FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Standardized Connection from Personal Healthcare Devices to AAL Platforms

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Medizinische Informatik

eingereicht von

Patrick Stern
Matrikelnummer 0625389

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung: Ao.Univ.Prof. Dipl.-Ing. Dr. techn. Wolfgang Zagler
Mitwirkung: Ing. Andreas Hochgatterer
Thomas Fuxreiter

Wien, 15.10.2012

(Unterschrift Verfasser) (Unterschrift Betreuung)

FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Standardized Connection from Personal Healthcare Devices to AAL Platforms

MASTER'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Medical Informatics

by

Patrick Stern
Registration Number 0625389

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Ao.Univ.Prof. Dipl.-Ing. Dr. techn. Wolfgang Zagler
Assistance: Ing. Andreas Hochgatterer
Thomas Fuxreiter

Vienna, 15.10.2012

(Signature of Author) (Signature of Advisor)

Erklärung zur Verfassung der Arbeit

Patrick Stern
Goergengasse 9-11/11/1, 1190 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

(Ort, Datum) (Unterschrift Verfasser)

Danksagung

Diese Abschlussarbeit gilt als Einzelleistung. Ich als Einzelner werde mit dieser Arbeit direkt in Verbindung gebracht. Sie ist Teil meines Studiums, und ich übernehme die Verantwortung dafür. Doch ich habe nicht nur die Verantwortung für den Inhalt. Ich habe auch die Verantwortung all jenen zu danken die mich auf meinem Weg, und nicht zuletzt bei dieser Arbeit, unterstützt haben.

Zuallererst gilt mein aufrichtigster Dank meiner Familie - meinen Geschwistern, meiner Mutter, meinem Vater, und meinen Großeltern. Ohne eure Unterstützung wäre ich nicht der, welcher ich heute bin. Vor allem bedanke ich mich für die Art und Weise in der ich mein Studium leben durfte. Ihr habt mir immer den Freiraum gegeben den ich zum Weiterentwickeln brauchte oder zumindest wollte. Ich weiß, dass die mir gegebenen Möglichkeiten fernab jeder Selbstverständlichkeit sind. Ihr habt immer auf mich vertraut und an mich geglaubt - selbst wenn ich selber zweifelte. Dieser Beitrag war so unerlässlich wie mein Dank groß ist.

Des Weiteren will ich mich auch an all die Anderen richten, welche mich in den vergangenen Monaten oder auch schon langjährig unterstützt haben. Allen voran danke ich meiner Freundin, Elisabeth Bitzinger, welche im letzten halben Jahr so manche Laune ertragen musste und immer eine Stütze für mich war. Nicht vergessen will ich jene, die einen wissenschaftlichen Beitrag zu dieser Abschlussarbeit geliefert haben. Hervorzuheben sind hier meine Betreuer, aber auch manch anderer stand mir mit Rat zu Verfügung.

Mehr als die Anerkennung dieser Hilfe bleibt mir auf dieser Seite nicht. Es sei mein gebührender Respekt hier in Worte gefasst, denn ohne diese Unterstützung wäre für mich vieles nicht möglich gewesen.

Abstract

In the upcoming 40 years the number of above-65-year-old citizens will almost double in Austria. Even though this number is extreme in Central Europe it is the expression of a global trend of an ageing population. Furthermore, this affects the ratio between dependent people and their caretakers. The increase of overall expenditure of time and costs for the social system is the consequence of this development and puts the middle-aged citizens under further growing pressure because they have to carry this burden.

Ambient Assisted Living (AAL) is a field of research tackling this problem and comprising methods, concepts and systems, with the goal to support old and handicapped people in daily life's business. People have the desire to grow old in their usual homes instead of retirement homes. AAL solutions facilitate this by increasing their independence in this living environment. AAL platforms are socio-technical systems that combine technical infrastructure, consisting of hard- and software, and services provided by third parties, such as healthcare professionals or call centres. The range of manufacturers of AAL products is diverse and companies from several branches have entered the market. They use proprietary protocols for communication between Personal Health Devices (PHD) (e.g. oximeter) and their manager software running on Computational Engines (CE). A standardized approach with widespread adoption is lacking. Therefore, achieving interoperability is the goal of research groups and initiatives.

The UniversAAL project is one of these. Its objective is to encourage new developers of AAL services and support them during the design and the implementation phase by providing an infrastructure (e.g. a store where they can offer their products) and a reference implementation of an AAL platform. The goal is to reduce their development costs and time. Indirectly the customers and users of these AAL solutions will benefit due to reduced prices and shorter innovation cycles. Furthermore, the use of common knowledge bases and the availability of a reference architecture and implementation will improve interoperability between products of different manufacturers. The ISO/IEEE 11073 is an existing and approved set of standards for the exchange of vital sign information. In this thesis an interface between Continua-certified PHDs and its device manager is implemented in the UniversAAL reference AAL system by using these standards. The Continua Health Alliance is a non-profit- open industry organisation dedicated to establishing a system of interoperable connected personal health solutions. Bluetooth is its flagship transport protocol and the one we use to connect the PHDs. The manager runs on a CE and we use a desktop computer with Linux as its operating system. The developed interface software is open source and a best practice for future developers.

Kurzfassung

In den kommenden 40 Jahren wird sich die Anzahl der Bürger im Alter über 65 beinahe verdoppeln in Österreich. Auch wenn diese Zahlen im internationalen Vergleich hoch sind, so handelt es sich dennoch um einen globalen Trend einer alternden Bevölkerung. Dadurch verschiebt sich zunehmend das Verhältnis zwischen Hilfsbedürftigen und deren Helfern und Helferinnen. Die Konsequenz ist ein Anstieg der Kosten für das Gesundheits- und Pflegewesen. Zudem steigt auch der zeitliche Aufwand für Pflege kumuliert über die Gesamtbevölkerung. Die Folge ist ein erhöhter Bedarf an Pflegepersonal. Diese Entwicklung erhöht zunehmend den Druck auf die Bevölkerung mittleren Alters, die diese Belastung auffangen muss.

Ambient Assisted Living (AAL) ist ein Forschungsfeld, welches sich die Bearbeitung dieses Problems zur Aufgabe gemacht hat. Es umfasst die Entwicklung von Methoden, Konzepten und Systemen mit dem Ziel, ältere Menschen bei den Tätigkeiten des Alltags zu unterstützen. Der Wunsch, im eigenen Heim zu altern, ist weit verbreitet. AAL-Lösungen wollen diesem Wunsch nachkommen, indem sie die Unabhängigkeit von externer Hilfe stärken. Solche AAL-Plattformen sind eine Kombination aus technischer Infrastruktur, bestehend aus Hard- und Software, und Dienstleistungen von Dritten, wie Pflegepersonal, Callcenter oder Datenbanken. Die Vielfältigkeit an Herstellern von AAL-Produkten ist groß, und die unterschiedlichen Firmen haben Hintergründe in den verschiedensten Branchen. Viele verwenden die ihnen bekannten, proprietären Protokolle, welche zueinander inkompatibel sind. Es gibt keinen standardisierten Ansatz, um medizinische Endgeräte (z.B. Oximeter) mit deren Manager-Software verbindet.

Ein Projekt mit dem Ziel diese Interoperabilität zu verbessern ist UniversAAL. Es sollen neue Entwickler und Entwicklerinnen von AAL-Diensten in der Design- und Implementationsphase mit der nötigen Infrastruktur (z.B. Webshop für AAL-Dienste) und der Referenzimplementation einer AAL-Plattform unterstützt werden. Des weiteren soll die Orientierung an einer gemeinsamen Referenz-Architektur die Kompatibilität der Produkte unterschiedlicher Hersteller verbessern. Die ISO/IEEE 11073 ist eine Reihe von Standards für den Austausch von Vital-Daten. Diese Diplomarbeit beschreibt die Entwicklung einer Standard-konformen Schnittstelle zwischen Continua-zertifizierten medizinischen Endgeräten und deren Manager-Software, welche im Referenz-AAL-System von UniversAAL eingebunden ist. Bluetooth ist das Transport-Protokoll, welches momentan am häufigsten zum Einsatz kommt. Darum orientieren auch wir uns in dieser Arbeit daran. Die Manager-Software läuft auf einem Linux-System als Open Source implementiert. Dadurch wird zukünftigen Entwicklern und Entwicklerinnen eine Basis für eigene AAL-Lösungen geliefert.

Contents

CHAPTER 1

Introduction

From 1910 to the year 2000 in Germany the population of over-64-year-olds rose from three million to 13 million people [16]. This is a secular trend since life expectancy is rising, while birth rates are decreasing [19]. Consequently the population is ageing and this puts the existing social system and care givers under pressure due to the increase of expenditure of time and costs [16].

In the introduction this problem of an ageing society and its consequences will be discussed. Furthermore, we want to introduce a field of study named Ambient Assisted Living (AAL) that is concerned with this development [16]. An AAL system is a socio-technical system with the purpose to support people with diseases and disabilities in their daily routine when performing daily living activities [67]. Due to demographic trends the primary target group are older people. This chapter introduces this branch of research including its legal situation and points out the major issues. We focus especially on problems of interoperability between AAL devices and managers which is the motivation for this thesis. The last part of this chapter describes the goals of the master thesis itself and explains its methodological approach.

1.1 Background and Motivation

The ageing population is a challenge for the subsequent decades [41]. The female life expectancy in the respective record-holding country has risen continuously three months every year over the past 160 years [19]. Due to regional crises the country which holds the record changes, but the global trend is stable and remarkably linear. Consolidated, this ageing of the population cannot be expected to have a limit in sight [19]. In combination with decreasing birth rates the consequence is an increase of senior citizens in absolute and relative numbers [8]. The growing size of the age-group of above-65-year-olds is startling and threatens the Austrian generation contract by an increase of costs for the public pension and healthcare systems [16]. Additionally to these growing expenses for the government as a whole, the change of the relative numbers results in

less economic resources being available [41]. Figure 1.1 illustrates the age structure diagram of 2010, and the mean prospected values for 2030 and 2050. It indicates the current size of the older population (those of the age of 65 and older) in Austria which is approximately 1.5 million in 2010 and is projected to be above 2 million in 2030 and 2.6 million in 2050 [8]. This trend is accompanied by an increase in the overall population from 8.4 million to 9.4 million [8]. Compared to the 70% ascent of the above-65-year-olds this is not proportional.

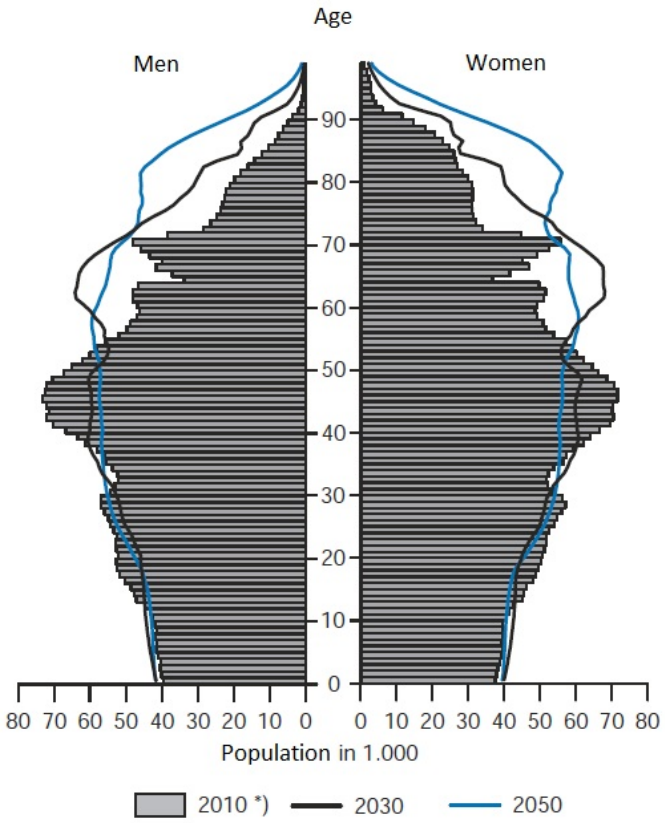


Figure 1.1: Age Structure of the Austrian Population in 2010, 2030* and 2050* (* Mean prospected values.
Source: Statistics Austria - Modified Figure [8]

The changes in total population are important numbers, but now we focus on demography, which primarily is interested in the dependency ratio of the population. Figure 1.1 illustrates the prospected age structure of Austria. We are already far away from the pyramid-look which applied to the commencement of the last century, moreover this development will move on to a mushroom-like shape with an overhead of older people [64].

In Figure 1.2 we see the prospected age structure for Austria until 2050. The old age de-

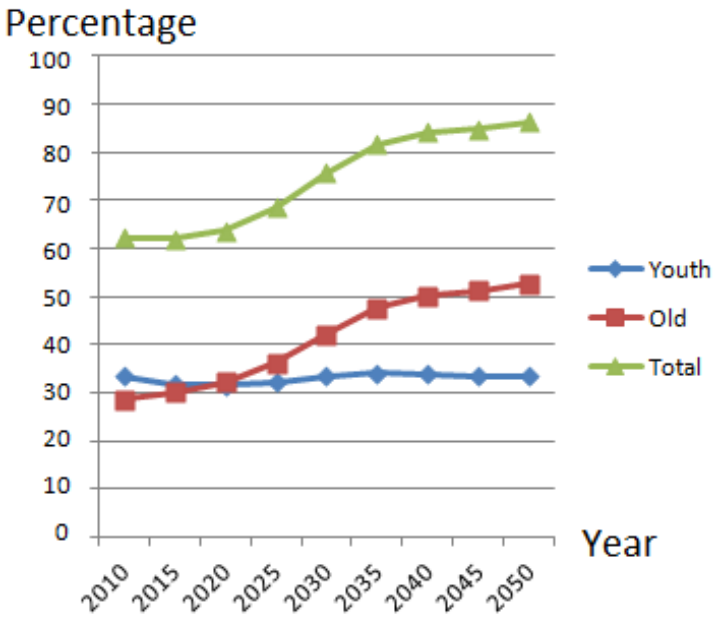


Figure 1.2: Projected Dependency Ratio 2011-2075 (Alternative Definition of broad Age Groups)
Source: Statistics Austria - Self-made Figure [8]

Dependency Ratio The age dependency ratio opposes the number of possible caretakers to the number of individuals who are likely to require care; or in other words: dependent people. Hence it is closely connected to the basic concept of the generation contract as it is based on our legal working period. The underlying theory of dependency takes into account that before the age of 20 we are in school and depend on our parents or legal guardians (youth age dependency). From the age of 65 on we are retired and depend on the middle-aged again as they settle our pension by paying their taxes (old age dependency) [8].

pendency ratio will increase from 17.6% in 2010 up to 28% in the subsequent 40 years (and the Eurostat values are even higher). The young age dependency ratio is stagnating on a low level [8]. The default definition of broad age groups is below 15 years for children, from the age of 15 to 60 for the working population, and old-aged are the above-60-year old. However, we prefer the alternative definition with boundaries at the age of 20 and 65 since this reflects the real working age better. The consequence of the rising dependency ratio affects the required balance of caretakers and nursing cases, which causes an increasing pressure on the middle-aged citizens [8]. This pressure is induced by raising expenses of the generation contract and by the demand for personal care for older people as well. The time effort to take care of the increasing number of old people is distributed on a decreasing number of middle-aged, which is illustrated in Figure 1.1. The burden to be responsible for the well-being of their children and older family members at the same time has to be carried by people in their 40ies and 50ies, for whom the metaphor 'sandwich generation' is introduced [41]. The term 'generation' connotes here the generational position within the family and not a birth cohort. Our widespread division of work causes an especial burden for women, as they are the traditional caregivers in western culture [41]. The growing number of beneficiaries challenges the welfare system since an increasing amount of payees is supported by a decreasing number of depositors. In this case the welfare system has fewer possibilities to support individuals and put this duty onto the shoulders

of the relatives once more [41]. Concluding it can be said that the increasing old age dependency ratio is leading to a growing financial and time-consuming burden for the middle-aged generation [8].

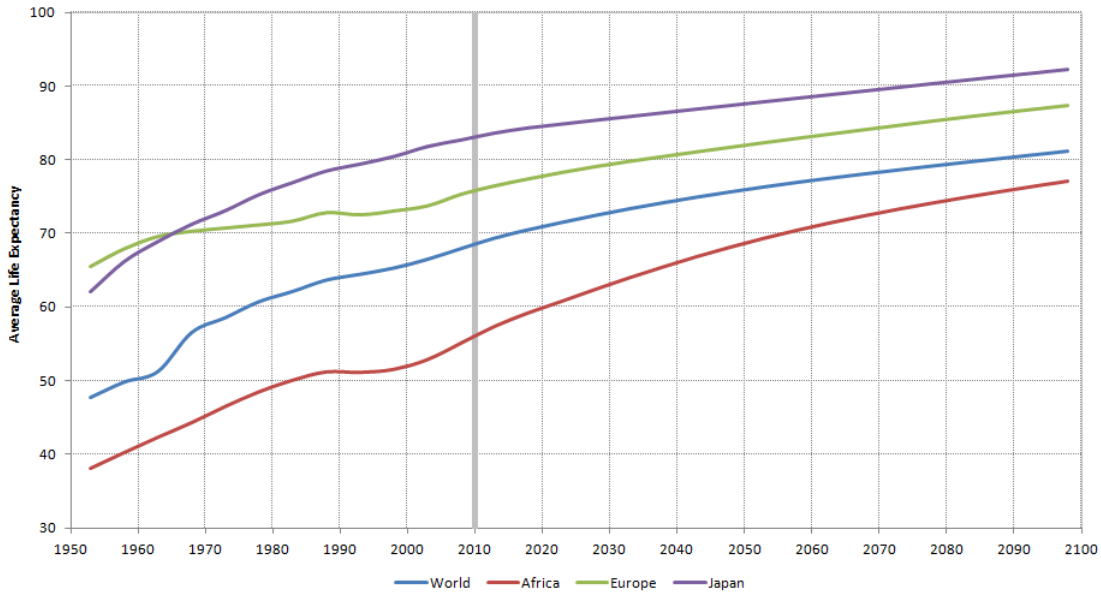


Figure 1.3: Life Expectancy at Birth
Source: United Nations Statistics Division - Original Figure [49]

We were discussing Austrian data so far. However, it is important to point out that the trend of an ageing population is not regional but global [57]. The average age is increasing worldwide even although there are still big differences between the continents. In Europe in the post-war period the average age was 29.2, then 37.7 at the turn of the millennium and will be 47.7 in the year 2050 [49]. This average age is about 10 years higher than world's average. In Figure 1.3 it can be seen that in 2010 the life expectancy in Europe is 20 years higher than it is in Africa [49]. However, Africa made the biggest progress in the past 50 years [49]. The general trend is in evidence for all continents and the rise of average life expectancy all over the world is approximately 20 years since World War II [57] and the number is expected to grow even bigger in the future [49].

Expenses and time effort of healthcare correlate with the fitness of the individuals. The Berlin Age Study from 2009 investigated the degree of multi morbidity in older age groups in Germany, Europe's most populous country, and the result can be seen in Figure 1.4 (the statistic is based on self-assessments). With rising age the number of chronic diseases is increasing and while one third of people in their 40ies have no disease, hardly anyone is untroubled from the above-70-year-olds [57]. Since the mortality is much higher (26% in the upcoming 4 years com-

pared to all other groups [57]) we desire to take a closer look at the older citizens with 5 or more diseases. The degree triplicates (from 4% to 12%) from the first to the second age-group and comprises one quarter (24%) of the oldest ones [57]. People with diseases and disabilities have difficulties performing daily living activities and have a need for support. Therefore, one of the major challenges is to maintain health and function of the body and avoid further diseases [67].

