AMBIENT ASSISTED LIVING, AAL

JOINT PROGRAMME

ICT-BASED SOLUTIONS FOR ADVANCEMENT OF OLDER PERSONS' INDEPENDENCE AND PARTICIPATION IN THE "SELF-SERVE SOCIETY"

# D3.1 System Technical Requirements

# Final Version

| | |
|---|---|
| Project acronym: | **ProMe** |
| Project full title: | **ProMe – Professional Intergenerational Cooperation and Mentoring** |
| Contract no.: | **AAL-2013-6-026** |
| Author: | **GLUK, SIVECO, PLUS, INV** |
| Delivery date: | **28.02.2015** |
| Dissemination | **Public** |

## TABLE OF CONTENTS

## LIST OF FIGURES

# 1. EXECUTIVE SUMMARY

## 1.1 Link with the objectives of the project

The purpose of this deliverable is to provide the main requirements and technical specifications associated with the ProME platform implementation. A starting point for the production of these requirements is the end-users' requirements that were produced as part of the earlier deliverable of WP2 (namely deliverable D2.2) and that have been updated as basis for this deliverable. Starting from end-users' requirements, the present deliverable elaborates on the technical requirements to be used in the platform architecture design.

## 1.2 State of the art

The Technical requirements collection started in month 4 of the project, respectively with WP3, taking account the international literature and common practices that have been used in similar platforms. During the previous months the requirements were defined adopting the preferences of the end users that have been identified in user studies. The result of this deliverable is the "translation" of the functional requirements into technical requirements.

## 1.3 Structure

The Deliverable is structured as follows: Section 2, following the introductory section, illustrates the methodology that is followed as part of this deliverable. In particular it is explained how the requirements in Deliverable 2.2 were used to derive and describe technical specifications and design options. In Section 3 the categories of the Technical Requirements that have been collected as well as the template of the requirements collection form are presented. In Section 4 the details of the technical requirements of ProMe platform are presented and finally in Section 5 the conclusions of the document are given.

# 2. FROM FUNCTIONAL REQUIREMENTS TO TECHNICAL SPECIFICATIONS

## 2.1 Summary of main requirements

The first step in the development phase of the ProMe project was a requirements analysis (see D2.2) that focused on three major research questions:

1) How can ICTs support communication, collaboration and knowledge transfer?

2) What are key factors for successful collaborative relationships?

3) What are common pitfalls that hinder successful collaborative relationships (offline and online)?

In order to answer the research questions we applied a multitude of methods, i.e., literature research, expert interviews, workshops with potential users and an online survey. The results of the studies allowed us to extract four major requirements that need to be addressed within the development process, which will be briefly described in the following paragraph. We only highlight most important results (a full description of all requirements can be found in D2.2)

1) Provide meaningful profiles
   Profiles are a core element on the platform as information that is provided there is essential to support the matching process. This encompasses, for example, information about the expertise or interests to a user. Based on the results we identified the need for a “personal profile” and a “public profile”, because our participants indicated that they won’t be willing to reveal any kind of information and consider, for example a personal photo or the home address as sensible information. Whereas, the personal profile contains all kinds of information, the public profile provides restricted information to users, who are not registered yet.
2) Provide features that support the collaboration process
   With respect to the process of working together (i.e., sharing professional knowledge) we identified the need to make activities that take place and the progress visible for all involved parties (e.g., mentor and mentee). Therefore, a timeline or a calendar could be useful tools that might support the collaborative process. Additionally, we also suggest triggering face-to-face meeting beside the virtual meetings in order to support the process of “getting to know each other”.
3) Support setting up the organizational structure
   The organizational structure encompasses activities that need to take place in order to define the collaboration process (e.g., agreement between provider of support and receiver). In the beginning it encompasses also the process of finding/defining one’s own role as a provider or receiver. We suggest providing information for provider (responsibilities when taking over a certain role) as well as receiver (what they can expect).
4) Ensure perceived security and trust
   Security and trust were two important topics that were raised within the workshops with potential users. Security issues address for example how data is handled on the platform (e.g., who has access?

What kind of data is stored and how long?). Trust is a broad topic and affects the relationship between the collaboration partners. Social presence, the reduction of virtual distance can positively influences the development of close relationships. Thus, one possibility to address this issue is to foster social presence via the platform (e.g., by video-conferencing). Moreover, a kind of reputation/recommendation component could enhance trust/security assurance on the ProMe platform.

