfaces

Wiki Page: missing!

Maven Repository: <middleware-maven-repo>/mw.acl.interfaces

Description: defines P2PConnector as the interface of ACL as well as interfaces expected from the users of ACL, namely PeerDiscoveryListener, SodapopPeer.

Status: Adapted version of a PERSONA artefact. Depending on the overall extension plans given in Table 3, these interfaces will change and consequently the alternative implementations in this building block as well as the implementation of the bridging mechanism will change. Also the Communication building block that is a user of ACL must be adapted accordingly.

4. mvn:org.universAAL.middleware/mw.acl.upnp

Artefact ID Card

Group-ID: org.universAAL.middleware, Artefact-ID:mw.acl.upnp

SVN Repository: <middleware-svn>/trunk/acl.upnp

Wiki Page: missing!

Maven Repository: <middleware-maven-repo>/mw.acl.upnp

Description: Uses an implementation of UPnP for OSGi to implement the P2PConnector interface (see mw.acl.interfaces) for discovering other nodes with the same logic and enables message exchange.

Status: Adapted version of a PERSONA artefact. There will be changes depending on the overall extension plans given in Table 3 and changes in the ACL interfaces. Most probably not much of

the existing code can be used due to switching from UPnP to the underlying SSDP. The main reason for importing this artefact is to keep the alpha release of the middleware executable.

5. mvn:org.universAAL.middleware/mw.acl.rosgi

Artefact ID Card

Group-ID: org.universAAL.middleware, Artefact-ID:mw.acl.rosgi

SVN Repository: <middleware-svn>/trunk/acl.rosgi

Wiki Page: missing!

Maven Repository: <middleware-maven-repo>/mw.acl.rosgi

Description: Uses R-OSGi to implement the P2PConnector interface (see mw.acl.interfaces) for discovering other nodes with the same logic and enables message exchange between them.

Status: Adapted version of a PERSONA artefact. There will be changes depending on the overall extension plans given in Table 3 and changes in the ACL interfaces. For the changes, also the comparable implementations provided by the Eclipse and Apache communities will be considered. Probably not much of the existing code can be used due to concentration on the underlying SLP. However, the overhead of R-OSGi on top of SLP is limited to creating proxies of the discovered peers that might be needed anyhow.

6. mvn:org.universAAL.middleware/mw.acl.bridge

Artefact ID Card

Group-ID: org.universAAL.middleware, Artefact-ID:mw.acl.bridge

SVN Repository: <middleware-svn>/trunk/acl.bridge

Wiki Page: missing!

Maven Repository: <middleware-maven-repo>/mw.acl.bridge

Description: Implements the PeerDiscoveryListener and registers to all implementations of P2PConnector in order to bridge between them so that all peers, no matter which ACL implementation they have, can discover each other and communicate. By default, only middleware instances that share the same ACL implementation can discover each other and communicate, but an instance that uses two ACL implementations can make the peers from one technology visible to the others and vice versa if it also runs this bundle.

Status: Adapted version of a PERSONA artefact. There will be changes depending on the overall extension plans given in Table 3 and changes in the ACL interfaces. The scope of changes is

expected to be limited but it needs extensive integration tests because it was never used in the PERSONA pilot sites.

4.1.4 The Communication Building Block

The PERSONA implementation had two artefacts for the realization of this building block: (1) a so-called Sodapop engine that defined the underlying bus model with bus strategy, bus members, and bus messages while representing a whole instance of middleware as a distinct peer in the AAL space; (2) a second artefact provided the an implementation of RDF and OWL (see mw.data.representation artefact above) as well as an implementation of the four concrete buses (context, service, and input / output buses). This second PERSONA artefact has been split up into four universAAL artefacts, namely mw.data.representation, mw.bus.context, mw.bus.service, and mw.bus.io. From a practical point of view, the latter three belong to the realization of the Communication building block of the middleware in universAAL as they are providing the most important part of the concrete node-level API. But, as a matter of fact, they are also determining about the fundamentals of context, service, and UI management so that their further development plan had to be delegated to the respective expert groups. For this reason, there remains only one artefact to report about in this section.

7. mvn:org.universAAL.middleware/mw.bus.model

Artefact ID Card

Group-ID: org.universAAL.middleware, Artefact-ID: mw.bus.model

SVN Repository: <middleware-svn>/trunk/bus.model

Wiki Page: <wiki>/middleware:bus.model

Maven Repository: <middleware-maven-repo>/mw.bus.model

Description: Represents an instance of the middleware as a peer in an ensemble of nodes by implementing the interfaces PeerDiscoveryListener and SodapopPeer (see acl.interfaces); it also introduces the concepts Bus, BusStrategy, BusMember (Publisher, Subscriber, Caller, & Callee), Message, MessageType, and MessageContentSerializer.