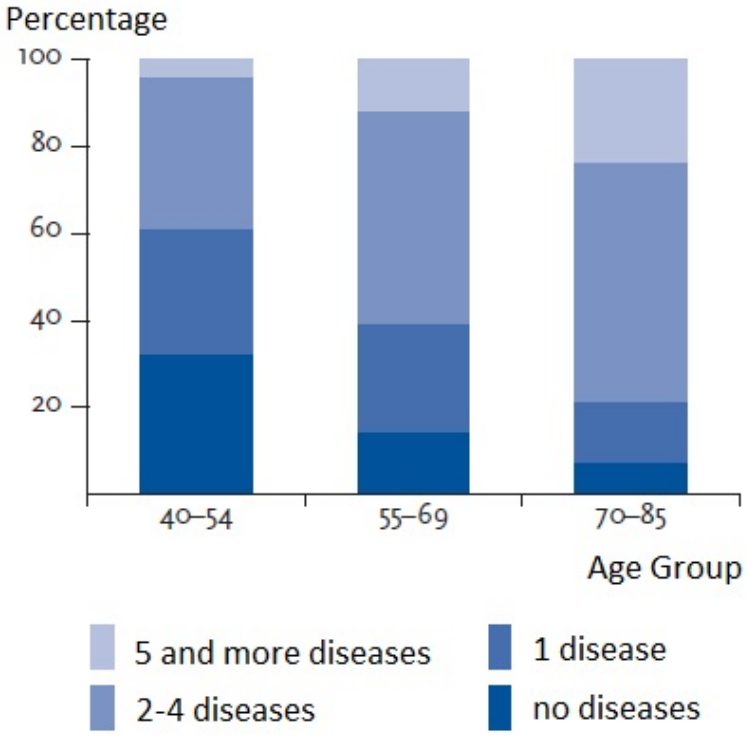


Figure 1.4: Ratio of People with multiple Diseases, arranged in Age Groups
Source: Federal Statistical Office of Germany - Modified Figure [57]

A further important factor is the general desire of the older generation to continue living in their usual living environment as long as possible. A study of the U.S. Census Bureau from 1995 indicates that more than 95% of Americans desire to age in their own homes [67]. The healthcare facilities of 2012 are not capable of the development of our ageing society and its inherent correlation between age and diseases [16]. Considering the progressive imbalance between caretakers and nursing cases [41], involving technology in healthcare is unavoidable and solutions for the increasing demand for the assistance systems such as home automation systems have to be provided [16].

1.2 Ambient Assisted Living

AAL System An AAL system is a socio-technical system that provides AAL services to people with diseases and disabilities to facilitate daily living activities. It aims to support persons in need as well as their helpers [69].

In the previous section we discussed our ageing population and the resulting problems society has to face. In this section the necessities that arise when the population as a whole grows older will be pointed out and we will present a solution for this demand for assistance, namely AAL systems.

The four most distressing doubts that are connected to ageing (in ascending order) are the feeling of being a burden for others, loneliness, diseases and the loss of independence [51]. The desires of older people are to [51]:

- Reside longer and self-determined at home
- Take part in social life for a longer time and age actively
- Increase the quality of life
- Preserve and gain freedom

And this implies the following necessities [51]:

- Support of caregivers
- Reduction of costs for healthcare
- Less institutionalization

Neither older citizens themselves desire to reside in a retirement home, nor do their relatives want them to do so. Only one out of five persons requiring nursing and one out of ten family members thinks that she or he has to go to a home for senior citizens [67]. Like mentioned in the previous section, older people prefer to reside at home in their best-known living environment [51]. All these desires require a healthcare system that has to become more efficient and more flexible and result in an increasing demand for assistance systems [16]. The ageing society confronts our healthcare system with a higher prevalence of age-related diseases, like hypertension or dementia. Furthermore, the diseases restrict independent living and lead to a loss of quality of life [16]. AAL made an issue out of these upcoming problems and positions itself as a practicable solution for supporting older people to overcome these fears in the future.

Concept

Smart Environment *„A smart environment is a physical world that is richly and invisibly interwoven with sensors, actuators and computational elements, embedded seamlessly in the everyday objects of our lives, and connected through a continuous network*“- Mark Weiser.

Ubiquitous computing Ubiquitous computing is a post-desktop model of human-computer interaction in which information processing and computational capabilities of networks have been thoroughly integrated into everyday objects and activities [73].

The rising number of smartphones contrasted with the shrinking market for conventional cell phones [1] is one example for the triumph of ubiquitous computing. Smart environments make use of this development and are small worlds consisting of a network of smart devices that cooperate seamlessly with one another having the underlying goal to support occupants in their activities. Ambient technologies refer to electronic environments that are sensitive and responsive to the presence of people and are used to build such smart environments [69]. A smart environment acts context-aware and personalized in order to provide an adaptive response that is reactive or pro-active [69].

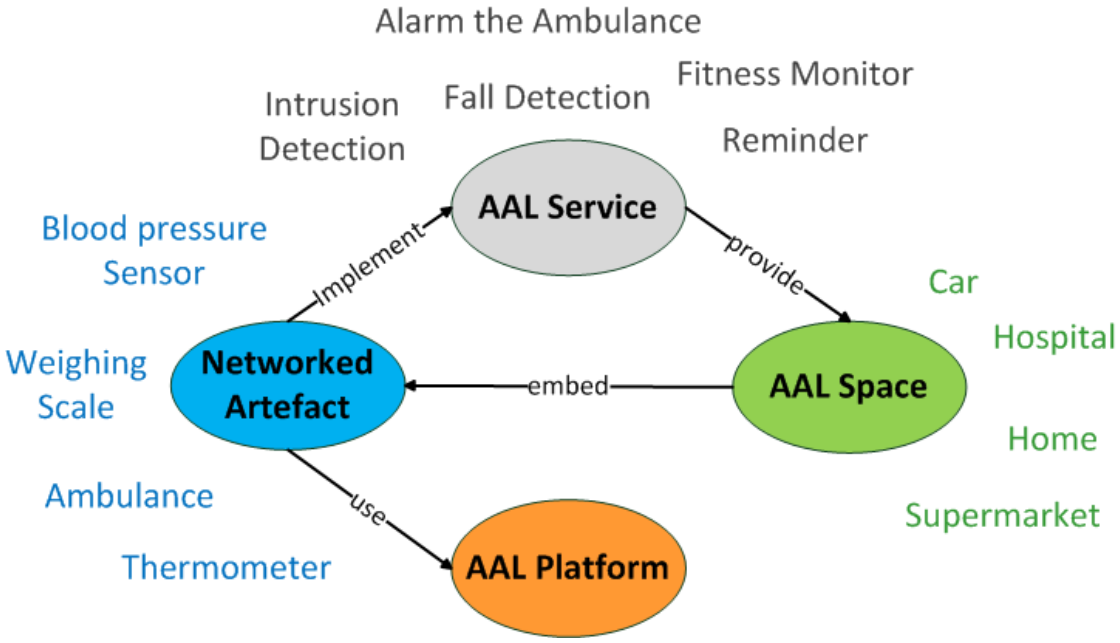


Figure 1.5: Concept Map of AAL Projects
Source: Self-made Figure [69]

Smart Device A smart device is an electronic device that is cordless, mobile, always connected to a network of other devices (e.g. to the internet) and is capable of computation, communication, using internet services and that can operate to some extent autonomously [73].

AAL Spaces are special cases of such smart environments and close off with their domain and its specific purpose. They provide services in the context of AAL. There exist two types of *AAL Spaces*: Private environments, like homes or cars, and public areas, such as hospitals or supermarkets. This differentiation is the same in smart environments.

Furthermore, their respectively inherent artefacts hardly differ. The primary distinction between *AAL Spaces* and usual smart environments can be made when analysing the provided services. In contrast to smart environments an *AAL Service* has the intention to assist the needy person or their caregiver to help them to live independently [69]. This support makes use of Information and Communication Technology (ICT) but is not restricted to it. Any combination of hardware, software, and human resources is possible [69]. These *Networked Artefacts* are embedded in the *AAL Space*, which provides the *AAL Services* that are based on them [69]. This relationship is illustrated in Figure 1.5. Furthermore, the figure depitcs examples for *Networked Artefacts* and *AAL Services*. As mentioned above *Networked Artefacts* are not only devices, such as a body scale or a thermometer, but also include effort taken by humans, like an ambulance or other healthcare professionals. The cooperation among artefacts, which are distributed in an *AAL Space*, is facilitated by an *AAL Platform* [69].

AAL is an interdisciplinary field of study and involves basic technical infrastructure and service delivery in the fields of communication, mobility, self-supply and domestic living. In general AAL systems are spread over four locations [16]:

1. Body Area Network (BAN): Some of the sensors and actuators are mobile and the user carries them around with her or him. These mobile devices are located on the user's body and are used at home as well as outside.

2. Home Area Network (HAN): Ambient sensors and actuators are installed in the home of the patient.

3. Computer and data centres of the service provider: It is not foreseeable which data will be required in the life cycle of a HAN. Therefore, information has to be requested from external databases (e.g. the telephone number of the nearest hospital) [21]. Not only memory but computation capacity is restricted and when extravagating these limits computation has to be outsourced [21]. Remote maintenance and forwarding of data (e.g. transfer of measurement to a physician) are other use cases where external computer and data centres are necessary [21].

4. Third party: Agencies and Organizations which offer electronic or non-electronic services that are implemented by an AAL system, but without being part of it (e.g. weather forecast) [21].

Figure 1.6 illustrates these locations on three different operational levels. On the left hand side there are possible sensors as examples for Personal Health Device (PHD) [14]. Some actuators and sensors, such as pedometers, are inside BANs, while e.g. fitness equipment is installed at the home of the patient and is consequently a part of a HAN.

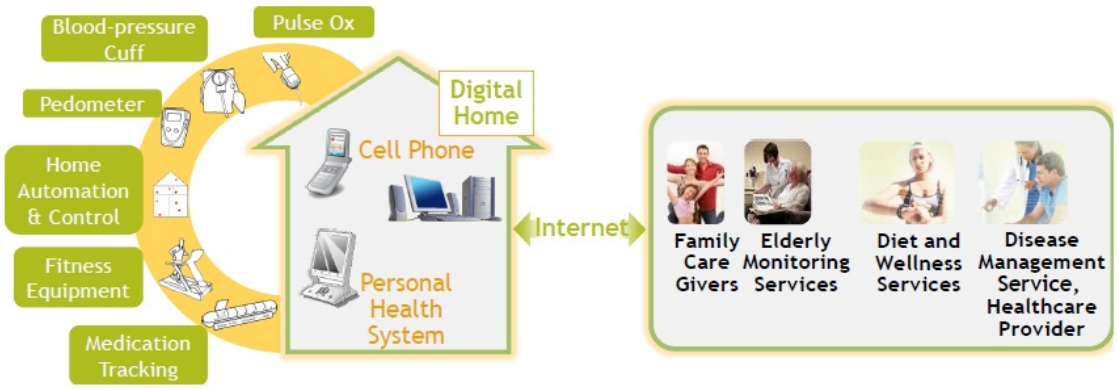


Figure 1.6: An AAL platform consisting of a Personal Health System with Sensors that provides Information to Services over the Internet.
Source: Continua Health Alliance - Original Figure [14]

The organizational structure of an AAL-system is tree-like with PHDs such as body scales or blood pressure monitors as its leaf nodes. They collect biomedical data of the patient and send them to the health device manager of the AAL platform where the data are further processed [46]. In Figure 1.6 this is illustrated in the centre and the manager runs on any kind of Computational Engine (CE). Furthermore, the manager is a distributor of this vital information and sends it to AAL services and caregivers which can be seen on the right side of Figure 1.6 [14].

Before we can think about the demands of the user that have to be met, first we have to clarify the group of potential users of an AAL system. The Deutsche Kommission Elektrotechnik Elektronik Informationstechnik (DKE) named them as follows [16]:

- People without acute exigency for support, but matching desires. E.g.
  - People with an increased desire for comfort and security
  - People with a high interest in wellness and sports
  - People demanding assistance to prevent future problems
- People with acute desire of support. E.g.
  - Older people suffering from age-dependent diseases and functional losses,

  - Patients suffering from dementia
  - People limited in their mobility
  - Patients during their functional rehabilitation
  - People within risk groups (e.g. hypertension)
  - Handicapped people
- People taking care of those requiring support. E.g.
  - Family members or friends supporting the caretaker
  - Cohabitants of people in need
- Healthcare professionals like e.g.
  - Care services
  - Physicians
  - Hospitals
- Employers of older people

We see that users of AAL systems are not just the people in need of help themselves, but the user group is diverse since their caretakers require such services as well [16]. The older citizens are not the only users, but remain as a main target group [16]. In 2012 handling telecommunication and mobile devices is routine for the average population. However, it is not compelling for older generations [16]. Therefore, background knowledge that is assumed for the use of products in other technological branches (e.g. usage of a smartphone) must not be assumed for AAL systems. Yao et al. enumerates several connotations for the ease of use of medical device usage and management at home [75]:

- No obstruction: The device has to respect current workflows and must not impede normal user activity or increase work for instead of making life easier.
- Attention free: The user is not pointlessly bothered and the device claims attention from the patient only if necessary.
- Automatic (device) association: Commonly known as plug-and-play, manual configuration is not required during the device setup. This applies to devices of different manufacturers too, and provides a certain level of interoperability.
- Just-in-time information: User requests enquiring a device information and assistance are answered timely.
- Intuitive operation: Sequencing of the device operations have to be done sensibly and naturally.

- Easy to understand: Whenever a device offers a user interface element or provides support, the lacking background of the user does not make it more difficult to use that element or to follow up on the provided information.

Following these guiding principles usability will increase and user's acceptance for AAL systems will rise [75].

AAL systems are hybrid and combine a technical infrastructure, consisting of hard- and software, and services provided by a third party, like healthcare professionals, call centres or databases [16]. The technical infrastructure is a network of several sensors and actuators [16]. To benefit from the synergy of processing the data of different PHDs, interoperability is a core feature and reliable interfaces are crucial [16]. Unfortunately the majority of device manufactures implement proprietary protocols for communication [44] and automatic device association [75] is not possible. A standardized approach with widespread adoption is lacking and this status quo becomes a problem if devices from different manufacturers have to be connected [44].

Legal Issues

IT-systems in the medical field are possibly medical products and are not operating in an unlegislated area [54]. There exist national laws and European Union (EU) Regulations and Directives to ensure a certain quality-level for software, primarily through special standards. Furthermore, it is regulated how data and especially medical data are allowed to be processed. In Austria the Data Protection Act (DSG 2000), the Health Telematics Act (GTelG and GTelV) and the Medical Product's Act (MPG) become effective under conditions we are going to discuss here. Due to the DSG the utilization of personal data is prohibited [55]. This includes processing as well as transmission. Individual-related data are every kind of information about a person in particular [55]. This applies if the person can be identified directly (e.g. social card number) as well as if she or he is ascertainable by other personal distinguishing marks or patterns [55]. There are six exceptions of this rule (§9 DSG 2000):

- Special legal foundation
- Allowance of the concerned person (valid until revoked)
- The data are not directly individual-related (pseudonymized)
- Vital importance for the person concerned or for a third party
- Scientific research according to §46 DSG 2000
- With the reason of health and patient-centred care

AAL systems have per definition the purpose of supporting people with the need of care and providing healthcare services inter alia [16]. These services process biomedical data and, as they are personalized, they underlie the data protection act [10]. Consequently the purpose has to be legitimated by one of the six exceptions named above. Additionally, before one is allowed

to start recording personalized data it is mandatory to inscribe in the data processing register of the Austrian data protection commission mentioning the concrete data which are going to be used and arguing the intended purpose of the information processing and the legal basis [12]. Furthermore, it is mandatory to protect such data and conform security measures according to §14 DSG 2000. One target group for AAL systems are healthcare professionals accessing the biomedical data of their patients. For the electronic transmission of vital information the Health Telematics Act has to be considered. There are regulations regarding confidentiality, integrity and documentation (GTelG and GTelV). Depending on the classification as a medical product according to §2 MPG different legal guidelines apply. Software is a medical product if the purpose is one of the following ones [54]:

- Detection, prevention, monitoring, treatment or palliation of diseases
- Detection, monitoring, treatment, palliation or compensation of injuries and disabilities
- For the examination, support or substitution of the anatomical build or physiological processes
- Regulation of conception

Certainty of justice is a question of common concern and this subsection points out that it is not easy to render judgement which regulations apply for an AAL product and if it is classified as a medical product. Although we have a complex legal situation it is even more important to pay attention to this topic which is the entire quintessence of this section. We want to point out the general importance and although we spoke about the international dimension of AAL these legal boundaries are regional and the country-specific laws have to be observed.

1.3 Problem Definition

AAL is an evolving field of study [16] and has to overcome challenges [17]. Interoperability is one of them [17] and in the year 2012 there exists no widespread holistic engineering standard specifying the interface between devices and device managers in AAL systems [75]. In this section it will be discussed why such an interface is important and how we implement it.

Main Challenges of AAL

AAL is positioned as a research topic, but there are obstacles to take before AAL systems are market-ready [17]. Eichelberg et al. name six main challenges [17]:

- Robust technical equipment: The current assistance systems are designed for the implementation in laboratory environment and are not robust enough for daily business in normal homes.
- Sustainable business concepts: The systems have to be affordable for the public.

- Usability: Intuitive design is of capital importance for introducing a new product field in which the possible users have little experience.
- Training: Designers and technicians that are trained on AAL technologies are required.
- Integration into the current healthcare system: Apart from fitness applications which are for private use the healthcare system with its stakeholders is the most important partner and has to be involved. Therefore, the data-collecting AAL system has to share its health information.
- Interoperability: Good interfaces which make the different parts of the system communicate with each other are crucial.

Interoperability Interoperability is the ability of two or more systems to work together by communication via interfaces for the fulfilment of a given task [16]. It implies that data are exchanged and correctly interpreted by the receiver. Conformity is a measure of how thoroughly a standard is implemented, and is crucial to ensure interoperability [16].

AAL is a diverse field of research and its platforms are hybrid systems [16]. The manufacturers come from different technology branches and make use of proprietary protocols which interfere with interoperability [44]. Presently, no widespread standard is in use, but several initiatives desire to change this unsatisfactory status quo [44]. In a previous section we discussed demography across the globe and detected a worldwide trend of an ageing population. This development calls for global solutions and points to the importance of cooperation between the different organizations for standardization all over the world [16].

Goals and Limitations

Continua Alliance Continua Health Alliance is a non-profit, open industry organisation of healthcare and technology companies joining together in collaboration to increase the quality of personal healthcare and improve interoperability and compatibility of PHDs with the use of existing standards. Continua is dedicated to establishing a system of interoperable personal connected health solutions [14]. The ISO/IEEE 11073 standard is Continua's main protocol stack [22].

The aim of this thesis is to develop an interface between PHDs and external computer systems following an open standard with the purpose to increase interoperability in the field of AAL. The implementation is open-source and helps as a best-practice for future developers. In order to address them we stick to an existing and approved international standard, the International Organization for Standardization (ISO) and the Institute of Electrical and Electronics Engineers (IEEE) ratified the ISO/IEEE 11073, which standardizes the exchange of vital sign information [37].