## 2.2 Methodology

This deliverable is part of WP3 of the project dealing with technical requirements. Figure 1 demonstrates the role of the present deliverable within WP3. Furthermore, it illustrates the overall methodology adopted in relation to D3.1. As shown in the figure the main input to this deliverable are the results of D2.2, which have been produced on the basis of the ProMe user workshops and the requirements stemming from the state-of-the-art and the opinions of the various users. D2.2 has provided a set of consolidated functional requirements, which will be enriched in the present document on the basis of technical requirements.

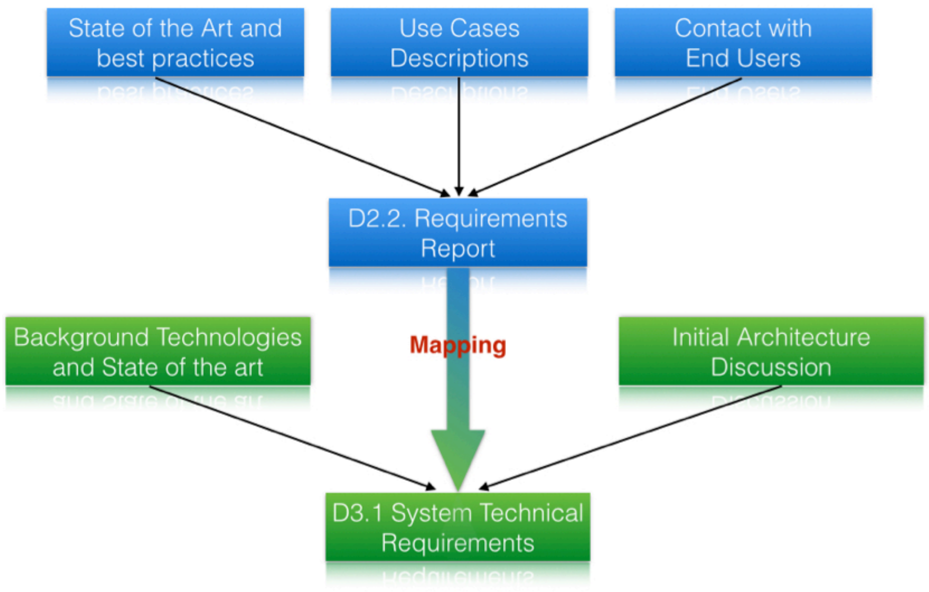
State of the Art and best practices
Use Cases Descriptions
Contact with End Users
D2.2. Requirements Report
Background Technologies and State of the art
Mapping
Initial Architecture Discussion
D3.1 System Technical Requirements

**Figure 1: Mapping of functional requirements to technical Specifications**

Deliverable 3.1 will serve as the main input for producing Deliverable 3.2. The later will illustrate the second iteration of the ProMe architecture as a more detailed description of the target applications and how they could be supported by the specified architecture.

# 3. PROME TECHNICAL REQUIREMENTS COLLECTION APPROACH

The requirements in ProMe were collected in a common format and are available in different representations with different levels of detail. This chapter introduces the format that is used to collect and maintain the requirements, the division in various requirement categories and the ways they are presented to the interested reader. The presentation is made in such a way to be clear and understandable also from a non-technical expert reader.

## 3.1 Categories of technical requirements

In the beginning the category definition was mainly focused on the technical point of view from other online community platforms. As the discussions continued and the users interactions started it was leading to a more elaborated view of the AAL landscape and its requirements. The requirements categories are influenced by the work done in WP1 (Ethical guidelines) and WP2, especially the functional requirements that have been set from the end users. This resulted in the following categories:

C1: Context Information and Context Management

C2: Transparency, Privacy and Security

C3: Support for Designing Human-Computer Interaction

C4: Service Orientation

C5: Configuration, Personalization, Maintainability and Scalability

C6: Execution Environment and Component and Application Container for Heterogenic Devices and Operating Systems

C7: Reliability and Safety

## 3.2 Requirements collection format

In order to collect the requirements in a more solid and common format the template we proposed is the following:

| | |
|---|---|
| **ID:** | **Name:** |
| **Description:** | |
| **Rationale:** | |
| **Type:** | |
| **Work package(s):** | |
| **Customers:** | |
| **Related Requirements:** | |
| **Comments:** | |
| **Responsible:** | |
| **Revision:** | |