Status: Adapted version of a PERSONA artefact. The implementation is not very well compliant with the original specifications. Class hierarchies are limited to the Java inheritance mechanisms. More OWL features are needed in order to be able to perform ontology mapping.

Table 4: Tasks for further development of mw.bus.model

Task	Description	Design contrib. until M18	Development plan per partner	
			Fh-IGD	USIEG
Upgrade to the new ACL	See the status of mw.acl.interfaces	CNR-ISTI	M18 (0.75 PM)	

Bus & messaging cleanup	Clean up certain historical traces of the original spec of Sodapop that were not used in PERSONA and are not planned to be used in universAAL either, but are causing confusion in understanding the design.	CNR-ISTI, ProSyst	M18 (0.5 PMs)	
Redundancy & voting of bus members	To increase reliability, it must be possible to deploy the critical platform and application artefacts on several nodes. This will affect the modelling of bus members but also needs handling of the different redundancy models.	CNR-ISTI	M18 (0.75 PMs)	M33 (4 PMs)
Security-related enhancements	Apart from detachment of the current encryption logic and delegating it to Discovery & Peering, here the modelling of bus members must be enhanced to carry the granted permission.	CNR-ISTI	M21 (0.5 PM)	

4.2 Local Device Discovery and Integration

The ID Card of the LDDI Expert Group

GForge Project Home: http://depot.universaal.org/projects/lddi

Maven Repository: http://depot.universaal.org/maven-repo/org/universAAL/lddi

SVN Root Repository: http://depot.universaal.org/svn/lddi

WikiHome: http://depot.universaal.org/wiki/lddi:Overview

Tracker Home: http http://depot.universaal.org/projects/lddi/tracker/

Forums: http://depot.universaal.org/projects/lddi/forum/

The LDDI group, originally named HAL as acronym for Hardware Abstraction Layer, has finished the consolidation procedure: the scope of the group work and the related challenges and requirements have been specified; the design decisions have been made; inspection and selection of reusable code is also done; finally, the action plan for further developments within universAAL has been worked out; the wiki pages, however, are almost restricted to the description of the scope, requirements, and design decisions (with much analytic explanations) and still have to be completed with regard to relationship to the reference architecture, results of code reuse, and the development plan.

Making the main design decision was pretty fast because all of the input projects were following a similar pattern. The group agreed on the PERSONA abstract representation of this pattern: a layered architecture consisting of the *access*, *abstraction* and *integration* layers for the discovery and integration of sensors and actuators embedded in AAL spaces.

Since there are many different standards, each with many different application profiles, related to the LDDI group, the group decided to limit the universAAL developments in the breadth and target some progress into the depth, instead. To choose the focus, the group agreed on following the Continua Health Alliance recommendations, on one side, and complement it by some typical standards related to the "smart environment" character of AAL spaces, on the other side. As a result, LDDI decided to adopt two main application profiles from three different standards for the integration into the

universAAL platform, namely home automation and e-health profiles from the ZigBee, Bluetooth and KNX standards.

In terms of software reuse, the group decided to consider the latest news from AALOA: two related reusable software packages reflecting the experiences from the MPOWER and PERSONA projects were adopted by the AALOA community for further development and maintenance in a broader community, especially in cooperation with the OSGi community with possibility that the software is later adopted by them (see http://aaloa.org/project_list):

1. HOMER – HOMe Event Recognition system (incorporates experience from MPOWER in the implementation of ISO/IEEE 11073)

   HOMER is an open and flexible OSGi-based software platform which aims at the integration of various home automation systems and consequential event and situation recognition for smart home (addressing comfort, energy efficiency, etc.) and Ambient Assisted Living (AAL) applications (addressing safety, autonomy, self-confidence, etc.). It provides a running KNX and ISO/IEEE 11073 e-health profiles integration.

2. ZB4OSGi – ZigBee Base Driver for OSGi (initial development in PERSONA)

   This software project aims at integrating ZigBee with OSGi environment, in particular by providing a ZigBee Base Driver, a set of refinement drivers (i.e. ZigBee Home Profile refinement driver), a set of library for parsing ZigBee packets, and other tools for interacting with ZigBee networks.

Hence, LDDI decided to contribute to the further developments of the above two packages in AALOA and use the results as libraries in universAAL-specific developments on top of them. Accordingly, the current software repository of LDDI does not contain any concrete software artefacts because they are either maintained by AALOA or new artefacts to be concretized and initialized until M18. However, the group has already agreed on the following development agenda summarized in Table 5.