In mobile devices a trade-off has to be made between provided resources and their size and weight [22]. Therefore, the computing power has to be located away from the telemonitoring device [22]. Using the ISO/IEEE 11073 standard, devices can be connected via plug-and-play to a health device manager where the vital data are processed [56]. This standard is already popular in the industry and the goal of this thesis is not the development of a new standard, but to apply ISO/IEEE 11073 to a specific application and connect Continua-certified PHDs to an open AAL system [22].

Avoiding proprietary protocols in the future will save costs in case of replacement and provide higher scalability [22]. This will lay the foundation for further developments and increase user acceptance [22].

1.4 Methodological Approach

UniversAAL The UniversAAL consortium consists of major industrial and research players in the field of AAL [68]. The project aims to produce an open platform that provides a standardized approach making it technically feasible and economically viable to develop AAL solutions [68].

We identified the need for an international standard for communication protocols improving the interoperability within devices inside an AAL system. In this section we are going to discuss our strategy to develop such an open interface to fill a pioneering role and give future developers a good lead.

Standards and Protocols

Widespread acceptance is an objective of the UNIVERsal open platform and reference Specification for Ambient Assisted Living (UniversAAL) project [68]. In the past, the ISO/IEEE 11073 standard has got special attention from the healthcare industry [21]. Because of the importance of widespread acceptance we stick to this already existing standard [21]. Therefore, a blood pressure monitor and a body scale are used as evaluation devices. Both devices are Continua-certified and use Bluetooth's Health Device Profile (HDP). Consequently, Bluetooth is the transport protocol this thesis focuses on. Nevertheless, the usability of other protocols, like ZigBee and ANT+, will be analysed as well to give a view that is more general.

Architecture

The first step is to establish a connection between the PHD in the role as an agent and a CE as its manager in order to send ISO/IEEE 11073 messages. This relation between source and sink is illustrated in Figure 1.7. Because we aim to build an open standard it is essential to be careful about licenses of libraries and software we use. After implementing a basic interface to the CE it has to be adapted for the UniversAAL platform. Therefore, the program has to be transformed

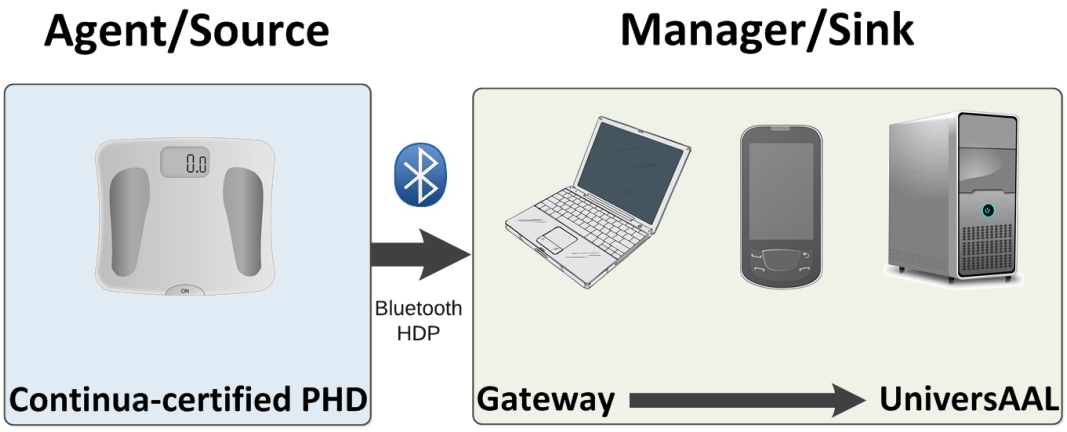


Figure 1.7: Basic Architecture of the Communication between PHD and CE
Source: Self-made Figure

into an Open Services Gateway initiative (OSGi) bundle, and the data have to be mapped to an ontology in order to provide incoming data to other AAL services.

1.5 Structure of the Work

This master's thesis is a respond to the unsatisfactory situation we explained in the previous sections. We pursue the objective to expedite AAL as a field of research. Nevertheless, we do not let the final aim of providing real-world solutions out of our scope. It is not a goal of this thesis to develop a product that is ready for the market. However, while designing our implementation we keep real-world constraints in mind to ease future development.

An overview on the approach and the elementary architecture is what we explained in the section above. This was the first out of four iterations where we go each time further into detail. The next step is a survey of the state-of-the-art. AAL is an evolving field of study and in order to contribute to the researchers' community we build our solution on the results of former projects. In Section 2.1 we tackle the task to analyse completed as well as ongoing projects in the area of AAL. Furthermore, Chapter 2 will discuss which transport protocols would suit for our purpose to connect PHDs to CEs. We focus especially on Wireless Sensor Network (WSN) and conduct a comparative study to point out the benefits and drawbacks of the protocols. We present the technologies we are going to use for evaluation as well as their alternatives. The last section of this chapter will introduce several ISO/IEEE 11073 manager implementations. Reusing existing solutions is favourable due to the availability of a developer community. In this way it is easier for them to get in touch with our implementation. A crucial factor for the reuse of existing code

and libraries are their dependencies.

A broad comparison spectrum of similar projects and other usable technologies is the foundation of a sound decision. However, since this thesis has a practical part we have to focus on one set of technologies to develop a solution that fulfils our demands. This set and the fencing-off of other projects sharpens the idea of the thesis' goals and is the foundation for the development of a concept. Chapter 3 is the next cycle of iteration and we get even more concrete. This chapter will present the theoretical concept of our interface. First we explain the UniversAAL platform and then we go on with the general purpose and value of ontologies. Due to the omnipresence of the ISO/IEEE 11073 in this thesis we dedicate a section to its structure and explain the different parts of this set of standards. The conclusion of this conscientious analysis reveals the architecture of our implementation.

The last chapter is dedicated to the actual development of the software and puts the designed concept into practice. We realize the connection of the separated parts of the implementation. On the one hand we introduce a gateway to read in vital data (ISO/IEEE 11073) to the UniversAAL platform and the library used for this process. On the other hand we discuss the code to feed this data to UniversAAL's middleware in order to make it available to other AAL services.

CHAPTER 2

State-of-the-Art

THE demographic change is the primary motivation for the AAL sector. Due to this trend being global international solutions are necessary [16]. The majority of the manufacturers of AAL products use proprietary protocols to exchange data between PHDs and CE [75]. The market is fragmented due to the development of AAL products into several different market segments like mobile phones, media servers or healthcare devices [44]. A standardized interface is the first step for interoperable frameworks that involve devices from different manufacturers [75]. Hence it is expedient to establish an international standard and provide ways to test new products for operational reliability [16].

Non-Functional Requirement In systems engineering and requirements engineering, a non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviours. This should be contrasted with functional requirements that define specific behaviour or functions. Non-functional requirements are often called qualities of a system because they cannot be quantified [73].

In 2012 there exists no widespread standard for transmitting vital information on device-level. Therefore, we aim to implement such an interface and implement a helpful best practice for future developers. Widespread acceptance is a non-functional requirement of ours. As we have a market of globally operating companies, our solution has to be internationally applicable. The desires of a broad spectrum of branches have to be met due to the different backgrounds of the manufacturers. These characteristics of AAL clarify our goals and influence our decisions.

In this chapter we will start with an evaluation of existing AAL projects and go into detail on the UniversAAL project. This AAL platform will be used for the implementation of our interface.

ZigBee, ANT+, and Bluetooth are the most popular wireless transport protocols. Here in this chapter their benefits and drawbacks will be compared. The main emphasis is on the one hand the availability of protocol stacks on the different operating systems and their quality. This is mutually dependent on popularity in the developers' community. On the other hand the wireless transport protocol's spreading at the PHD manufacturers' side of the road is a key criterion that has to be observed.

In the last section of this chapter several ISO/IEEE 11073 manager implementations will be introduced. We focus on open source software and discuss the architecture of four relevant solutions. This analysis is meant to point out possibilities and discover reusable libraries.

2.1 AAL Projects

In order to gain a better insight in the sector of AAL we will introduce some sample AAL projects. This overview provides a basis for deeper understanding of the current status of this field of study and the branch's present scope of tasks. Since we have to narrow down our selection we focus on recent flagship projects, preferred Integrated Projects of the European Commission.

SOPRANO

The Service-oriented Programmable Smart Environments for Older People (SOPRANO) initiative was an Integrated Project of the European Commission's 6th Framework Programme [61]. The goal was to develop an AAL system to enhance the lives of senior citizens and facilitate them to live independently and play a full role in society [61]. It was a consortium of commercial companies, service providers and research institutes with over 20 partners from Greece, Germany, United Kingdom, Netherlands, Spain, Slovenia, Ireland and Canada [61]. The project started in February 2007 and had a time horizon of 40 months [62].

Figure 2.1: Soprano Logo
Source: soprano-ip.org

A major objective of SOPRANO was to improve the way users interact with their Smart Home. It involved older persons and also their caretakers, especially professional care personnel and caring relatives, to increase the value of the smart living environment. It was not the goal to develop an AAL system that acts passively and reduces itself only to receive commands, but it aimed to act as an informed, friendly agent that takes orders, gives advice and reminds if necessary [62]. The SOPRANO system was designed to be modular with a scalable Service Oriented Architecture (SOA) [62]. The reason for this was that every user has different necessities and preferences and hardly anyone has need for the full system including all possible modules [62]. Additionally it is possible to offer alternative modules in such an architecture [62]. This im-

proves acceptance and includes sub-groups in the target group [62].

SOPRANO's approach was to move away from technology-driven research towards a more user-centred design [61]. To fulfil this aim it was necessary to start the project on an initial stage that is earlier on the time line of project development because the analysis of the market and the needs of the target group becomes a major task. To identify the needs and requirements of older citizens SOPRANO adopted the Experience and Application Research (E&AR) design method [61]. This method involved possible end users in all stages of Research and Development (R&D) and the product development life cycle [61]. This started at the requirements election phase and the research process and later on they accompanied the project further during development and design phase [61]. In contrast to this method classic projects involved users only for testing of the prototype and the product. It was not the tester who responded to the ideas of the developer, but rather the developer who listened to the suggestions of the older people as well as their caregivers [61]. Secondary, she or he responded to their input on the product [61].

This first phase was carried out in 14 focus groups with more than 90 participants as well as in individual interviews [61]. The outcome was the following set of subject areas [61]:

- Social isolation: e.g. loneliness, depression, boredom, social exclusion and disruption of patterns of daily living
- Safety and Security: e.g. falls, disorientation, control of household equipment
- Forgetfulness: e.g. taking medication or finding objects in the house
- Keeping healthy and active including physical and mental activity: e.g. exercise, good nutrition, daily routines and adherence to medication
- Community participation and contribution to local community
- Accessing information/keeping up to date
- Getting access to shops and services
- Quality management of care provision
- Mobility inside and outside the home: e.g. walking in the neighbourhood and use of public transport

The next step was the technological development based on these subjects [61]. A feature of SOPRANO is the use of multimedia equipment to remind, encourage and support people in their daily life [61]. Examples are information messages on the screen while watching television [62]. By doing so the user does not miss e.g. the doorbell because the television set is too loud and the person is focused on it [62]. Another idea is the use of an avatar on the television set to meet special accessibility and usability needs to e.g. follow an activity program [71]. The avatar can interact with the person and provide instructions [71].

More than 50 users lab-tested the prototypes and contributed in SOPRANO [62]. Based on these test data developers refined the usability and improved the overall system and its different components [62]. The field trials consisted of testing the full-function on the one hand and a large-scale demonstration on the other hand [62]. They were carried out with 300 users each [62]. The project SOPRANO ended in February 2011 [62].

OASIS

The consortium of the Open Architecture for Accessible Services Integration and Standardisation (OASIS) project consisted of 33 partners and 5 subprojects from 12 countries [53]. It was an international project with partners all over the world. The majority of the partners came from Europe and other members of the consortium were from China and Mexico [53]. This was favourable for OASIS regarding a wide coverage of cultures, mentalities and needs [53]. The consortium included all key actors as members, namely representatives of the industries, telecommunication providers, industrial Small and Medium Enterprises (SMEs), research centres, universities, end-user organisations and an association of cities [53]. To introduce a framework a critical mass of users is necessary to convince the market. The project was set up in the beginning of the year 2008 and ran 40 months until December 2011 [53]. OASIS was an Integrated Project of the European Commission with the scope to improve the interoperability, quality, variety and usability of AAL services and products [53]. Home environments are very heterogeneous and the application domains use different communication standards which are incompatible and the products of different manufacturers isolate each other [40]. This limits their functionality and it does not enable collaboration among them [40]. OASIS followed in the footsteps of the AMIGO project and chose the path of shared information management [40]. It defines a common ontological framework for content and functionalities [40]. The aim was to develop and deploy the following deliverables [53]:

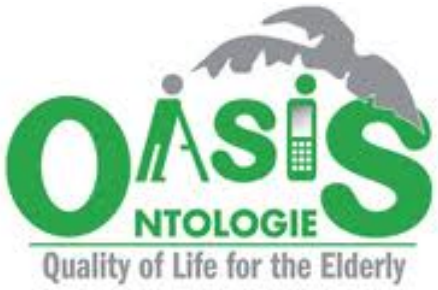


Figure 2.2: OASIS Logo
Source: aal.fraunhofer.de

- New Architectural Framework: This framework is called OASIS hyper-ontology in OASIS's wording. The OASIS project's objective was to enable the sharing of data and resources between heterogeneous AAL services. The hyper-ontology is open, easy-to-use and modular. Sharing the content between two single services is a good starting point, but OASIS went even one step further and its goal included connecting existing and emerging ontologies of the same or different application domains.

- Open Reference Architecture: This architecture is an open source software tool, that enables the automatic or semi-automatic connection of services and ontologies to the OASIS Architectural Framework.

- OASIS System: The system extends the *Open Reference Architecture* and adds an Ambient Intelligence (AmI) Framework and the Interaction Platform.
- Wide range of Connected Applications: Some applications were developed to demonstrate OASIS's use.
- Pilots test-bed: Test environments were set up at four sites. They included living labs as well as sheltered homes to test real world applications.

Although, there were over 30 exploitable products developed for the OASIS target group the developers claim the *Open Reference Architecture* as the most important deliverable of this project [53].

OASIS's approach was to integrate AAL services through the use of a common set of interconnected and extensible ontologies [40]. The unification of the domotics environment for interoperability reasons and the integration of technologies, products and services in the domestic environment were major tasks of the project [40]. In the design phase the functionalities, the project focused on, were defined [40]:

- Security and Safety: E.g. monitoring of the environmental parameters, such as reminder for locking, security procedures like alarm and warning devices, or safety procedures in case of storms
- Energy saving: E.g. energy efficient consumer products such as an efficient climate control
- Comfort: E.g. central remote control to easily change the settings of functions in the household like the desired temperature in the house
- Flexibility: E.g. flexible connection technology and wiring and a comfortable multimedia infrastructure
- Entertainment: E.g. home cinema, home server and networking
- Additional functions: E.g. emergency call, monitoring of subject vital parameters, telemedical services or supporting housework

Ontology In ICT, an ontology is the working model of entities and interactions in some particular domain of knowledge or practices, such as electronic commerce or the activity of planning. In Artificial Intelligence (AI), an ontology is, according to Tom Gruber, the specification of conceptualizations, used to help programs and humans share knowledge. In this usage, an ontology is a set of concepts, such as things, events, and relations. They are specified in a specific natural language) in order to create an agreed-upon vocabulary for exchanging information. [72]

During the OASIS project an ontology has been developed in order to enable the sharing of data and resources between heterogeneous domotic appliances [40]. This aims on a contextual

awareness of environmental and subject-related conditions for AAL services [40]. This ontological framework is based on universals and real world instances are added and assigned later on [40]. The four principles realism, perspectivalism, fallibilism and adequatism were applied in the development of the ontology [40]. This and needs of the developers led to three different approaches considered for the design of the inner structure of the ontological framework [40].

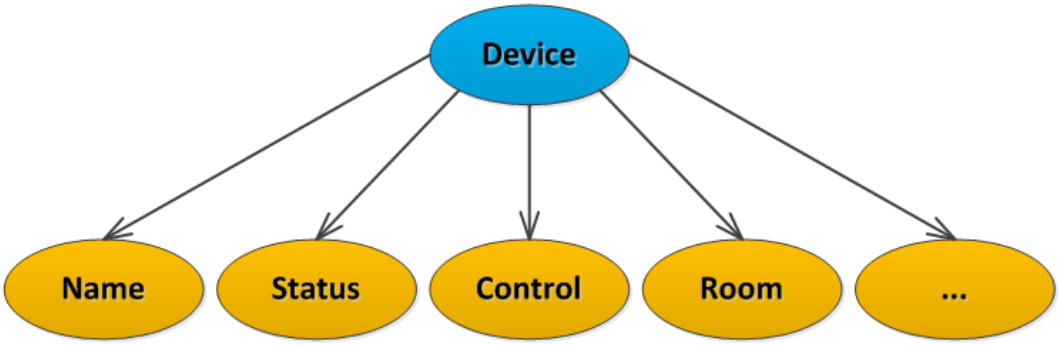


Figure 2.3: Ontology Scheme - One class fits all devices
Source: Self-made Figure [40]

A straightforward solution is to make one class device that applies for all kind of different devices [40]. The benefit of such an architecture is the simplicity, but this convenience has a drawback as well. It is necessary to investigate similarities and find the highest common factor. As illustrated in Figure 2.3 the common attributes are associated with the class *Device* and would make a lot of core functionalities inaccessible by the overall AmI system [40]. Only very basic functions are included and it would require further customization steps during the development [40]. Table 2.1 demonstrates this behaviour by two examples, light and washing machine. They have the properties *Name*, *Status* and *Room* in common, but a light bulb has no *Spin Speed* or *Water Temperature*. The other way round the *Washing Machine* has no *Luminance*.

In order to target this issue the second approach has a contrary philosophy than the previous one. Figure 2.4 explains the concept of having one class for every device. This enables the ontology to handle specific properties (due to reasons of clarity not illustrated in detail) and exhaust all the full functional possibilities [40]. This method has the disadvantage that the developer has to put a lot of effort into providing the appropriate device description classes [40].

Figure 2.5 depicts a compromise between the previous two concepts. In this design devices with common characteristics and types are united into categories. Sample classes are heating devices, brown goods or alarm and security devices. The figure uses the example of *White Goods* and its child nodes *Oven* or *Refrigerator*. In this approach the upper layer is the abstract generic ontology while the lowest one is the detailed functionality layer and the categorization is in be-

Property Name	Example 1	Example 2	Description
Name	Light	Washing Machine	The name of the device.
Status	On/Off	On/Off	The current status value of the device.
Room	Living Room	Bathing Room	The room name where the device resides.
Luminance	100%	-	The current luminance of the light bulb.
Spin Speed	-	1400	The selected spin speed for the washing program.
Water Temp.	-	30C	The selected water temperature for the washing program.