Where:

ID: the unique identifier of the requirement

Name: short description of the requirement

Description: the complete requirement description

Rationale: clarify why this is a necessary requirement

Type: one of "functional" or "non-functional"

Work package(s): indicates the WPs that may be influenced by this requirement

Customer: the stakeholders, who benefit if this requirement is considered

Related Requirements: references to requirements that are closely related to this requirement

Comments: remarks and comments that do not fit in any other category

Responsible group: partners or groups responsible for this requirement

Revision: date of creation and date of modification

Please note that for the sake of brevity this deliverable document will not present the requirements in this collection format.

# 4. TECHNICAL REQUIREMENTS

Each of the following subsections addresses one of the requirement categories and lists all requirements collected within that category. The requirements are presented with an easy-to understand definition and a brief rationale.

## 4.1 Context Information and Context Management

Software components that can adapt to the current needs, preferences and the situation of the user are easier to use and better accepted. Information that is used to dynamically adapt services to the situation of a user is called context information. Taking into account all the information gathered from the end-users studies in WP2 the context management category relates mainly to the following concerns:

- **Autonomy:** A user needs to make decisions about his personal status and context without the influence of an external actor;
- **Privacy (including confidentiality)**: Related to above, a user must be able to hide certain personal context data; that means that the user himself could decide which of his personal data could be visible in other users or even to the platform moderator.
- **Extendibility and reusability:** context management system must have the ability to be extended in the future;
- **Configurability and flexibility:** The purpose of context information is to capture information that might be used by applications in order to achieve flexibility;
- **Accessibility and usability:** One of the main use cases of context information is the provision of dynamic user interfaces that adapt themselves to the current needs of the users;
- **Wide variety of fast changing needs:** Considering its definition, context information captures user needs, especially if they are highly dynamic. Applications can then use this information to adapt themselves to these needs.

Taking account the aforementioned categories we present in the following the **Technical Requirements for Category 1.**

| Technical Requirements Category 1: Context Information and Context Management | | | |
|---|---|---|---|
| No. | Requirements | Rationale | Priority |
| R1.1 | Easy Access to Context Information | The architecture shall define a way for software applications that provide services to the end user to easily access context information so that they can dynamically adapt to situation changes. | High |
| R1.2 | Provision of Useful Context Information | The architecture shall define a way to provide context information that is useful for services to adapt themselves according to the needs, preferences and situation of the user. This includes | High |

| | | | |
|---|---|---|---|
| | | integration of context reasoning algorithms to derive useful and valuable information. | |
| R1.3 | Context Information API | The system shall provide a publish-subscribe and query-response API. Providing both access mechanisms provide maximal means of freedom for application developers in defining their interaction with context information. | High |
| R1.4 | Context History | The system shall provide access to past/historic context information. The matchmaking algorithms need access to historic context information to enhance detection reliability | High |
| R1.5 | Uncertain Context Information | The system shall provide a framework for handling of uncertain and imprecise context information. Information provided by complex algorithms can be uncertain and imprecise but useful. Context consumers should be aware of this uncertainty and imprecision if they choose so. | Medium |
| R1.6 | Context Model | The ProMe platform shall provide a shared, extendable, formal and human-readable context model based on a well-defined context modeling language. The developers should be able to easily incorporate context information into their applications. For this an easy to understand model is key. | High |
| R1.7 | Context Information Domains | The ProMe platform shall include context information capturing user profiles, vital signs, device status information, user location, user id and user activities. | High |
| R1.8 | Conflicting Context Information | The ProMe platform shall provide means for resolving conflicting context information coming from different context sources. Conflicting information can occur and accumulate in situations where more than one context source is detecting context information. The context consumer, however, is most of the time not interested or not able to consolidate this information appropriately. Consequently, the context management framework must step in and provide for a conflict-free representation. | High |

## 4.2 Transparency, Privacy and Security

Accounting for privacy, security and transparency is a key in a domain like online communities where confidential, personal information and services are present. Accordingly, as also defined in D1.2 and D2.2 the privacy and security category captures the following concerns:

- Ethical concerns: Transparency, privacy and security are related to almost entire ethical concerns category, including autonomy, freedom, privacy and confidentiality, transparency and consent.
- Laws and regulations concerns category, including fundamental rights and laws and regulations concerns.
- Safety: The ProMe platform must ensure a safe environment for all ProMe end users.