Table 5: Development Agenda of the LDDI expert group

Task	Description	Design Contrib. Until M18	Development plan per partner			
			AIT	CNR	ITACA	TSB
unified uAAL device model	Define the ontologies equivalent to the home automation and e-health profiles (consider the relevance to the corresponding profiles from Bluetooth, KNX, and ZigBee, as well as the IEEE 11073 data definitions) and provide templates for binding such devices regardless the underlying communication technology	CNR	M18 (1 PM)		M18 (0.10 PM)	M18 (0.50 PM)
porting PERSONA ZB devices to uAAL	Integrate existing PERSONA ZigBee devices within uAAL platform				M16 (0.25 PM)	
consolidate ZigBee HW	Evaluate and select the available hardware (e.g. Test and integrate the CC2531 TI			M18 (1.00		M18 (0.33

selection	ZBee dongle)			PM)		PM)
ZigBee commissioning tool	Develop the commissioning tool for final configuration of ZigBee devices deployed in an AAL space so that they work properly and according to the user preferences			M24 (3.00 PM)		M24 (2.00 PM)
ZigBee automatic provisioning	Support for automatic download of software / drivers (from external repositories) for enabling communication with non-standard (custom) devices when they join the network. This should be activated whenever the used profile is not known a priori. It will be based on a description mechanism of Zigbee and will enable the platform to find and download the right software also for unknown devices.	AIT		M24 (3.00 PM)		M24 (2.00 PM)
KNX integration	Adapt the Homer solution to the general 3-layer architecture	CNR	M24 (1.50 PM)			M24 (1.00 PM)
IEEE 11073 integration	Adapt the Homer solution to the general 3-layer architecture		M24 (1.50 PM)			M24 (1.00 PM)
Bluetooth integration	In a first try, check how far the PERSONA ACL implementation on top of Bluetooth can be adapted for LDDI-based integration of Bluetooth devices. If not successful, then define a new action plan.				M24 (0.50 PM)	

4.3 Security

The ID Card of the Security Expert Group

GForge Project Home: http://depot.universaal.org/projects/security/

Maven Repository: http://depot.universaal.org/maven-repo/org/universAAL/security

SVN Root Repository: http://depot.universaal.org/svn/security/

Wiki Home: http://depot.universaal.org/wiki/security:overview

Tracker Home: http://depot.universaal.org/projects/security/tracker/

Forums: http://depot.universaal.org/projects/security/forum/

The security group has finished the consolidation procedure partially: the scope of the group work and the related requirements have been specified; the design decisions regarding trust and access control are almost taken; inspection of reusable code is also mostly done; but, the design decisions regarding support for privacy-awareness are not finalized yet; also, the group is still lacking a clear action plan although some specific tasks could be identified as "security injection" in the middleware code; accordingly, the wiki pages must reflect the latest results as the group makes progress with its design decisions and action plans.

This group has confirmed the adoptability of the PERSONA solution for trusting HW and SW artefacts that join an AAL space but decided to move most of the related code from the Communication building block to the Discovery and Peering building block (see the related security task defined in Section 4.1.3 with regard to the secure handshaking phase in the course of peering). It has also confirmed that user authentication should be done by I/O handlers in the course of monitoring / controlling the I/O channels used for explicit interaction with human users. A scenario has been defined for further analyzing the consequences of this decision in terms of software development.

Furthermore, the group decided to use the Android solution for authorization where

- for each context event and service profile, providers state which permissions are required in order to receive the context event / utilize the service,
- subscribers / requesters declare which permissions they need in order to provide which functionality, and
- for each subscriber / requester, the system stores which of the requested permissions have been granted by the owners of the AAL space (see the related security task in the development plan of the artefact mw.bus.model in Section 4.1.4) and uses this info in the course of matchmaking between events and subscriptions (resp. between service profiles and service requests) → the Context, SIEG, and UI groups must define appropriate development tasks for performing a security-enhanced matchmaking when brokering messages.

With the above decisions and tasks, it should be clear that the trust and access control issues have been handled pretty thoroughly. The expert group must still further analyze how to handle the following challenges with more concrete design decisions and plan the related development work until M18:

- The mechanisms for determining the trustworthiness of the HW and SW artefacts as well as for granting permissions to them necessitate some administrative intervention, hence, the group must clarify which exact tools with which interfaces and workflows are needed.
- The whole solution for supporting privacy-awareness still needs to be specified.