Table 2.1: Light's and Washing Machine's Properties
Source: Kalogirou et al. [40]

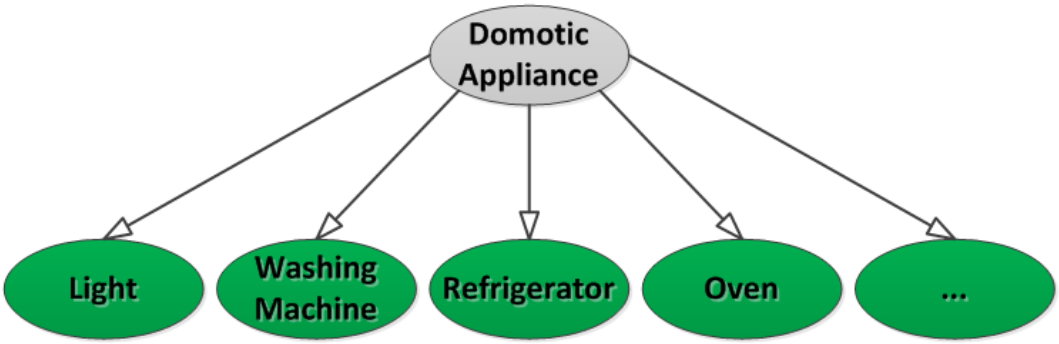


Figure 2.4: Ontology Scheme - One class per device
Source: Self-made Figure [40]

tween them [40]. If the vendor does not supply a detailed description of the product it is still possible to fall back on the more general category [40] and use the basic functionality that is defined in here. This results in a much higher modularity and flexibility [40].

OASIS's innovation is provision of a platform to connect different products and initiatives in the field of AAL [40]. To improve the interoperability it makes use of the method of unification and its tool is the ontology [40]. Its goal is to act as an umbrella for various functionalities and communication standards and it achieves this aim by developing an ontology that is an interface

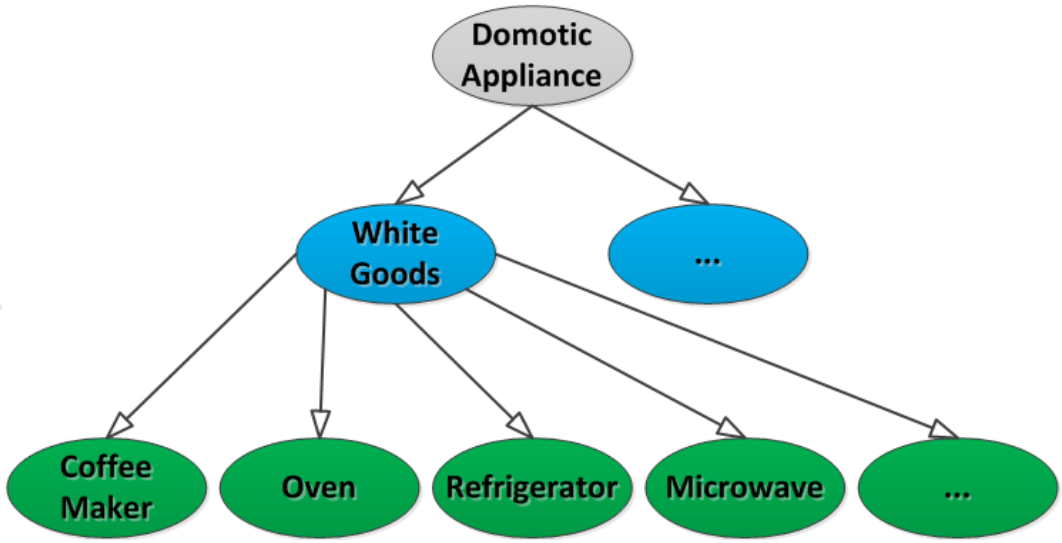


Figure 2.5: Ontology Scheme - One class per category
Source: Self-made Figure [40]

and passes on information [40]. This interface has no restriction to specific communication protocols and is easily expandable and not brand-specific [40].

CompanionAble

The CompanionAble project started on the 1st of January back in 2008 and should be completed by the end of the year 2012 [13]. In the project involved are 18 partners from all over the European Union [13]. Coordinator is the School of Systems Engineering at the University Of Reading, United Kingdom. The consortium included a partner from Austria, namely the Austrian Institute of Technology (AIT) (in the beginning of the project still known as Austrian Research Centers (ARC)).

Different than the previous projects, CompanionAble is not a framework for AAL services, but a service or a bundle of services itself. In Chapter 1 we discussed the demographic trends and the changing ratio of older citizens and possible caretakers. People have the desire to grow old at home and ICT plays an important role in prolonging independent living [32]. It is not possible nor wished to replace human professional caregivers and caring relatives, but the goal is to support them by

Figure 2.6: CompanionAble Logo
Source: legrand.fr

helping and relieving them from as many duties as possible. Smart Home technology is capable of interoperation with many assisting services and devices [32]. A personal assisting robot can fulfil some of these tasks and has the benefit of mobility [32]. CompanionAble goes even one step further and wants to develop integrated cognitive assistive and domotic companion robotic systems for ability and security [13]. The project focuses on cognitive stimulation to support the older people in preventing dementia and depression [13]. Social exclusion increases the risk of contracting these diseases [13]. CompanionAble provides a solution that concentrates the synergies of an embodied mobile robotic companion that works collaboratively with AmI technologies, typically a stationary Smart Home environment [13]. It combines the advantages of the Smart Home [13]:

- Wide Spectrum of Functionality: There are already many existing home automation solutions for all kind of issues. They can be easily included into the CompanionAble framework.
- High Acceptance Rates: An increasingly number of households uses Smart Home services.
- 24/7 Reliability: These automated systems have no leisure time and no holidays. They run 24 hours - 7 days a week.
- Interoperability: Adding new devices and their interconnection does not work unobstructed, but there are several initiatives working on this issue and the trend is towards the right direction.
- Barrier-free: Stairs or other constructional obstacles do not trouble the Smart Home system too much.
- Simultaneous Monitoring: The sensor system allows monitoring of all rooms simultaneously.
- Tele-Monitoring: Easy remote access to the sensor system and its controllable services and devices.
- Low Maintenance Costs: Once installed there are hardly any running costs later on.

Many of these advantages are seen in contrary to a robot system. Interoperability and high acceptance rates are only good in this relative context and when talking about remote access we do not examine the critical aspects, but only analyse the possibility. On the other hand the brownie points of the Mobile Robot Companion are [13]:

- Human-Like Behaviour: The robot is a real interaction partner that is embodied and has a natural interface. In simple words: It is something to touch.
- Visible Intimacy and Privacy: Embodiment gives back the control over the privacy. When the robot companion is in another room it is only at this single location and the intrusion into privacy is obvious and transparent.

- No Installation: The companion can do its job everywhere and there is no need to install it to the environment.
- Mobility: It can support wherever help is needed.

These advantages supplement each other and together they build a synergistic combination [13]. CompanionAble is designed to support personal therapy and involve home information spaces for this task [13]. An example can be a television set like in the SOPRANO project. Furthermore, health education for patient and caregiver is a field of application [13]. CompanionAble provides services for social inclusion and is able to help with improvement of contacts between the person and her or his caretaker [13]. It supports older people with an intelligent day-time-management and a context-aware reminder function for their medication [13]. Through the integration of the two sub-systems CompanionAble is capable of analysis of emotions [32] and facilitates audio-visual communication with relatives or other caregivers [13]. Its support involves cognitive stimulation, training and games to prevent diseases such as dementia [32]. It serves the needs of persons with minor cognitive impairments and provides a cognitive assistive companion [32]. If necessary the robot can be remotely controlled, e.g. by a therapist or a relative [32].

Figure 2.7 demonstrates how the CompanionAble robot looks like. On the left there is a pre-final version of the robot that has been used for functionality tests and user trials in the years 2010 and 2011 [26]. The right side shows the final version that has been used to evaluate CompanionAble [26]. This evaluation takes place in France, Spain, the Netherlands and Belgium [13]. The synergistic work of Smart Homes and the Mobile Robot Companion is tested there and this performance evaluation will reveal the effect of the CompanionAble Framework Architecture [13]. The regional segregation provides an evaluation that is not biased, but gets feedback from people with different nationalities, cultures and background and due to that with different needs [13].

General Description of UniversAAL

The UniversAAL project started in the beginning of the year 2010 and is the latest project of those which we are describing in this section. Its time horizon is 48 months and it is, like SOPRANO and OASIS, a large-scale Integrated Project and part of the European Commission's 7th Framework Programme (one generation after SOPRANO). UniversAAL involves 20 partners including the AIT.

Figure 2.8: UniversAAL Logo
Source: aal-kompetenz.de

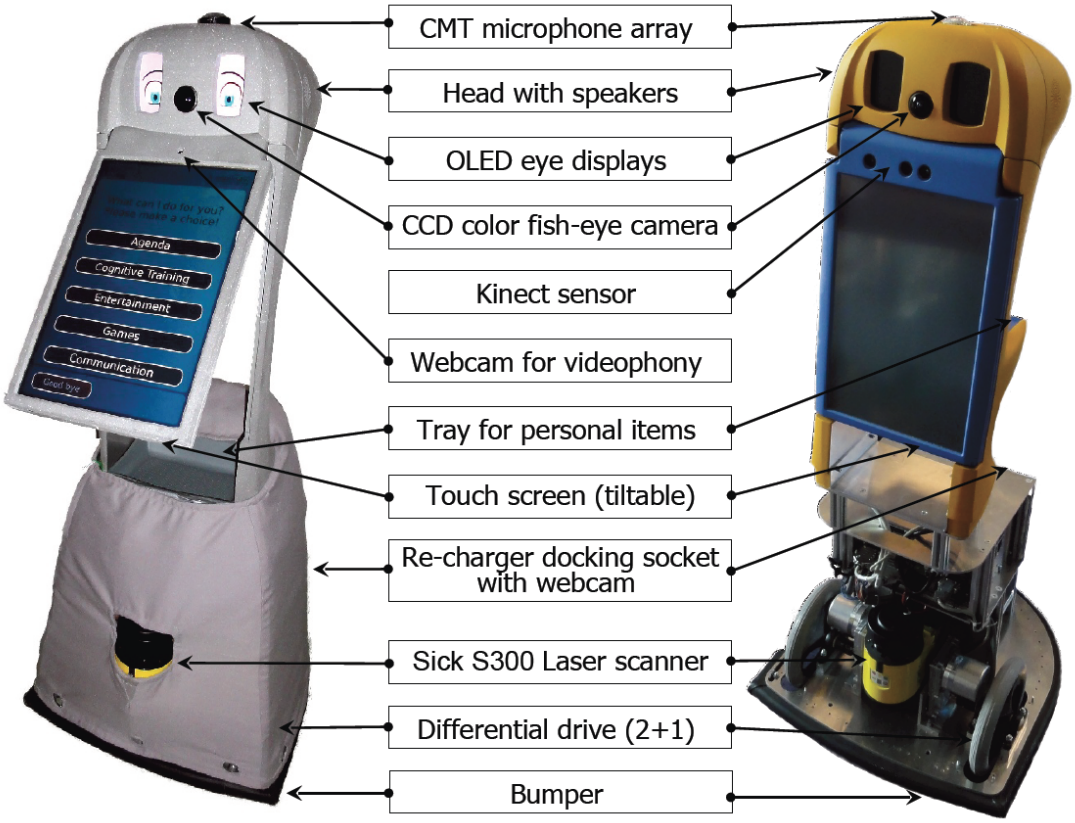


Figure 2.7: The robot companion developed in the CompanionAble project - Prototype for trials in 2010 and 2011 on the left and final version on the right.
Source: Gross et al. - Original Figure [26]

App An application is software designed to help the user to perform specific tasks. It is contrasted with system software and middleware, which manage and integrate a computer's capabilities, but do not directly perform tasks that benefit the user. In 2012 App usually means mobile App dedicated to smartphones, tablet computers and other mobile devices. [73]

Previously, we revealed the need for an international standard for communication protocols that are used in AAL systems. Especially the interoperability among PHDs and CEs has been identified as a major issue. In the previous subsections we introduced some other projects from the area of AAL. This is a good foundation to understand the character of UniversAAL and its goals better. Thus the delineation of contents is easier.

Figure 2.9: UniversAAL Slogan
Source: universaal.org

This subsection is dedicated to introduce the UniversAAL project and its AAL system. We discuss the strategy to develop such an open platform and fill a pioneering role to give future developers a good lead. Later on we go more into detail and in Chapter 3 we discuss UniversAAL's concrete architecture.

The UniversAAL project aims to reduce barriers for new AAL products. Its goal is to develop a holistic reference architecture showing how to connect AAL products best [68]. The main objectives of this project are [68]:

- Develop an open platform and provide technical support.
  - Establish a *uStore*: Like Apple's App Store© [7] for Apps, the *uStore* will be an online marketplace for AAL services for the end-user with content provided by developers.
  - Design and establish the UniversAAL developer depot: The developer depot is an online platform to support developers at their work and provide documentation of the reference system.
  - Consolidate existing work and integrate with new development: The platform is a mixture of new developments and state-of-the-art results of previous projects from existing initiatives.
  - Achieve interoperability amongst the platform's elements: Since lack of interoperability in current systems is a key factor for the launch of the project itself, a strategy has to be elaborated to make UniversAAL an improvement to the status quo.
  - Open design: The platform shall allow not only internal interoperability but external, with other existing systems and domains, as well.
- Promote widespread adoption for the platform.
  - Wide range of actors: The more actors that are working in the AAL sector are involved in the project the better the widespread acceptance is going to be. Accordingly it is a crucial objective to convince actors such as device manufacturers of the project.
  - Active promotion: Actively promote adoption of the UniversAAL platform. The success of the project is measured by its spreading.
  - Demonstrate the practical usefulness: This is done by providing sample code to demonstrate the best practice, and developing proof-of-concept scenarios.

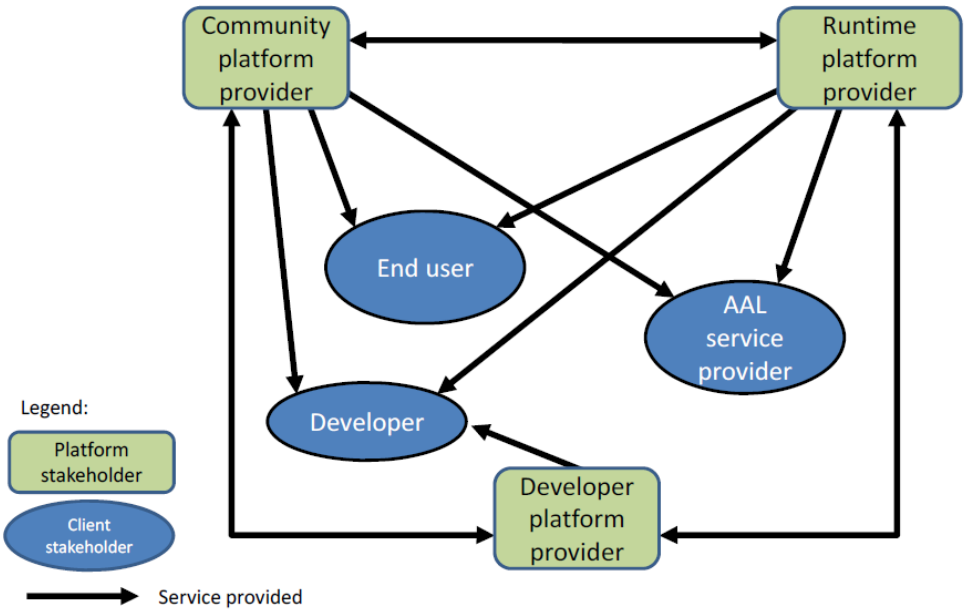


Figure 2.10: The AAL value network supported by UniversAAL.
Source: UniversAAL - Original Figure [69]

Community building In this context we principally mean online communities. Community building in general is not an ICT problem, but an organizational science problem.

Advancing the sector of AAL is the main objective of UniversAAL. Therefore, we have to investigate who are the stakeholders which are inherent in the domain of AAL. We identified two groups of stakeholders and Figure 2.10 illustrates their value network:

- *Platform Stakeholder*: These stakeholders provide and operate platform components to offer services of value to clients. Clients in this value network are either *Client Stakeholders* or other *Platform Stakeholders*.
  - *Runtime Platform Provider*: The AAL runtime environment facilitates an AAL space to provide services to the user. It is the contact point between the system and the users. Therefore, it is the place where they interact with AAL technologies. The *Runtime Platform Provider* provides this environment itself and corresponding technological support.
  - *Developer Platform Provider*: The developer platform focuses on the software developers and provides software engineering support for proper development, testing and maintenance.
  - *Community Platform Providers*: This platform provides a technological infrastructure for all *Client Stakeholders*, *End Users*, *Developers* and *AAL Service Providers*

to support communication among them. This community is the key issue for all kind of collaboration among service providers as well as the trading of services or collecting user requirements.

- *Client Stakeholder*: These are stakeholders who use one or more of the named platforms. The use is either liable to costs or for free. The developer platform affects hardware developers not directly because it provides software engineering support. Nevertheless, it concerns them indirectly because their design decisions open and close doors for the software on the top of it. It is crucial for the *Developer Platform Provider* to predict which technologies the *Developer* wants to use (e.g. choice of the wireless transport protocol, such as Bluetooth or ZigBee). On the other hand the hardware developer lightens the software developer's workload by sticking to supported technologies.

  - *End Users*: Assisted persons, their families and caregivers are only some of the *End Users* of AAL services. Apart from them people with a high interest in fitness or employers of senior citizens are other examples. In Section 1.2 a full list of the consumers of AAL services can be found. *End Users* are principally concerned with service quality and usability of AAL services. The better the development environment for the engineers is, the better the quality of the software is going to be and the more developers will be attracted by the domain. Additionally the user benefits from efficient communication with service providers such that the understanding is good and the user's expectations can be met better. The end user hardly benefits from the platforms in a direct way, but in the end it is all about the customer.
  - *Developer*: This includes everybody who develops AAL solutions and technologies. From the technical point of view we have networked artefacts which are software components that run on hardware nodes. Hence we identify hardware developers and software developers as stakeholders.
  - AAL *Service Provider*: As the name expresses these are the providers of AAL services to the *End User*. From the communication point of view they are interested in efficient interaction with *End Users* as well as *Developers*. Furthermore, service providers are concerned with fluent interaction with the runtime environment.

Figure 2.10 reveals the connection between the several stakeholders. The rounded rectangles are the *Platform Stakeholders* while the ovals are the *Client Stakeholders*. The *Developer* is multifaceted and has to deal with provided information from all sources [69]. The *End Users* and *AAL Service Providers* are only interested in the services provided by the *Community Platform* and *Runtime Platform* [69]. Furthermore, stakeholders exist that are not in this graph. Deployers design, construct and configure AAL spaces [69]. However, they do not fit in the model in Figure 2.10 since they are an intersection between *Developers* and *AAL Service Providers*. Therefore, we do not need to name them separately. Another withhold stakeholder are authorities. They are not addressed in the reference architecture, but their expectations are being considered in designing it (e.g. audits or privacy protection) [69].