Taking these issues into account we present below the **Technical Requirements for Category 2.**

| Technical Requirements Category 2: Transparency, Privacy and Security | | | |
|---|---|---|---|
| No. | Requirements | Rationale | Priority |
| R2.1 | Confidentiality | It is the way in which the information disclosed or managed by the system is treated. Each part of the system shall be aware of maintaining confidentiality of identifiable data, including controls on storage, handling, and sharing of data. | High |
| R2.2 | Integrity | Each part of the system shall be able to detect data modifications and prevent unauthorized modifications. This applies specifically to service user data and commands sent to actuators, but could also apply to some extent to private communications between end-users. | High |
| R2.3 | Non-repudiation | It shall be possible to trace back every action on sensitive assets to the person or system component that performed it (e.g. to deal with misuse that could not be prevented with technical security mechanisms). | Medium |
| R2.4 | Privacy | Privacy is the right of individuals to keep information about them from being disclosed. End-users want to share personal information about themselves only with specific parties unauthorized access shall be prevented | High |
| R2.5 | Entity identification and authentication | System components shall identify and authenticate an entity (i.e. human users and other | High |

| | | | |
|---|---|---|---|
| | | system components) that wants to use them. A human user here refers to mentors, mentees, etc. and external contributors. It requires identity management and also key management i.e. secure key generation, update and backup. Also system components need to identify and authenticate each other to create a trustworthy system and reduce malicious attacks. | |
| R2.6 | Entity Authorization | System components shall authorize human users and other system components before allowing them access to resources to prevent unauthorized access to and disclosure or usage of sensitive assets. This refers to easy security policy management i.e. policy generation and update. | High |
| R2.7 | Confidentiality and Integrity | System components shall protect data storage or communication to ensure confidentiality and integrity of this data. This refers both to network security (confidentiality and integrity) and security of profile including users data stored at user side or at the service. | High |
| R2.8 | Auditing | All actions on sensitive assets, including failed access attempts, shall be logged. Auditing will be an important component to assure end-users about privacy and prevent future threats. | High |
| R2.9 | Usable Authentication | Mechanisms for authentication that improve usability like Single-Sign-On or multiple security levels of authentication with different confidence levels (initially basic but when required more reliable authentication) shall be provided. With multiple services, a single-sign-on shall be the preferred way to authenticate. | Medium |
| R2.10 | Sticky policy enforcement | Advanced authorization mechanisms shall be provided in order to cope with the distributed setting of multiple third-party services exchanging personal data (e.g. policies attached to data instead of simple centralized access control). This should enable data to travel to multiple services and still enforcing privacy policies on them as defined by the owner of the data. | Medium |

| | | | |
|---|---|---|---|
| R2.11 | Context based access control | Access control mechanisms shall be context aware to for instance prevent leaking detailed information in situations where less detailed information would have sufficed. In an online community environment context should be an important determinant of how certain privacy policies can be modified to meet the needs of the situation especially in case of emergency. This allows better privacy during the normal setting instead of always having the lower security level required in emergency situations. | Low |
| R2.12 | Consent specification | A usable interface shall be provided to capture the consent of the end-user. User consent is an important component to enable users to be confident about sharing data among services. The consent should be easy to understand but still support finely grained policies that services can enforce. | High |
| R2.13 | Privacy protection | Mechanisms to prevent unauthorized profiling shall be provided to prevent leaking of information about who is using which service or leaking information between services. Some ways to achieve this is by pseudonymity, decentralized subscription, etc. ProMe services will be in multiple domains and the ability to profile users for purposes other than the intended purpose can prevent adoption of the platform. If services can be provided by using pseudonyms or with anonymity, they should be the preferred. | Medium |
| R2.14 | Service availability | Mechanisms to prevent malicious attackers and Denial-of-Service attacks shall be provided. The functionality to be provided would depend on the possible attack (threat) scenarios identified and at architectural layer the attack happens (physical to application layer). | High |
| R2.15 | Authentication and authorization in the multi-user settings | Entities shall be authenticated and authorized in the multi-user settings. Multiple users can share a ProMe space and therefore easy and reliable mechanisms to authenticate the different users and authorize their actions are required. | Medium |

## 4.3 Design for supporting Human-Computer Interaction

User interfaces are greatly responsible for the users' perception of the system and therefore they must be able to adequately fulfill the users' needs and expectations and lead to an enhanced quality of experience. They shall be intuitive, easy to learn and configure, easy to maintain and easy to adapt. Their usability shall certainly be a top priority for designers. Especially within AAL this is a very complex task, since the target audience is often not very familiar with the technology, therefore it makes sense to define support for UI design into the architecture, so that all services can benefit.