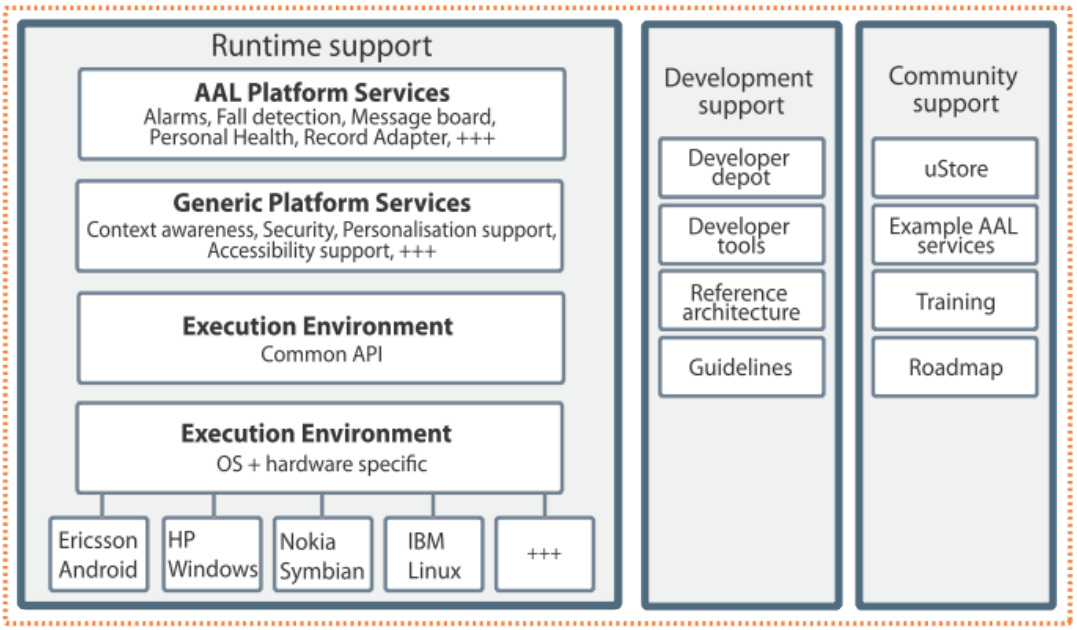


Figure 2.11: The scope of the development within UniversAAL.
Source: UniversAAL - Original Figure [69]

Platforms are a double-edged sword. On the one hand, they have the positive effect of accelerating service deployment by providing reusable functionality. On the other hand, platforms can inhibit innovation by imposing too many constraints on service providers. Finding the golden mean is one of the main challenges of UniversAAL. The goal of UniversAAL is to support these stakeholders in developing, trading and using AAL solutions. In Figure 2.11 we see the three scopes of duties (to reach UniversAAL's goals) that have been identified [68]:

- Development and supply of an infrastructure to run the implementation: The *Runtime Support* (the left-most box in Figure 2.11) for the execution of services and applications is the UniversAAL platform itself. Therefore, the software development effort is bigger for this part than for the other deliverables. The project is open source and the primary goal of the *Runtime Support* is to provide an infrastructure for developers of AAL services so they can focus on this work rather than having troubles with the underlying platform. Since the platform is not built for a specific environment and purpose, it is up to fulfil the requirements of all developers and domains. It is indispensable to implement it generic and platform-independently. The figure illustrates that the UniversAAL platform does not depend on a specific operating system it is running on [68].

- Support of developers: The *Development Support* (the middle box in Figure 2.11) assists software engineers during the development process. It includes guidelines and tools for the

developers to ease implementing AAL services for the UniversAAL platform with minimal effort. The *Developer Depot* makes resources and the program code available. This includes sample code and tutorials [69]. Especially for new developers this is a good starting point because they can take a look at best practices and thereby receive an impression about the possibilities. The reference architecture is more general than the development of the UniversAAL platform itself because it is independent of specific concrete solutions, such as protocols, technologies or products [69]. Its purpose is to facilitate the implementation of the AAL platform and improve fundamental understanding [68]. Later on in this subsection it will be discussed how this basic architecture has been developed.

Killer Application In marketing terminology, a killer application is any computer program that is that necessary or desirable that it forms the core value of some larger technology. A killer app can substantially increase sales of the platform on which it runs [73].

E-Commerce Electronic Commerce refers to the buying and selling of products or services over electronic systems such as the internet and other computer networks [73].

M-Commerce Mobile Commerce is the ability to conduct commerce using a mobile device, such as a mobile phone, a Personal Digital Assistant (PDA), a smartphone, or other emerging mobile equipment [73].

Social Commerce Social commerce is a subset of electronic commerce that involves the use of social media, online media that supports social interaction and user contributions, to assist in the online buying and selling of products and services [73].

- General support of the community: An active community is the core aim UniversAAL is heading to and collaborations among communities of developers, service providers and end-users are supported by an organizational infrastructure to exchange products on the one hand and knowledge on the other hand [69]. The *uStore* is the killer application of the *Community Support* (the right-most box in Figure 2.11). It is an E-Commerce solution that considers the special requirements of UniversAAL. There are standard solutions for web shops, but unfortunately these are proprietary and cannot be used as a base solution [69]. Current trends in E-Commerce are M-Commerce and Social Commerce, which are suitable for the *uStore* as well [69]. Following the principle of Apple's App Store©it is an online marketplace for AAL services including hardware, software and human resources. Furthermore, end-users have a one-stop-shop to find suitable easy-to-use plug-and-play AAL solutions which can be installed on modern operating systems in a simple way. It is a forum for all kind of stakeholders to organize the process of offering and searching, and of selling and buying in a comfortable, straightforward and fast way. The *uStore* offers example AAL services [69].

In comparison to the code samples which are offered for the software engineers, these examples are addressed to end-users, and rather explain what an AAL service looks like and how to install it on the user's system than to support the development of new software [69]. Since AAL is a comparatively young field of study only few specialists exist [68]. Therefore, trainings for new developers and installers as well as end-users take care of this issue [68].

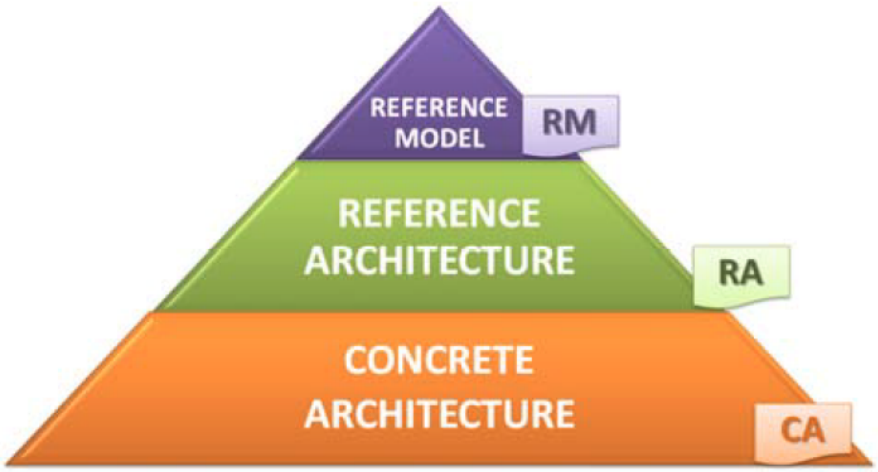


Figure 2.12: UniversAAL overall model
Source: UniversAAL - Original Figure [69]

We discussed UniversAAL's holistic approach to the AAL value network and the different levels of support. Here we focus on the AAL platform itself. The concept of the platform does not follow the stakeholder's view we had before when reflecting the levels of support. The UniversAAL platform has three abstraction levels, and each one has its own scope and semantics [69]. For the design the knowledge is derived from analysis of other projects in the AAL domain [69]. In Figure 2.12 they are enlisted [69]:

- *Reference Model*: The model is an abstract concept framework that specifies significant relationships among relevant real-world entities and is the top level of abstraction, as illustrated in Figure 2.12. It is rather an abstract view on the structure of the AAL domain than on the UniversAAL platform itself. Large problems are broken down into smaller pieces that can be understood, tackled, and refined. Furthermore, it is the starting point of the platform's architectural design and provides a context for understanding of the *Reference Architecture*.

- *Reference Architecture*: The middle part is the *Reference Architecture* which provides a view on the abstract architectural elements in the domain and how they exchange data with each other. It is independent of specific technologies, protocols, products or any other feature of the real solution. As a part of the *Development Support* it facilitates the development of the platform and its concrete architecture by defining abstract interfaces.

The *Reference Architecture* is sandwiched between the *Concrete Architecture* and the *Reference Model* and derives from both of them. It is designed bottom-up from empirical data provided by the *Concrete Architecture* and collected during the consolidation process where other AAL projects were analysed. Additionally it is designed top-down from the *Reference Model* in order to introduce the technical contents in an abstract way. It is the consensus between these two points of view and assures that this central level is an interface to both of them.

Application Programming Interface (API) An API is an interface a software system provides to enable software components to communicate with each other and the system. An API may include specifications for routines, data structures, object classes, and variables. However, mostly we would think of a set of libraries of a programming language, such as the Java API.

- *Concrete Architecture*: The *Concrete Architecture* is not a complete component specification, but adds concrete solutions, such as APIs or protocols, to the abstract *Reference Architecture*. The main goal is to reveal dependencies and overlapping in terms of shared data on a high level in order to guarantee coherence of the reference implementation of the UniversAAL platform. Its purpose is to gain a more generalized view on the UniversAAL architecture.

There are numerous research projects and industrial labs dedicated to the area of AAL and this number is constantly increasing [69]. There are other projects in this field of research which discussed similar questions of architecture, technologies and protocols. Since the UniversAAL project builds on the results of former projects, it is necessary to analyse related work and identify and utilize available general-purpose solutions and identify reusable results [69]. Therefore, evaluation of related projects is the first step and a consolidation process is part of the design process of the architecture [69]. Own solutions are important for the identification of a project, but UniversAAL sees itself as a pioneer in reusing existing AAL technologies. Examined input projects are AMIGO, GENESYS, MPOWER, OASIS, SOPRANO, and the PERceptive Spaces prOmoting iNdependent Aging within dynamic ad-hoc Device Ensemble (PERSONA) project [69]. Figure 2.13 lists the background projects and the corresponding project members. The projects are either projects of the 6th or the 7th Framework Programme of the European Commission. We discussed two of them, SOPRANO and OASIS, earlier in this section. PERSONA is another crucial project UniversAAL is based on and has a big influence including reused source code.

UniversAAL is our evaluation framework and the developed interface connects PHDs to this specific AAL system. This development takes place in cooperation with the AIT which is partner in the UniversAAL project. In the subsequent chapter the concrete architecture of this framework will be discussed and we will go into technical details.

Figure 2.13: Background Projects of UniversAAL
Source: UniversAAL - Original Figure [68]

2.2 Transport Protocols

In Section 1.2 we discussed AAL's field of research and the influence of ubiquitous computing and networks of smart devices that cooperate seamlessly and are usually cordless. In the following section first we are going to think about the desirable characteristics we have for transport protocols. Afterwards, we will go more into detail on a selection of possible transport protocols and compare them later on in order to point out the benefits and drawbacks of the several protocols in particular. The specifications of the protocols have been developed by separate alliances of companies and the fields of application differ. Our aim is to evaluate which protocol suites best for us and what are the reasons for this decision.

Desirable Characteristics of out Transport Protocol

There exist so many different standards and technologies for transport protocols that first we have to make a selection before we start our comparison. Our specific application aims to connect PHDs to CEs. These PHDs are sensors and actuators that are principally smart devices that are mobile and battery-powered. The first limit we put is the transmission medium used for communication. We focus on wireless transport protocols with a minimum range of 10 meters. Technologies with the use of cables, such as Universal Serial Bus (USB), are not practical for daily use of wearable sensors. Since we focus on BANs and HANs the range is no critical issue because the CE is never far away. Another insight is that we seek a protocol without any need of visual contact between the participants because this cannot be guaranteed. This causes an exclusion of the Infrared Data Association (IrDA) standard and a strategic focus on radio systems. A decisive factor for mobile devices is the power profile. So we are going to examine this characteristic of the different protocols. Data rate is a counterbalance to range and energy consumption. Sensor data are not big amounts of data, so we are going to mention the data throughput of the technologies, but it is not crucial. Further qualities are possible topologies and the operating frequency. The network topology has an influence on the robustness, flexibility and scalability of the network. Not all radio frequency bands are license-free and because of that we have to include the operating frequency in our decision. Our solution places the demand to involve as many developers and users as possible. Therefore, spreading and costs are characteristics we are examining as well. The context is crucial which means that we are looking for existing medical applications. Applying these criteria we decided to scrutinize the technologies Bluetooth, ZigBee and ANT and choose one protocol out of these three for our evaluation.

Bluetooth

Bluetooth is a wireless technology standard that is based on a wireless radio system [11]. It is designed to connect fixed and mobile devices and enable wireless short distance data exchange in a secure way [11]. Its original purpose was to replace RS-232 data cable connections between computer peripherals and all kind of mobile devices including their accessories such as headsets [11]. These devices operate in the Wireless Personal Area Network (WPAN) [47].

Figure 2.14: Bluetooth Logo
Source: mobilfunk-talk.de

In the early 90ies the ICT sector was looking for a technology to replace conventional short distance communication, namely cables, with wireless solutions [47]. In 1993 the IrDA was founded. The main weakness of infrared is the indispensability of intervisibility. Therefore, Bluetooth was initiated after the positive result of a feasibility study of the ICT vendor Ericsson in 1994 with the goal to develop a radio-based alternative to infrared connections. The Bluetooth Special Interest Group (SIG) was founded four years later in 1998 with Ericsson,

IBM, Nokia, Intel and Toshiba as its initial members [47]. Achievements of the IrDA standard were integrated. After Bluetooth 1.0 caused many problems the SIG introduced specification 1.1 in 2001 [25]. This version was ratified as IEEE Standard 802.15.1-2002. The most recent version is Bluetooth v4.0 that has been adopted in 2010 and combines the positive aspects of v3.0 and the new Bluetooth Low Energy (BLE) [25]. Nowadays, in the year 2012, there exist already more than 16.000 member companies from the areas of telecommunication, computing, networking, and consumer electronics.

Bluetooth uses short-wavelength radio transmissions in the Industrial, Scientific and Medical radio frequency band (ISM) from 2400-2480 MHz [11]. The operation of Bluetooth devices is license-free. The trade-off is that due to the inexistence of an regulating authority interferences with other devices that are using this frequency band can occur. In order to gain robustness and avoid such interferences Bluetooth uses frequency hopping [11]. The frequency band is split into 79 steps of frequency in an interval of 1 MHz [11]. It depends on the type of the packet how often the bands are changed, but there are at most 1600 hops a second [11]. Additionally to the frequency hopping Bluetooth encrypts sent data and provides thereby a secure way to connect and exchange information [11]. For the encryptions keys with a key length of up to 128 bits are used [11].

Class	Maximum permitted power		Range
	(mW)	(dBm)	(m)
Class 1	100	20	100
Class 2	2.5	4	10
Class 3	1	0	5

Table 2.2: Bluetooth Classes
Source: Garroppo et al. 2011 [23]

The protocol's purpose is to replace wired information exchange and connect devices by using a radio broadcast communication system with low-power consumption [11]. As can be seen in Table 2.2 there are three classes with different power demand, and the range varies dependent on the used energy. The values in Table 2.2 give an idea on the relation between power consumption and range, but there are also other influences on range, such as propagation conditions, material coverage, production sample variations, antenna configurations and battery conditions.

The data rate of Bluetooth differs between the versions, but varies in the band of MBit/s. Bluetooth 1.2 has a data rate of 1 MBit/s while Version 2.0+EDR offers up to 3 MBit/s. Bluetooth v4.0 introduced BLE, a feature that suites better for wireless devices in healthcare and fitness devices which operate in a WPAN. As the name already indicates its power consumption is lower and a lower data throughput is the trade-off. BLE uses the same frequency band like the

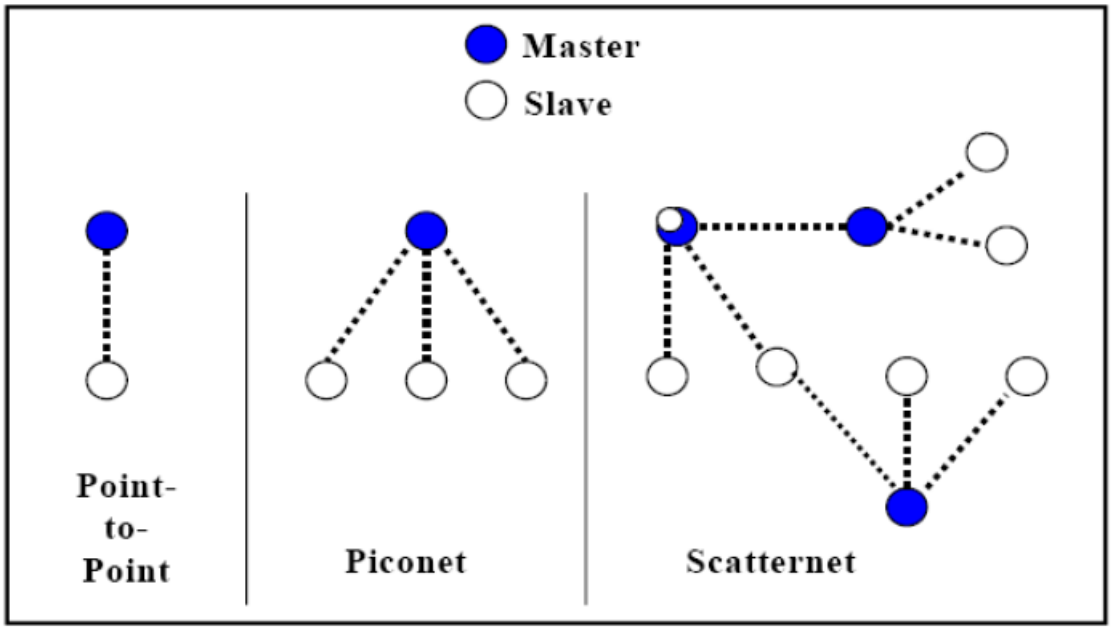


Figure 2.15: Possible Bluetooth Topologies
Source: ARS Software GMBH - Original Figure [25]

classic Bluetooth protocol, but it is not backward compatible. Even though there exist Bluetooth 4.0 dual-mode devices to bridge between these two protocol standards. In the release v3.0+HS and later Bluetooth links are used to bridge to Wi-Fi networks and high data rate traffic is carried over an 802.11 link. The benefit is a much higher connection speed up to 24 MBit/s with Bluetooth v3.0+HS. Furthermore, the data rate depends on the network topology. Figure 2.15 illustrates three possible network topologies. Apart from point-to-point connections a piconet can be used to build a network of several communication devices [47]. Up to 255 devices can be in one piconet and 8 are allowed to be active simultaneously [47]. One of the devices is the master, the others are slaves. A device can be in more than one piconet, but can be master in only one of them [47]. Several piconets combined build together a scatternet [47].

The design of the Bluetooth standard is user-centred [11]. In order to reduce the overhead of the protocol there exist specific procedures with a minimal scope of work for several use cases [11]. These procedures are called profiles and apart from profiles such as the File Transfer Protocol (FTP), the Headset Profile (HSP) or the Basic Printing Profile (BPP) there exists a Health Device Profile (HDP) that we want to discuss more in detail [11]. For a long time the Bluetooth Serial Port Profile (SSP) was very popular for health device communication, but in the year 2008 the SIG introduced the HDP to replace this interim solution [52]. For sending vital data the HDP uses the ISO/IEEE 11073-20601 Data Exchange Protocol [52]. Figure 2.16 pictures a part of the Bluetooth protocol stack with a focus on the HDP. While the lowest layer,

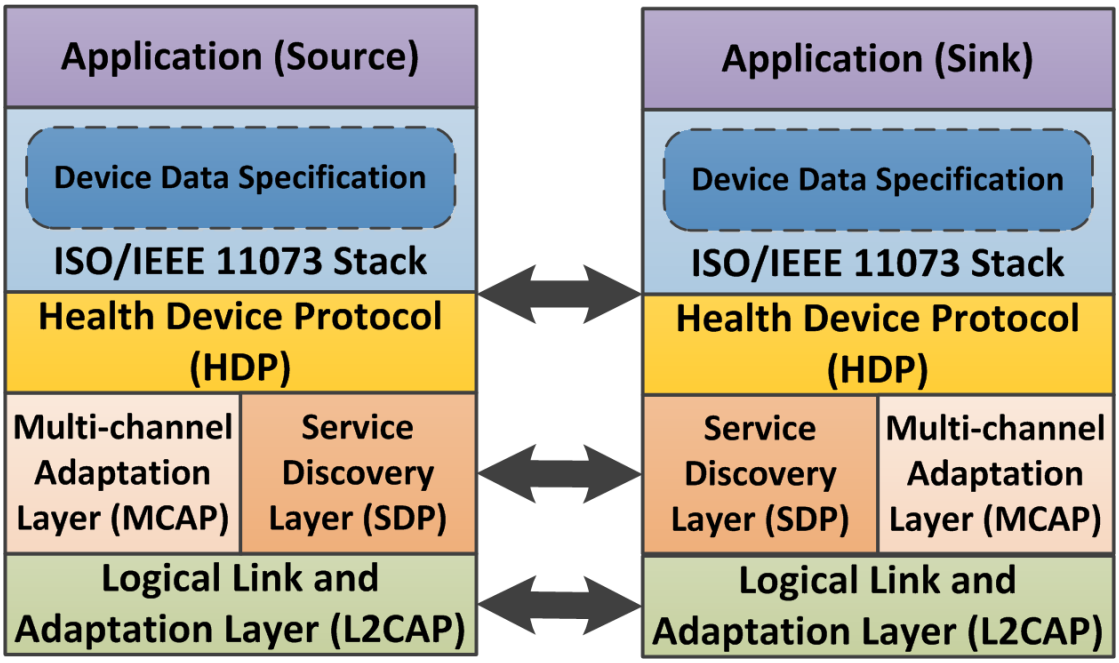


Figure 2.16: Bluetooth Stack with focus on HDP
Source: ARS Software GMBH - Modified Figure [24]

namely Logical Link and Adaptation Protocol (L2CAP), is responsible for physical tasks like timing and structure of the packages the mounted layers are more application-oriented [24]. The Service Discovery Protocol (SDP) is responsible for the registration of new services and to discover the available services on remote devices [24]. Multi-Channel Adaptation Protocol (MCAP) is capable of creating communication and data links for the exchange of generic commands and medical device data. Contrary to SDP, which is used by several Bluetooth profiles, MCAP is specific for the HDP [24]. Bluetooth knows two different roles: The source and the sink [24]. The source is a transmitter for medical data, such as PHDs are [24]. The sink receives the medical data and processes it further [24]. The HDP's purpose is to facilitate the communication between these two partners. A device can be both, source and sink. However, for a single connection the role is always assigned [24]. In Figure 2.16 the layer on top of HDP is the ISO/IEEE 11073-20601 protocol stack. We will go further into detail on this standard in Section 3.4.

ZigBee

Like Bluetooth, ZigBee is a radio network standard to bridge short distances for low-cost applications and replace cable networks [20]. Bluetooth is designed for services with moderate data rate (about 1 MBit/s), such as for voice applications of mobile phones [20]. However, ZigBee focuses on home networking [20]. The requirements for this purpose differ on the one hand in the needed data throughput. Bluetooth provides a signal rate which is much higher than required for most home networking solutions [9]. On the other hand especially the limited range of Bluetooth is a problem and has been a decisive factor for the foundation of the ZigBee Alliance in the year 2002 [20]. Motorola's initiative started in the year 1998 [9] and its first outcome, the IEEE 802.15.4 standard, was ratified in the year 2003 [9]. Since we are discussing the operation of mobile devices the energy consumption is crucial [20]. The IEEE 802.15.4 standard is a low-power networking technology that has been developed for the use of ultra-low-power applications [20], such as sensors with batteries as their power supply [20]. Bluetooth's BLE protocol stack was introduced years later in 2009 together with the specification Bluetooth v4.0.

Figure 2.17: ZigBee Logo
Source: alpwise.com

OSI Reference Model The Open Systems Interconnection (OSI) model has been ratified by the ISO in 1983 and provides an abstract view on communication protocols in computer networks [15]. The communication is separated in seven stacked layers, namely Physical, Data Link, Network, Transport, Session, Presentation and Application Layer [15], which depend on each other and divide up the functions of a communication system [15]. Each layer has a particular scope of responsibility and the data are hand through all of them step by step [15].

The IEEE 802.15.4 defines a robust radio physical layer and a media access control layer [9]. It is designed for tasks where high data throughput is not of big importance and power consumption and the costs are the most decisive requirements [20]. ZigBee is based on the IEEE 802.15.4 and builds the Network, Security and Application framework on this foundation [9]. The Media Access Control address (MAC address) layer is comparable with the data link layer of the ISO OSI Reference Model and the overlying ZigBee layers complete the stack [9]. Figure 2.18 visualizes this separation into altogether five layers and two protocols.

Since ZigBee is based on radio technology, it does not depend on a direct line of sight [20]. It has an omnidirectional radiation and the used frequencies can penetrate most materials used in homes [20]. Interferences can be evaded by a change of the frequency band. Like Bluetooth and Wi-Fi, ZigBee normally operates in the ISM radio bands and has 27 non-overlapping channels [4]. In Table 2.3 we list the frequency bands, their number of channels and jurisdiction [4]. The 2.4 GHz band offers 16 channels and is apart from some exceptions valid worldwide [4]. In Northern America, Australia and a few other countries the 915 MHz band can be used while in

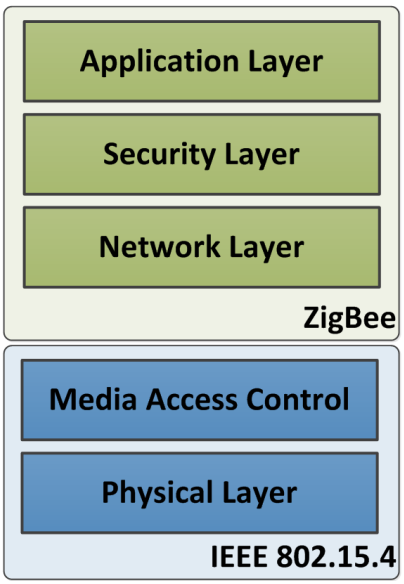


Figure 2.18: ZigBee Stack
Source: Self-made Figure

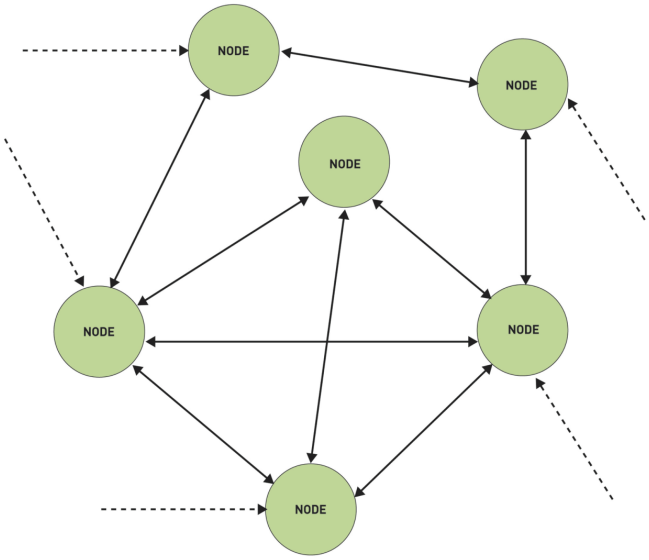


Figure 2.19: Mesh Network Topology
Source: ANT Alliance - Original Figure [5]

Frequency Band	Number of Channels	Jurisdiction
2.4 GHz	16	Global
915 MHz	10	Northern America, Australia
868 MHz	1	EU Countries

Table 2.3: Radio Frequency Bands and Channels of ZigBee
Source: ZigBee Alliance White Paper [4]

EU countries 868 MHz are a possibility [4]. The corresponding data rates are between 20 and 900 KBit/s [20].

ZigBee is configured as a mesh networking technology [4]. A mesh network is a type of ad hoc network where every node has routing responsibility without an inherent hierarchical structure. This is illustrated in Figure 2.19. As a reaction to broken or blocked paths self-healing algorithms reconfigurate the network if required. Home networks benefit from these dynamic characteristics since they improve their reliability [9]. The end nodes are mobile within the network due to the self-organizing and self-healing [9]. They can be extended to a maximum of 65.000 devices per network [9].

Several application profiles are provided by ZigBee. For healthcare applications there exists the Personal, Home and Hospital Care (PHHC) profile [3]. It is based on the ISO/IEEE 11073-20601 standard for an optimized exchange protocol for vital data [3]. ZigBee Health Care fully supports the ISO/IEEE 11073 for point-of-care medical device communication. There already exist a number of PHDs, such as blood pressure monitors, body scales and glucose meters using ZigBee [3].

ANT and ANT+

ANT is a proprietary wireless networking protocol designed for low-power and low-cost WSN technologies [39]. It operates like Bluetooth and ZigBee on the 2.4 GHz ISM band [5]. Initially the protocol was developed to make a connection between Nike's foot pods and watches [5]. The Triax Elite foot pod by Dynastream Innovations was designed in the year 2000 to measure speed and distance while running [5] and theory became practical in 2004 in a cooperation with Nordic Semiconductor [5]. Other members of the ANT Alliance are the GPS equipment manufacturers Garmin (since December 2006 the parent company of Dynastream) and Magellan, the manufacturer of semiconductors Texas Instruments, Sony Mobile Communications, many sports equipment manufacturers like Nike or Geonaute (Decathlon Group) or the Austrian AAL and e-Health company Spantec [5].

Figure 2.20: ANT+ Logo
Source: cnet.se

Fundamentally, the ANT protocol is for general purpose, but ANT+ is an interoperability function that can be added to the base protocol and its characteristics are principally suited well for sports, fitness, wellness and home health applications [5]. ANT+ facilitates the collection, automatic transfer and tracking of sensor and actuator data for monitoring of all kind of personal wellness information in a BAN or WPAN. It targets the ultra-low-power wireless device's market and needs an average current of a few micro amps which allows operation with button cells [5]. An additional feature of ANT is the optimized code space. A heart rate monitor specific ANT code fits in a 2k code space [5]. The bit rate of 1 MBit/s is high for sensor data exchange [39]. On the other hand the message rate is low (4, 8, 16 messages/s) [39].

An ANT-enabled device consists of a Microcontroller Unit (MCU) hosting an ANT module or chip as a black box [5]. This chip is pre-loaded with the protocol and is controlled by an application processor [5]. USB is a possible alternative [5]. The ANT communication sessions are established and maintained by this application host MCU [34]. Figure 2.21 illustrates the relationship between ANT's protocol stack and the ISO OSI Reference Model [34]. Physical, Data Link, Network and Transport Layer are implemented by ANT [34]. Additionally, some low-level security methods are the foundation for more complex ones. These have to be implemented by the developers of new ANT solutions [34]. The protocol is aimed to be effective. Therefore, such features are outsourced to upper layers and only used if needed [34].

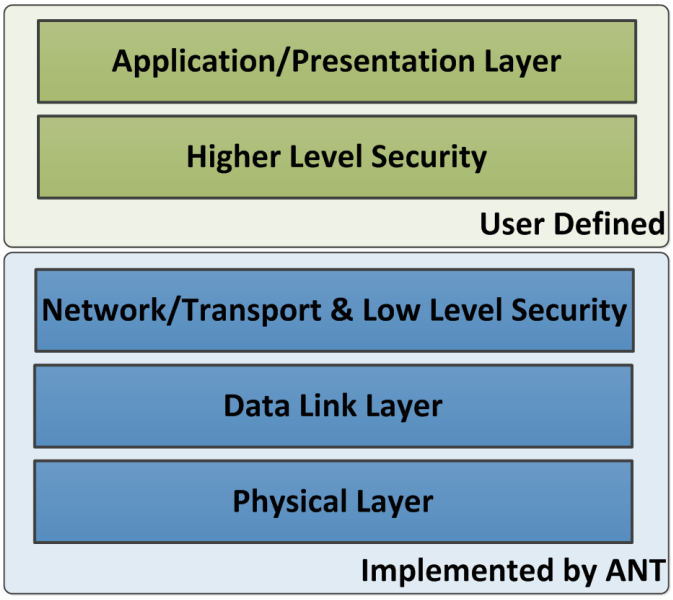


Figure 2.21: OSI Layer model of ANT
Source: ANT Alliance - Modified Figure [34]

There is no specific network topology ANT is configured for. As illustrated in Figure 2.22 it supports a large range of scalable topologies from broadcast over a star down to complex ones like a multi-transceiver system with full point-to-multipoint communication capabilities [34]. For complex topologies no coordinator is required [34]. ANT's communication is channel-based [5]. Their number is restricted to 8 per device at a time and there is always at least one slave and one master [34]. If the channel is shared this number can be increased up to 65533 nodes [5]. Doing so the devices only send data when their address is pinged by the hub [34]. This can be seen in the second row of Figure 2.22 on the left. The maximum number of nodes in a network depends on the address range which is 32 bit for ANT on a specified network key [5].

In the previous subsections we discussed the different strategies to evade interferences and so we do here since we have the same issues in the same radio frequency band. ANT uses a system called frequency agility [34]. Similar to frequency hopping, channels are changed to prevent interferences [34]. However, the ANT protocol carries out these changes only on-demand and if a significant degradation is observed [34]. There exists a background scanning channel that performs continuous search operations in order to find open channels [34]. Every device can do such a search like it is sketched in Figure 2.22, but it is recommended that every device has at maximum one such channel open at a time [34].

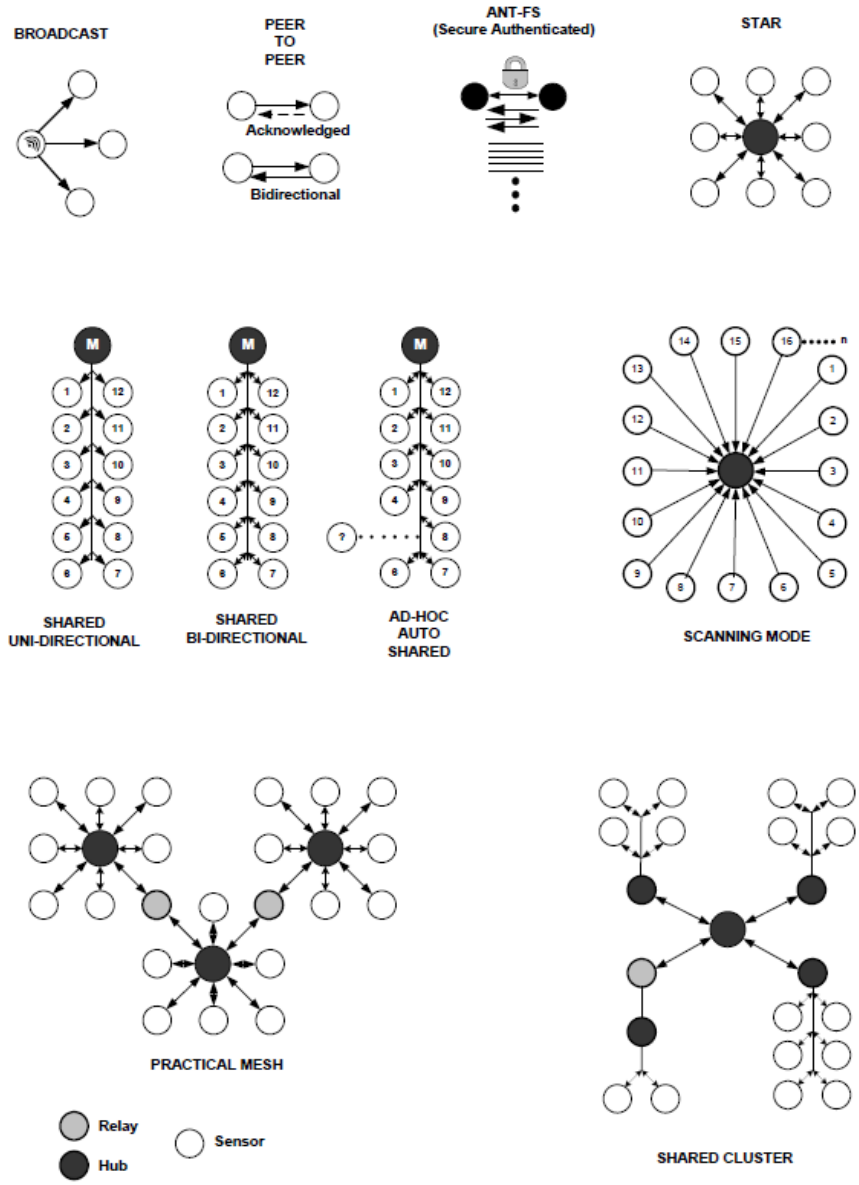


Figure 2.22: Example ANT networks
Source: ANT Alliance - Original Figure [34]

ANT is a proprietary standard that focuses on ultra-low-power wireless networking applications like WSNs. It uses the ISM band and avoids interferences by frequency agility. The data rate is lower than Bluetooth's throughput, but higher than the one of ZigBee. Current ANT+ profiles are e.g. Heart Rate Monitor, Bicycle Power or Body Scale [5]. There exist hardly any CEs with preinstalled ANT-enabled hardware. Most Sony Xperia Smartphones and the HTC Rhyme provide ANT connectivity [5], but usually the CE has to be expanded with an ANT transceiver.

Comparative Study and Conclusion

Bluetooth over the IEEE 802.15.1, ZigBee over the IEEE 802.15.4 and ANT+ are three wireless transport protocols that are suitable for communication between a CE and a sensor or actuator. They are all designed to avoid cable connections and transmit data with a short distance action range with low power consumption (especially for mobile, battery-powered devices). Bluetooth targets on manufacturers of auxiliary equipment and input devices for mobile phones and computers including hands-free headsets, cordless keyboards, or printers. ZigBee's speciality is reliable wireless networked monitoring and to control networks. This protocol is used for home networking applications with periodic or intermittent data transmission and relatively low data rates, such as wireless light switches, wireless thermostats or all kind of remote controls. The third analysed standard, ANT+, is an interoperability function to the base ANT protocol [5]. It is focused on diagnostic applications for sports (bike computers) and the healthcare sector (heart rate monitors) [5].

Table 2.4 outlines the specific characteristics of the wireless transport protocols Bluetooth (classic Bluetooth and v4.0's feature BLE), Zigbee and ANT and gives us the foundation to compare them to each other. Because of the lack of an already existing comparative study including all of the chosen protocols, we considered a several studies which discussed subsets of our selection. Due to the variety of protocol versions we have to be careful when talking about exact numbers. Besides we want to declare that the source of ANT's characteristics is in most instances information announced by the ANT Alliance itself and not by an independent author. The development in this area is fast and it is crucial to be honest and realise that the numbers in a comparative study concerning transport protocols are not to be taken as accurate but more to understand the magnitude. We focus on the applications, the generic strengths and weaknesses of the protocols, because these are stable reasons for decisions. Nevertheless, these numbers are a useful snap-shot and help to put the idea across.