The support for designing human-computer interaction category captures the following concerns:

- Ability of easy interaction: If interaction is complicated, a user may be reluctant to use the system;
- Accessibility and usability: Special consideration on accessibility and usability must be taken under account when designing user interfaces for older adults;
- Configurability & flexibility: Interaction between user and computer must be easy to configure and adaptable to various changing parameters in environment;
- Extendibility and reusability: Adding of new UI components at any time without the need to restart the system must be possible;
- Quality related concerns: human-computer interface must ensure both proper quality of service and quality of experience levels;
- Compliance to standards (including interoperability): Using open standards facilitate the openness of the platform and thus must be supported.

Consequently in the following table the technical requirements are specified.

| Technical Requirements Category 3: Support for Designing Human Computer Interaction | | | |
|---|---|---|---|
| No. | Requirements | Rationale | Priority |
| R3.1 | Consistent look and feel | The UI framework shall support a consistent appearance between different interfaces and applications. Consistent colors, shapes and symbols should be used together with natural language and vocabulary familiar to the users. They have to feel comfortable using ceratin applications. By maintaining the same look and feel across the application screens users do not get confused and their interaction with the system becomes much easier and natural. By using different colors and symbols for different types of messages designers can emphasize the semantics of the message even more, but this is up to a designer to decide having in mind end-users' specific needs and possible impairments. It | High |

| | | | |
|---|---|---|---|
| | | is however very important to keep the amount of information on the screen at minimum in order not to cognitively overstrain users. Interaction flows should also be as simple and error prone as possible. | |
| R3.2 | Multi-modal user interaction | The UI framework shall support multi-modal user interaction. Support for different modalities is becoming more and more important in user interaction. With the advancement of technology there are different means of interaction that are being utilized. Thus, the UI framework must provide adequate support for the fusion of the input modalities as well as for fusion of output modalities. | High |
| R3.3 | Input fusion and output fission | The UI framework shall be able to combine the input from multiple devices into one in-put representation (input fusion) and to split one output representation into the output to multiple de-vices (output fission) in a meaningful way. This requirement directly follows from the support of multi-modal interaction (TR2). Input is captured in different modalities and the meaning of an interaction can only be inferred by taking the input of all modalities into consideration. Accordingly, the output can be presented to the user with different modalities. | High |
| R3.4 | Adaptability and adaptivity | The UI framework shall support adaptability (allowing change in order to meet user requirements and preferences prior to the start of the interaction) and adaptivity (allowing the system to alter its interaction during run-time). Adaptation of user interfaces is one of the most important aspects of the user interaction framework that has to be addressed appropriately. Adaptation itself can be done during design time or during run time. Since users' preferences and requirements are rarely static not all parameters can be defined prior to the start of a user's interaction with the system. For this reason it is important to have the possibility to | High |

| | | | |
|---|---|---|---|
| | | dynamically modify user interfaces. UI elements should be handled as independently as possible in order to be more easily rearranged and personalized. Also, the change from one device to another device (e.g. from a big screen to a small screen) could be part of this adaptation. Please note that the term adaptation in this context only addresses adaptation of how information is presented to the user not a user-dependent selection or adaptation of the information content itself. | |
| R3.5 | Independence between application and presentation layer | The UI framework shall enable independence between application and presentation layer. The UI framework shall enable abstract representation of user input and system output, regardless of the modality. The support of adaption mechanisms to provide the output according to different output devices and for different impairments or preferences, introduces the challenge to present one and the same output in various ways. To prevent application engineers from implementing all these output techniques for all applications and services in the system, the underlying UI framework must realize this. Thus, the output is actually only an abstract representation, and a clear distinction between application and presentation layer must be ensured. | High |
| R3.6 | Adequate data presentation | The UI framework shall be able to provide adequate data presentation based upon the information received from the user's application (such as the target language). Personalization of user interfaces to users' specific needs and preferences is very important. Apart from inferring some aspects user interfaces should also be customizable based on explicitly received instructions from end users. | High |
| R3.7 | Short and understandable feedback | User interfaces shall provide short and understandable feedback to the user. Whenever possible, actions performed by the users shall be reversible. For high quality user experience, the | Medium |