Previously each protocol was discussed separately. In this subsection we oppose them to each other. We compare the output of three initiatives, namely the Bluetooth SIG, the ZigBee Alliance and the ANT Alliance. Bluetooth and ZigBee base on IEEE specifications [42] while the ANT protocol is proprietary [5]. Since the output of this thesis is open source this is a major drawback for ANT. We want to attract the biggest possible community and do not want to focus on the members of a single strategic alliance of manufacturers. The frequency band is the one attribute where there is hardly a difference. ZigBee offers additional frequency bands conditioned by the location [42]. However, this becomes relevant when talking about interferences and the protocol specific mechanisms to facilitate coexistence with other applications on the ISM frequency band. Studies demonstrate the big effect of Wi-Fi on the signal quality [23]. ZigBee's data rate drops 41% on average and Bluetooth performed even worse with a loss up to 68% [23]. Furthermore, ZigBee is less vulnerable by the position of the Wi-Fi transmitter than Bluetooth is [23]. This influences the necessary distance and separation to get a connection working under normal conditions [23]. Wi-Fi is the biggest problem for the WPAN [23], since there are numerous WLANs in an increasing number of buildings. Furthermore, the many public area (e.g. squares and parks) are connected to the internet using Wi-Fi. The results for

Standard	Bluetooth	Bluetooth Low-Energy	ZigBee	ANT
Certification Body	Bluetooth SIG	Bluetooth SIG	ZigBee Alliance	ANT Alliance
IEEE spec.	802.15.1	802.15.1	802.15.4	Proprietary
Frequency Band	2.4 GHz	2.4 GHz	868/915 MHz; 2.4 GHz	2.4 GHz
Number of RF Channels	79	40	1/10; 16	125
Coexistence mechanism	Adaptive Frequency Hopping	Adaptive Frequency Hopping	Dynamic Frequency Selection	Frequency Agility
Max. Signal Rate	1-3 MBit/s	1 MBit/s	250 KBit/s	1 Mbit/s
Nominal Range	10-100 m	40 m	10-100 m	30 m
Network Topology	Piconet, Scatternet	Star-Bus	Peer-to-peer, star, tree, mesh	Peer-to-peer, star, tree, mesh
Max. number of cell nodes	8 active / 255 passive	8 active / 255 passive	65.000	65.533
Encryption	56/128-Bit E0	128-Bit AES	128-Bit AES	128-Bit AES
Data Protection	16-Bit CRC	24-Bit CRC	16-Bit CRC	16-Bit CRC
Data Coding Efficiency	94.41	n/a	76.52	47
Battery Life	days	months & years	months & years	years
Developer Community	big	-	medium	insignificant

Table 2.4: Comparison of classic Bluetooth, BLE, ZigBee and ANT
Source: Several Datasheets and Papers - Self-made Table [42] [35] [48] [59] [34] [5]

coexistence of Bluetooth and ZigBee is better and two of such networks can work next to each other with almost no drop of performance [23]. ANT is not included in this comparative study so we have to rely on ANT Alliance's announcement that the protocol is robust and can coexist with WiFi/BlueTooth/GSM/3G [5]. They claim that the Message Error Rate is low enough for distortion-free reception [5].

The highest throughput has the classic Bluetooth protocol. On the other side of the spectrum is ZigBee with a data rate of only 250 KBit/s. However, even this is sufficient, as can be seen in Table 2.5. Mulyadi et al. determined vital signs of several medical devices and their respective data rate [48]. These rates for a blood pressure monitor is insignificantly low and for a stethoscope it is 40 KBit/s. In relation to the data transfer rate of ZigBee the entire set of medical devices occupies less than a fifth of its maximal signal rate. Therefore, this criterion is obsolete.

Medical Device	Data Rate (KBit/s)
ECG	2
Stethoscope	40
Blood Pressure Monitor	0.02
SPO2	0.02
Total	42.04

Table 2.5: Vital signs of several medical devices and their data rate
Source: Mulyadi et al. in 2009 [48]

In the first part of this subsection we discussed which requirements have to be fulfilled. The nominal range of all protocols meets the targeted minimum range of 10 meters. For Bluetooth it depends on the class. Class 1 and Class 2 facilitate a communication over more than the placed demand [23]. BLE has a range of 40 meters [35] while ZigBee connects devices over up to 100 meters [42]. ANT claims to have a nominal range of 30 meters, which is the shortest of these four, but still enough for our application [5].

Ciphers In cryptography the algorithm used for the encryption of the information is called cipher. There exist two main types: Block Ciphers and Stream Ciphers. A Block Cipher separates the data into parts of equal length, so called blocks. Every block is encrypted separately. Stream Ciphers encrypt bit by bit or character-serially. This way of proceeding saves transmission capacity and is favourable for streaming of multimedia data. The drawback is the quality of the encryption itself. In general Block Ciphers are more secure [73].

Bluetooth is designed for smaller networks than the other protocols and this causes on the one hand a little maximum number of nodes and on the other hand it makes complex topologies impossible [42]. ZigBee [42] and ANT [34] have a similar nature and make the same offer to the user. Umpteen thousand cell nodes can be connected using a great variety of topologies.

All of the four wireless transfer protocols use techniques to facilitate encryption and authentication. The classic Bluetooth protocol encrypts its messages with a 56/128-Bit E0

stream cipher [42]. Its brother BLE [35] and the other two protocols in this comparative study, namely ANT [5] and ZigBee [42], use a 128-Bit Advanced Encryption Standard (AES) block cipher. The AES is the state of the art-block cipher and is approved by the National Security Agency (NSA) for top secret information. For the authentication classic Bluetooth, ZigBee and ANT use 16-Bit Cyclic Redundancy Check (CRC) and BLE utilizes 24-Bit CRC [42].

This use of a Stream Cipher for encryption has positive effects on the data coding efficiency, which is the ratio of the data size and the size of its content, the message itself. Bluetooth performs well due to its insignificant overhead. ZigBee has a good efficiency for tiny messages (less than 102 bytes) [42], but for growing data size Bluetooth has a much better efficiency of over 94% [42]. We have no comparable numbers for BLE, because of the actuality of the Bluetooth 4.0 standard. However, classic Bluetooth is almost 20 percentage points better than Zigbee [42] and twice as efficient as ANT is.

Regarding battery life there was no detailed comparative study available to base our statements on. Therefore, we only want to give a vague magnitude in order to provide an overview. Devices using classic Bluetooth will last only for a couple of days if the power supply are only button cell-type batteries and the device is turned on without any break [48]. That was the reason for the development of BLE which performs at eye level with ZigBee [59]. ANT claims on its website that it has the lowest power consumption and its batteries last for several years [5]. Moreover, ANT Alliance announced that their protocol has four times lower power costs than ZigBee [5].

We want to establish our interface as a proof of concept for developers. Therefore, it is crucial to use technology with already wide spreading and a big community of developers (preferred open source). Bluetooth has the biggest community. Furthermore, Bluetooth profiles exist for many applications, e.g. the HDP which is even implemented in Android 4 ICS. Due to its great variety it is popular in the field of mobile phones. Furthermore, the Continua Health Alliance had to decide in 2009 which technology it wants to push ahead as its radio technology selection for its Version 2 guidelines [50] - Sensium, ZigBee, ANT+, BLE, BodyLAN, or Z-Wave [50]. Contiua decided to use Bluetooth as its flagship transport protocol in version 1 and BLE and ZigBee for its version 2 guidelines. Therefore, this influenced the market [50] and in August 2012 there are 38 Continua-certified products using Bluetooth, one ZigBee product and not a single one using BLE. Bluetooth is designed for fast data exchange between devices in a WPAN, but its stack is four times bigger than the one of ZigBee and 100 times larger than ANT's stack [5]. The power consumption is 10 times higher and the hardware costs are higher as well [5]. Even BLE cannot compete with its bare characteristics due to its slightly higher costs and higher power consumption. ZigBee and ANT have many benefits and are superior especially for small battery-powered devices [48] which operate in a WSN or BAN. These protocols have advantages in their low power consumption, their simplicity and their ability to cover a large number of devices per network [48]. They have lower data rates, but this does not matter in view of the fact that vital information is no big data [48]. These two protocols would perform well for the interface that is directly connected to a PHD [48]. ZigBee is popular for medical devices [48]

and there are Continua-certified products on the market. Nevertheless, the majority of these devices, and especially the first generation of Continua-certified PHDs, use Bluetooth [48] because it is very common for mobile phones, PDAs and laptops [48]. This makes it a lot easier for end users and developers to use it out of the box. The popularity and the availability of evaluation devices are the reasons for the focus on Bluetooth and the HDP in this thesis. It is the most popular wireless transport protocol, but the Bluetooth stack and the profile's implementation is on a different level of development for the different operating systems [21]. BlueZ is an open source Bluetooth stack and is the most popular one in the developers' community [75]. It works for Linux as well as for Android.

2.3 ISO/IEEE 11073 Manager Implementations

Tracking the development of the market for PHDs it is easy to identify the trend regarding their increasing availability [31]. The devices are affordable for usual households and their accuracy is improving [31]. One issue of these devices still is the interoperability as we mentioned above in Section 1.3. We are looking for an open standard approach to allow AAL systems to easily extend to third party providers and exchange vital data with devices customary in the trade. Since the situation and this demand is nothing secret nor a totally new trend we are not the first ones discussing this topic. In this section we are going to discuss other approaches to this topic. Our main focus of attention will be put on the inner structure of the developments and on the kind of publication. The section is directed towards learning for our implementation by analysing the approaches of the different ISO/IEEE 11073 manager solutions and search for reusable libraries.

OXPlib and Proprietary Software in General

The OXPlib is a commercial library written in C++ and Java that supports the operation of manager components following the Optimized Exchange Protocol (ISO/IEEE 11073-20601) [33]. The implementation provides an API for the development of Continua Health Alliance's PAN client component [33]. OXPlib is Continua-certified for both, Windows and Linux systems, and claims that its use eases the certification for customer's products [33]. OXPlib was developed by Lamprey Networks Inc. (LIN) in the year 2009 and is derived from the software that was originally written for the Continua Health Alliance [33]. This original software is available for Continua Members, but is not open source [33].

Figure 2.23: LNI Logo
Source: lampreynetworks.com

Since it is our goal to facilitate communication among PHDs and CEs in an open approach this library and other solutions like the ones by Stollmann [63] or Wipro [65] are not of interest. We cannot make use of them and cannot compare them to our interface. Hence we are not going further into detail on them.

STEPSTONE

Project STEPSTONE is a joint industry-university project and started as a collaborative effort between the IBM and the Mobile and Pervasive Computing laboratory at the University of Florida [30]. It was created in 2009 and was published under the terms of the Eclipse Public License [66]. Later the work was continued by several other contributors while hosted at the Eclipse Foundation [30]. Initially it was launched to provide a reference implementation for device-driven patient monitoring services at the university's *Gator Tech Smart House* laboratory [66]. The *Gator Tech Smart House* is an experimental laboratory and an actual live-in trial environment for validating technologies and systems developed by the university [66]. Its goal was to enable scalable wireless integration of commercial off-the-shelf healthcare devices into enterprise systems in an open standard approach and using a SOA [31]. Since August 2010 the project is hosted at Open Health Tools.

At the Integrating the Healthcare Enterprise (IHE) Europe event STEPSTONE demonstrated its usability by inputting live vital data from PHDs into a network of hospitals [66]. On its way to the hospital the data pass the following stations and document types [66]:

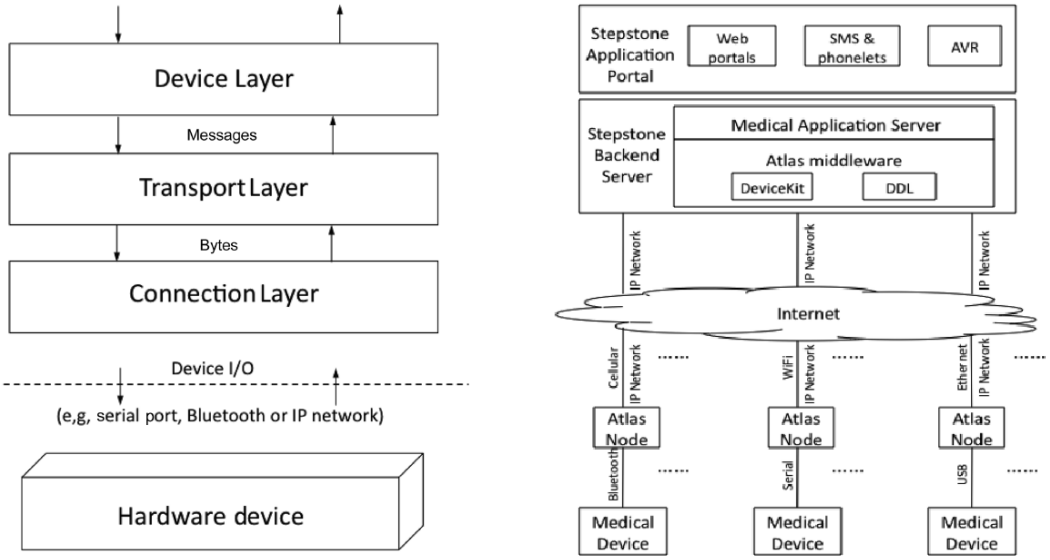
HL7 Personal Healthcare Monitoring Report (PHMR) The non-profit organization Health Level 7 (HL7) develops and introduces open interoperability standards for electronic health information. It is an American approach with international members and global influence. The HL7 framework is commonly used for transfer, integration, sharing, and retrieval of all kind of health-related data. The PHMR document contains analysed and raw information of data generated by PHDs.

Medical Device Encoding Rules (MDER) These rules are used to encode the data objects of the Domain Information Model (DIM) into a binary message. This transformation is necessary in order to transfer the data via the communication model which we will describe in Section 3.4.

1. Medical Device: ISO/IEEE 11073-20601 MDER encoding
2. STEPSTONE PHD Manager: ISO/IEEE 11073-20601 XER encoding
3. STEPSTONE Backend: HL7 PHMR document
4. IHE Registry/Repository

Figure 2.24 (b) displays the inner system architecture of STEPSTONE. The PHDs send their measured vital data to the *ATLAS Node* [31]. This node connects the heterogeneous devices with their different interfaces to access services [31]. From there on the data are passed on to a LAN or Wide Area Network (WAN) (e.g. the internet) and the middleware in the backend processes the information [31].

The STEPSTONE project is an implementation build on Eclipse Service Oriented Device Architecture (SODA) to be a stepping stone for other developers of health and wellness solutions in an similar environment [66]. Its purpose is to exchange data with devices and, furthermore,



(a) Device Kit Abstraction Layers

(b) System Architecture

Figure 2.24: STEPSTONE
Source: Helal et al. - Original Figure [31]

it offers a well-defined API [66] and a rich server platform web application which is based on the OSGi framework Eclipse Equinox [66]. The use of OSGi and Java facilitates dynamic interaction with medical devices [66]. The *STEPSTONE PHD Manager* is an implementation of the ISO/IEEE 11073 Personal Health Device Standards (11073-20601 and selected device specializations 11073-104xx) and its core class represent the ASN.1 models (further information about ISO/IEEE 11073 details follow in Chapter 3).

STEPSTONE is divided into two separate applications, the edge and the backend [66]. The backend is the manager software sticking to the vocabulary of this thesis [66]. It receives the vital data from the edge, which runs on the device itself and serves there as a gateway [66]. For reasons of modularity the drivers are implemented in a device adapter framework and their adapters shared out between three distinct software service layers that are depicted in Figure 2.24 (a) [31]:

- *Connection Layer*: This layer is the connection interface for the transport protocols. It establishes connections to the hardware device, reads in byte streams and publishes data. The STEPSTONE Device Kit has native support to Bluetooth, Internet Protocol (IP), USB, and RS-232 Serial Port.
- *Transport Layer*: Wraps up the data stream as messages or the other way round. It is the interface between the device and the *Protocol Adapter* component of the SODA framework.

- *Device Layer*: Interprets the messages semantically.

Due to this modular architecture the pre-built adapters can be replaced with self-generated ones [66]. Eclipse tooling supports such tasks [66]. The OSGi framework provides a high degree of flexibility and facilitates easy exchange of single modules, namely bundles.

IHE XDR/XDS The IHE is an initiative of healthcare professionals to improve interoperability of healthcare IT systems. XDR/XDS are profiles for secure cross-enterprise document sharing.

The backend uses a dynamic application framework like the edge, and provides a web-based view on the data [66]. After familiarizing reading the information in it converts the messages into HL7 PHMR documents which can be passed on via IHE XDR/XDS later on [66].

LibreSoft OpenHealth

OpenHealth is a Spanish research project at Grupo de Sistemas y Comunicaciones (GSyC)/LibreSoft group at the Rey Juan Carlos University in Madrid [43]. Formerly also known as Morfeo OpenHealth, the project develops eHealth solutions involving mobile devices and its goal is to deploy a full socio-technical platform to enable plug-and-play integration of AAL services for patients at home [29]. The aim is to develop a manager-agent system to facilitate open standard connections between PHDs and mobile CEs in BANs [29]. The mission is [43]:

Figure 2.25: LibreSoft Logo
Source: openhealth.libresoft.es [43]

- The development of a Free/Libre Open Source Software (FLOSS) implementation according to the ISO/IEEE 11073-20601 standard.
- To promote the FLOSS as a new way to understand technological solutions related to the eHealth area.

The PHD is the agent in this context and the CE is an Android-based mobile device that runs the manager service. This piece of software has the purpose to collect the vital measurement the medical devices send. Within the BAN Bluetooth and its HDP is the transport protocol in use [43]. For the independence of the platform Java is used and to meet the requirements of security of sensitive data the development is FLOSS [43].

The project is still ongoing and they already managed to reach parts of their goals [43]. The LibreSoft group developed Bluetooth MCAP/HDP for BlueZ [43]. This implementation

was integrated into the official Linux Bluetooth protocol stack since BlueZ 4.77 [43]. Today in 2012 BlueZ is the most popular stack for applications that use HDP [43]. Furthermore, this stack is officially available for Android ICS 4.0.3 [43]. The OpenHealth project has developed a manager with an ISO/IEEE 11073-20601 standard implementation for PHDs [43]. Moreover, they integrated this manager and the HDP/MCAP API for Android ICS 4.0.3 and Bluetooth and at the moment they are developing a solution for BLE [43].

Signove HDPy and Antidote

A beneficiary of Libresoft's contribution is Signove, a software developing company from Brazil with the focus on the area of pervasive computing [60]. In the year 2010 Signove joined the Bluetooth SIG and the Continua Health Alliance which specified the use of the ISO/IEEE 11073 standards for device communication [60].

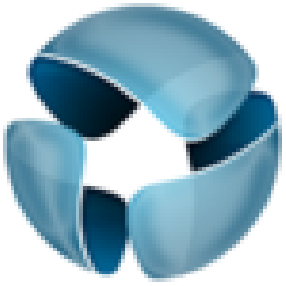


Figure 2.26: Signove Logo
Source: twitter.com

Furthermore, they released HDPy in 2010, a HDP/MCAP implementation in Python based on BlueZ [60]. In this project they aimed on first experiences how to implement a testing and prototyping program for MCAP and HDP Bluetooth protocols [60].