| | | | |
|---|---|---|---|
| | | user should always know what the system is currently doing. This can be achieved by providing short and understandable feedback. For example, long processing phases between interaction phases should be accompanied with appropriate status messages like progress bars. | |
| R3.8 | Predictable and easy to learn UIs | User interfaces shall be predictable and easy to learn. Assisted persons might be reluctant to adapt to complex user interfaces therefore interaction techniques should be easy to learn. Predictability also supports user acceptance and can be achieved by providing consistency across various dialogs and services. This way, the user can build on prior knowledge. | High |
| R3.9 | Highly responsive UIs | User interfaces shall be highly responsive, giving the user, especially older adults, the impression that changes are instantaneous. The UI framework and the underlying communication layer shall support this functionality by guaranteeing certain Quality of Service parameters like latency. The application itself should minimize the amount of unnecessary updates and round trips between IO Handler and the application. Use of Web Content Accessibility Guidelines (WCAG) 2.0 would also be a must. | High |
| R3.10 | Pluggable UIs | The UI framework shall support pluggable UIs. As an open system the platform – and, thus, the UI framework – shall support the inclusion of new components at any time without the need to restart the system. The system can then be modified to support new devices or to extend to different users and user groups with different impairments or preferences. | Low |
| R3.11 | UI model based on open standards | Using open standards facilitate the openness of the platform. Well-defined standards ease development by existing tutorials and tools, especially when the developer is already familiar with the technology. Optimally, these standards have already been used and proven valuable in similar application domains. | High |

## 4.4 Service Orientation

Service orientation is a set of design principles that ensure reusability and correlation of services. In the AAL domain these two features are especially important since a wide variety of needs have to be fulfilled. From its technical core, service orientation stands for decoupling of components, easy deployment and integration, and scalable solutions.

Accordingly, this category mostly relates to the following concerns:

- Quality of service: Ensured through service orientation for building scalable solutions;
- Deployment: Tool support for deployment of services is necessary to reduce deployment time and costs;
- Extendibility and reusability: It should be possible to easily extend service capabilities and reuse existing components;
- Configurability & flexibility: Users should have the possibility to access, configure and administer relevant parts or properties of deployed services;
- Compliance to standards (including interoperability).

Based on this we define the following requirements:

| Technical requirements Category 4: Service Orientation | | | |
|---|---|---|---|
| No. | Requirements | Rationale | Priority |
| R4.1 | Enabling service orientation | The reference architecture shall define a framework that enables service orientation. | High |
| R4.2 | Easy development and deployment of services | The reference architecture shall define a framework that supports development, packaging and deployment of services. | High |
| R4.3 | Service Description | The service infrastructure shall support service description based on a shared, formal and explicit domain model. | High |
| R4.4 | Service Composition | The service infrastructure shall support the composition and packaging of services by means of easy to-use service composition tools that rely on a service composition language. | High |
| R4.5 | Service Registry | The platform shall define a service registry where available services are collected and service can be queried and accessed. Managing listeners for registration/cancellation of services must be supported. | High |

## 4.5 Configuration, Personalization, Maintainability and Scalability

Configuration and personalization is a key when one wants to support "Wide variety of fast changing needs". This concern can only be achieved by providing a configurable and personalized solution, where application and service can be implemented independently of configuration information. Maintainability was added to this category because all three aspects deal with aspects after the actual implementation of the application. Scalability is an important aspect of every system and it is especially important in ProMe, which potentially could create a dynamic and changing community on a growing system.

So, this category includes the following aspects:

- Maintainability: It should be possible for users, such as technical support personnel to easily maintain the system after deployment (i.e. to monitor the state of the system, to identify exceptions or faults). Managing the scalability configuration should be easy even in large, distributed and heterogeneous systems. It's important for long-term support of evolving environments.
- Configurability & flexibility: The system must provide services and tools to support the configuration of new software, hardware and service, and configuration language should be used by application developers to clearly specify how and what parts of an application can be configured. The system should propose different scalability configurations, which could be dynamically changed according to changes of systems size and its topology.

The following requirements are coming through the above:

| Technical Requirements Category 5: Configuration, Personalisation, Maintainability and Scalability | | | |
|---|---|---|---|
| No. | Requirements | Rationale | Priority |
| R5.1 | Provision and Collection of User Profile Information | The Reference Architecture shall define what information about the user is needed. Also it has to define how and where to save the profiles, how the privacy will be preserved without bothering users with questions. This information is used for the integration of the different applications and to adjust it to the special needs of the end-user. | High |
| R5.2 | Maintenance of the system | If the ProME is running for a long time, maintenance is needed. A service provider or user of the system should be able to maintain the system easily. | High |
| R5.3 | Installation of new services | The system shall provide services and tools to support the installation of new software or hardware and service into the ProMe platform and to install precompiled software-modules. It should be possible to abstract the applications from the hardware. Moreover, suppliers that | High |

| | | | |
|---|---|---|---|
| | | want to protect their intellectual property may only provide precompiled code. | |
| R5.4 | Configuration of the ProMe services | The system shall provide services and tools to support the configuration of new software, hardware and service. The configuration mechanisms should be easy to manage also for relatives with little or no technical knowledge, this would enhance acceptance of the platform | Medium |
| R5.5 | Personalization of the AAL Service | The system shall provide services and tools to support the personalization of new software, hardware and service. The personalization helps to install the necessary software and to adapt the ProMe space to the special personal needs of the end user. This is in dynamic configuration approach, which can be done during runtime. | Medium |
| R5.6 | Configuration language | A configuration language shall be used to clearly specify how and what parts of an application can be configured. The application developers should be able to easily incorporate context information into their applications. | High |

## 4.6 Execution Environment and Component and Application Container for Heterogenic Devices and Operating Systems

It is intended that the ProMe platform is accessible from various devices, i.e., smartphone, tablet TV and a home PC. “Ability of easy integration” and “extensibility” suggests that still a somewhat common format, infrastructure, etc. is needed for easy development and deployment of applications. From a technical point of view a common component and application container that can be layered on top of different devices can address this and operating systems but still provide a shared infrastructure for integration.

This requirements category captures the following concerns:

- Extendibility and reusability: User should be able to use different devices across applications and services;
- Maintainability: Ease maintenance through identification of faulty subsystems
- Deployment: The container should simply being deployed by being independent from the underlying hardware or operating systems;
- Quality related concerns: The container must provide time constraints for applications and allocate resources differently to different applications, according to their importance.

The following requirements concern this category:

| Technical Requirements Category 6: Execution Environment and Component and Application Container for Heterogenic Devices and Operating Systems | | | |
|---|---|---|---|
| No. | Requirements | Rationale | Priority |
| R6.1 | Real time | Ability of providing time constraints for applications. Some applications can have time constraints. | High |
| R6.2 | Proper execution of components and applications | The reference architecture shall define measures and mechanisms to ensure a proper execution of components and applications. | High |
| R6.3 | Resource management | The container shall allow for scalable allocation of resources, shall guarantee the allocation of resources during execution, and shall ensure that components do not exceed component-specific resource boundaries, if required by an application (also real-time requirements) | High |
| R6.4 | Redundancy of the functionalities | Functional redundancy and functionality overlap shall be supported by the system. Especially in case of emergency situations, the fulfillment of this requirement would provide for reliable and robust service provision. | Medium |
| R6.5 | Log trace for the middleware subsystems | It shall be possible to reload the history state of a subsystem (also remotely). This prevents data loss and enables faster recovery after failures. Fulfillment of this requirement demands for backup procedures. | High |

## 4.7 Reliability and Safety

Reliability and safety are two important aspects of AAL. Thus, the ProMe platform should be able to evaluate, ensure and constrain certain properties of the services (also end user services), so that a certain level of reliability offered to the end user can be ensured by the platform itself. The reliability and safety category captures the following concerns:

- Safety: The platform must protect users against any physical, social, financial, or other type of damage;
- Quality related concerns: Proper levels of service and experience quality must be ensured through various fault-tolerance mechanisms. Furthermore, guarantees of the lower bound on the communication bandwidth, upper bounds on the latency and jitter must be determinable;

- Configurability & flexibility: The system should be capable of adapting to changed reliability of subsystems over lifetime, and provide different levels of reliability of the communication service.

The following requirements derive in this category:

| Technical Requirements Category 7: Reliability and Safety | | | |
|---|---|---|---|
| No. | Requirements | Rationale | Priority |
| R7.1 | Design for Testability | Testability shall be supported by the architecture (design testing, system-integration testing, manufacturing testing and assembly testing). When test requirements are already considered by the design decisions, the effort of testing can be reduced. | High |
| R7.2 | Delay/Disruption-tolerant Networking | Communication services that tolerate delays/disruption shall be provided by the architecture. Delays and disruption cannot be completely eliminated (e.g., wireless networks). | High |
| R7.3 | Communication Resource Guarantees | For messages that are exchanged within a certain subsystem, guarantees of the lower bound on the communication bandwidth, upper bounds on the latency and jitter shall be determinable. Resource utilization of subsystems that share the same communication infrastructure has to be bounded, especially in the worst-case scenario, in order to ensure the correlation of subsystems. | High |
| R7.4 | Fault Hypothesis | Assumptions shall be identified that define the type and frequency of faults that the system has to be able to tolerate. For a fault-tolerant system it is important that a fully specified fault hypothesis exists. If real-world faults are not covered by the fault hypothesis, even a fault-tolerant system may fail. | Medium |
| R7.5 | Error-Containment | The architecture must support the establishment of error containment regions, where errors can be detected with defined error-containment coverage. The definition of appropriate error containment regions and error detection mechanisms help to prevent the propagation of the consequences of an error within one subsystem to other subsystems. | Medium |

| | | | |
|---|---|---|---|
| R7.6 | Tolerance of Software Errors | Protection mechanisms within the architecture shall be able to handle software errors. Often software is provided that is not free of errors. The application provider wants to contact a technician or reboot the system each time an error occurs. | Medium |
| R7.7 | Bounded Start-up and Restart Time | A known, bounded and minimal start-up time of system components has to be assured by the architecture. Predictable start-up/restart time is a key issue for fault tolerance and user satisfaction. | High |
| R7.8 | Fault Classification | Error-detection mechanisms provided by the architecture have to distinguish between transient and permanent faults. Classification of faults is important for fault tolerance and maintenance, as in case of transient faults recovery mechanisms like component restart can be applied, while permanent faults require explicit repair action. | Medium |
| R7.9 | Pre-emptive Resource Allocation | The architecture must ensure that individual subsystems cannot dominate/block shared communication resources. | High |
| R7.10 | Worst Case Execution Time Analysis | The calculation of the worst-case execution time (WCET) of software modules with feasible effort shall be supported by the architecture. Deterministic behavior in each layer of the system is a prerequisite to analyze the WCET of a system. Knowledge about the WCET is a key issue for robustness. | High |
| R7.11 | Systematic Diagnostic Methods | The detection of application-independent failure modes (e.g., communication errors) should be supported by providing systematic diagnostic methods. | High |
| R7.12 | Different Levels of Reliability | The architecture shall provide different levels of reliability of the communication service. Distinct levels of reliability are required by different applications (e.g., text chatting vs. video conference). The higher the level the higher the communication cost. | High |

# 5. CONCLUSION

This deliverable describes the technical requirements of the ProMe platform, while it has also makes a step towards the design of the ProMe architecture. As far as requirement-engineering processes are concerned the present deliverable has elicited new sets of technical requirements stemming from the description of the AAL framework, as well as from the elaboration of non-functional requirements that were discussed in D2.2. As far the design of the system is concerned, the deliverable has synthesized the initial step for the architecture design.

# 6. REFERENCES

Christakis, N. A., & Fowler, J. H. (2010). *Connected: the amazing power of social networks and how they shape our lives*. London: HarperPress.

Fukuyama, F. (1996). *Trust: The social virtues and the creation of prosperity*(Vol. 457). New York: Free press., 1996

Gkorou, D., Vinkó, T., Pouwelse, J., & Epema, D. (2013, September). Leveraging node properties in random walks for robust reputations in decentralized networks. In *Peer-to-Peer Computing (P2P), 2013 IEEE Thirteenth International Conference on* (pp. 1-10). IEEE.

Than, C., & Han, S. (2014). Improving recommender systems by incorporating similarity, trust and reputation. *Journal of Internet Services and Information Security (JISIS)*, *4*(1), 64-76.