In 2011 Signove released the first candidate of the follow-up project Antidote [60]. Antidote is an open source ISO/IEEE 11073 stack that enables communication with Continua-certified (ISO/IEEE 11073-20601) PHDs [6]. It is a modular and standalone part of the SigHealth Platform, a platform for remote patient health monitoring and data management [18]. The non-functional requirements are portability, simple usage and simple integration in client applications [6]. Antidote can be used out-of-the-box for BlueZ Bluetooth stack (>= 4.01) and Linux or Android ICS 4.0.3 as its Operating System (OS) [6]. However, the architecture, platform and transport dependencies are out-sourced to transport plug-ins with a well-defined API [6]. They can be replaced by other separately developed ones. Hence, Antidote's core implementation is architecture-neutral [6]. The Antidote IEEE 11073 stack library consists of following components [6]:

- Core Stack: The ISO/IEEE 11073 core stack is written in C language. It implements MDER encoding for Abstract Syntax Notation One (ASN.1), selected PHD types, the DIM, an Agent and a Manager. Further details about the ISO/IEEE 11073 and descriptions for MDER and ASN.1 will be explained in Chapter 3.

- Specializations: The stack uses separate compile-time plug-ins for the specific devices as its specializations. Some sample devices, such as blood pressure monitor or glucose meter, are already implemented. If there is need for other devices than covered by the given specifications, it is possible to add a new plug-in.

- Sample transport plug-ins: Antidote is transport-agnostic and the interface to a given transport protocol is realized as a plug-in, like the specifications are. Already included are plug-ins for the use of Transmission Control Protocol/Internet Protocol (TCP/IP) and Bluetooth HDP.

Agnostic (computing) A software component that is unaware or non-committal regarding the specific nature of the components with which it interacts is agnostic [74]. To be agnostic in some context describes the characteristic of the component that it is modular and has no specific dependency.

D-Bus The Desktop Bus (D-Bus) is an Inter-Process Communication (IPC) open-source system for software applications to exchange information among each other [73]. An implementation of D-Bus supports most OS and programming languages [73]. In Ubuntu it is possible to use the freedesktop.org project for operating.

TCP/IP The TCP/IP is a family of network protocols that is popular for WAN connections and most famous because of its importance for the internet [73]. The primary reason why it is of importance in this context for the Antidote library is because it may be used to connect Continua testing tools [6].

- Health D-Bus Service: The communication between the library and the program that makes use of it as kind of its client is based on the D-Bus. This realization has the benefit that a direct link to the library is not necessary, but only an access to the D-Bus. This component of the library exports the received vital data as a D-Bus service. As a result an agent, written in any major programming language, can readout this data and process it.

In this thesis we decided to make use of the Antidote IEEE 11073 stack library which is the foundation of our ISO/IEEE 11073 interface on the CE side. Antidote is available under the GNU Lesser General Public License (LGPL) which grants a lot of permissions [6]. Additionally to the library itself the project provides a sample health service application written in Python [6]. Since we need an OSGi bundle we use the Java Native Interface (JNI) and develop the agent that receives the vital data from the D-Bus in Java.

CHAPTER 3

Concept

EVALUATION of other people's work is where every project has to start. Applying this gained knowledge and contribution to the community is evidently the next step. We already know what this thesis is aiming for and in this chapter we go further and develop a concept how to implement the interface.

In the first section the UniversAAL project will be introduced. We announced this project previously, but here we go deeper into detail and explain its architecture and reveal technical details. While it is possible to apply our interface to other AAL platforms, UniversAAL is being used for evaluation purposes in this thesis.

As we aim for widespread adoption already existing open solutions and architecture decisions of other players in AAL are supporting our approach. The Continua Health Alliance has the goal to increase interoperability of AAL products with the use of existing standards. Moreover, the ISO/IEEE 11073 standard is Continua's main protocol stack [22]. Although only few products use this norm currently, this trend is changing and an increasing number of Continua-certified devices have been introduced onto the market in the recent past. The ISO/IEEE 11073 enjoys widespread acceptance and there exist no alternatives if interoperability is the goal. It is a suitable standard for our purpose and that is the reason why we focus on this standard. We will describe the architecture and shine light on its historical development.

After clarifying the details we are ready to present the architecture of our implementation and explain the underlying concept and the used technologies. The following sections will explain more technical details and even while maintaining generic layer of abstraction we sacrifice the level of abstraction for the sake of technical details.

3.1 The Concrete Architecture of UniversAAL

We mentioned above the architecture of UniversAAL in general and got acquainted with its three layers. The *Reference Model* summarizes a common understanding of AAL and the *Concrete Architecture* provides a common approach to software development [69]. The *Concrete Architecture* is a specific software solution and this section will be focused on this layer and we will investigate UniversAAL's inner structure.

Consolidation and Instantiation

UniversAAL taps the results of former projects and existing solutions. Input projects are AMIGO, GENESYS, MPOWER, OASIS, PERSONA and SOPRANO. Figure 3.1 illustrates the analysis of their reference models. The first column depicts the Description of Work (DoW) of the UniversAAL project and the other columns contrast this partitioning to the one in other projects.

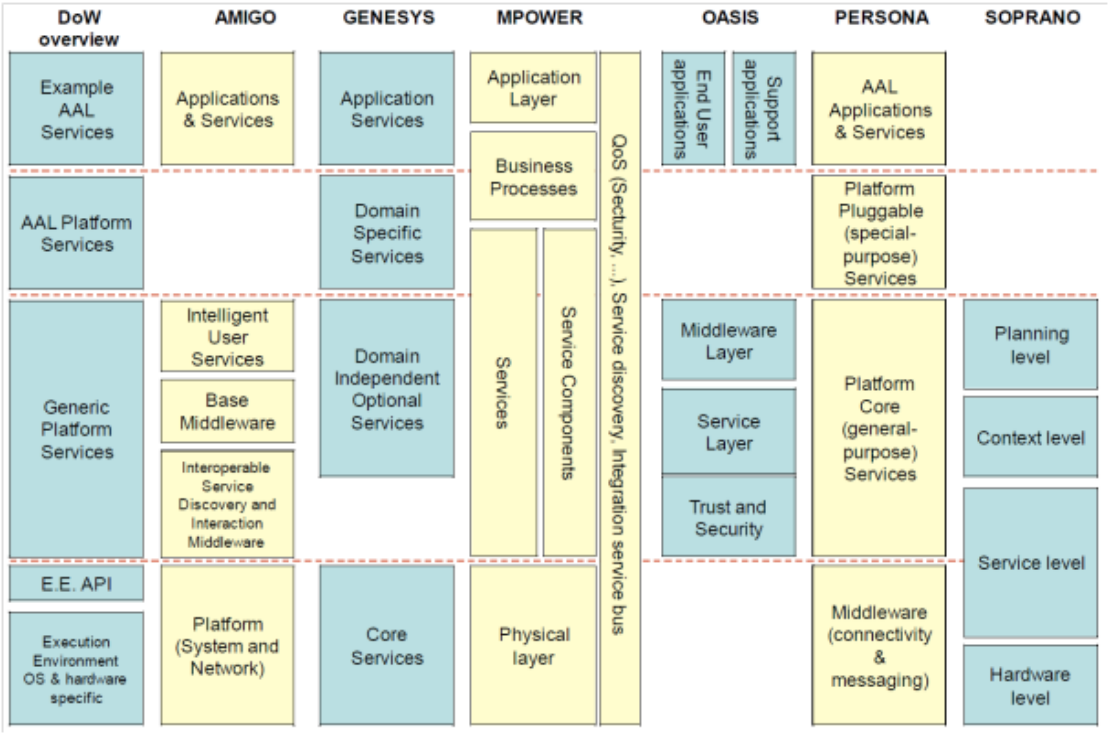


Figure 3.1: Consolidation of layer models
Source: UniversAAL - Original Figure [69]

Even though none of these projects had a reference model, there are overlaps in the concepts of the different projects and these patterns were used to derive a set of categories [69]. Furthermore, terms from the input projects were mapped to these categories to gain a basic

understanding of them. At this point of the design phase representatives of the input projects depicted their own work following the guideline of using the categories as a common terminology. Starting from here discussions lead to the UniversAAL DoW [69]. In Figure 3.1 the result of this consolidation process is illustrated. The figure reveals that the model of GENESYS and PERSONA are very similar to UniversAAL DoW. As the result was very compact no further consolidation steps have been taken inside each category separately [69]. The DoW provided a basis for subsequent work and the analogies to refine of development within UniversAAL in Figure 2.11 can be seen easily. The runtime support compared to the DoW is identical. These similarities also result in reused code of PERSONA in the UniversAAL project.

The reference architecture interacts with specific AAL solutions and technologies in two distinct ways [69]:

1. Consolidation: Empirical data are extracted out of experiences from real-world implementations of AAL solutions. Subsequently this data are used to design a valid reference architecture.

2. Instantiation: New AAL solutions are created based on the reference architecture.

Hiding Distribution and Heterogeneity

Middleware A distributed system consists of several autonomous computers that interact with each other working together for the purpose of accomplishing a common goal. The software that facilitates the communication among the participating nodes is called middleware. In order to connect different computer systems, it has to hide the distribution of the system on the one hand and any possible heterogeneity of its nodes on the other hand. The middleware is the connecting gateway between the different computer systems and the applications run on the top of it. In contrast to application software it has no specific task that helps the user. However, it enables applications to fulfil their purpose by providing a proper environment for them.

Previously, we gave a short impression of the consolidation process and in this subsection the design of the *Concrete Architecture* of the UniversAAL platform will be investigated. Since it is a stand-alone AAL solution it can be assessed as the first instantiation of the *Reference Architecture*. Therefore, some software artefacts of the PERSONA project [45] are reused [69].

The principal motivation for consolidation and development of a layer model for UniversAAL's runtime platform was to provide a basis for comparison of the input projects. In Figure 2.11 and 3.1 the two lowest layers have the same labels with different subtexts. In order to gain more consistency value is added to the layer *Execution Environment: Common API*. More than this common API a mediating software is desired and consequently the layer is renamed to *AAL Node Middleware* [69].

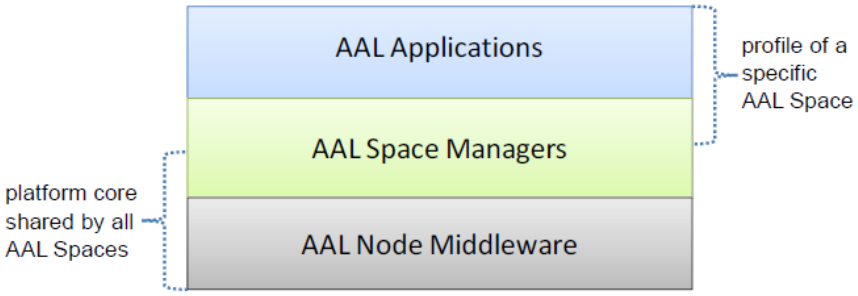
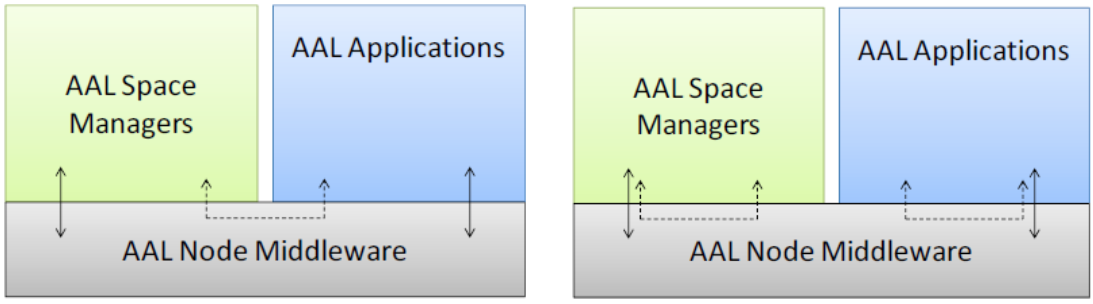


Figure 3.2: The consolidation of layer model for the runtime environment in AAL Spaces
Source: UniversAAL - Original Figure [69]

Furthermore, the labels *Generic Platform Services* and *AAL Platform Services* cause confusion and the inherent purpose of these layers is to guarantee that the network of AAL-ready nodes provide a coherent view on the AAL space [69]. Therefore, both components are merged into one new layer named *AAL Space Manager* [69]. Figure 3.2 depicts the modified layer model for the runtime environment. The nodes operate with a high degree of freedom in dealing with distribution and without any kind of protocol stack being replicated on all of them [69]. Nevertheless, a basic footprint of the middleware layer has to be present [69]. Nodes are AAL-ready if they are networking-enabled and run the middleware or parts of it [69]. Since we have the middleware, deployers do not need to know anything about the internal structure and the underlying logical layers. All requests are handled by the middleware and in this way UniversAAL releases developers and deployers from the burden of maintaining an interface to other *AAL Space Managers* or *AAL Applications* [69]. Another benefit this solution provides is a high degree of flexibility and extensibility of the system in all layers, because components of the platform are easily replaceable [69].



(a) Inter-Layer Communication (b) Indirect Intra-Layer Communication

Figure 3.3: Average Communication through the Middleware
Source: UniversAAL - Original Figure [69]

There exist communication rules between the layers, as can be seen in Figure 3.3. We differentiate between inter-layer communication and intra-layer communication. Intra-layer communication takes place between partners in the same layer and in inter-layer communication is the information exchange between different layers. Any exchange of information between an *AAL Application* and an *AAL Space Manager* is indirectly, addresses the middleware first and is passed on from there [69]. However, this does not only happen if a manager addresses an application or other way round [69]. Even for communication among managers and among applications the middleware is the postman.

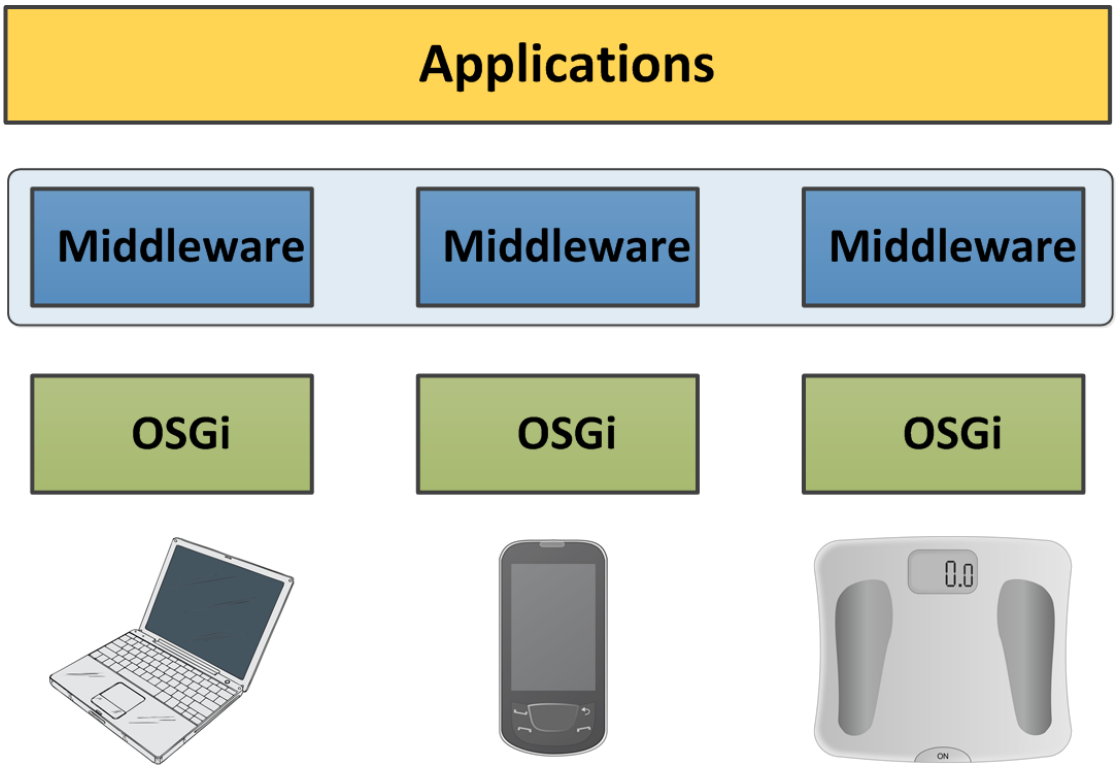


Figure 3.4: Hiding Distribution and Heterogeneity through Middleware
Source: Self-made Figure [69]

The purpose of this layer arrangement is to release the applications of background knowledge about the environment they work in. Applications can behave as if all of them run on a single device, while benefiting from OSGi and the local runtime environment [69]. The middleware hides both the distribution and the heterogeneity of the system's individual parts from each other [69]. Figure 3.4 depicts the behaviour that on application layer it does not matter on which device the specific applications are running. They wrap the information or request as a message and feed it to a bus of the middleware [69]. The bus decides how to handle the message and sends it to the other nodes if necessary [69].

OSGi

A challenge for AAL frameworks is the dynamic behaviour of devices and applications. Mobile devices, such as smart phones, body sensors or other portable devices leave the framework and may re-enter the network later again. This can be caused for instance by a change of location, a device system error or by being switched on and off. There are infinite possibilities and the specific behaviour is unpredictable. The same applies for applications. They can come and go due to a new installation, an update, a fail, a restart or because they get uninstalled. Such changes are routine and have to be handled uncomplicated and without a restart or even a short interruption of the entire AAL system. The platform and the applications have to adapt themselves dynamically to any changes. Dependencies grow with the installed applications. A solution that supports such continuous operation is OSGi.

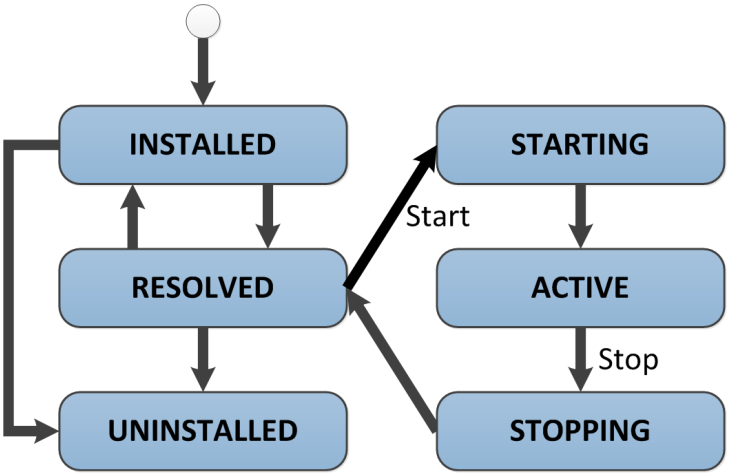


Figure 3.5: OSGi life cycle
Source: Self-made Figure

The OSGi framework is a service platform for Java where every application is a separate bundle with an own life cycle. This life cycle is illustrated in Figure 3.5 with its states *Installed*, *Resolved*, *Started*, *Active*, *Stopping* and *Uninstalled*. Bundles in the OSGi framework can change their state without affecting the overall framework directly (apart from providing its specific service as an application). The life cycle management administers the bundles dynamically and provides a modular way to tackle problems and decouples between service providers and service requesters. Generally the bundles can provide services to one another and these services can come and go at any time.

Java Virtual Machine (JVM)/Java Runtime Environment (JRE) The JVM is the source code execution component of Java. Such JVMs are available for many platforms and is an additional layer on the top of the operating system that translates the bytecode to the respective system environment in particular. Thus it is not necessary to adapt the source code to the used hard- and software. The JVM, bundled with the libraries that implement the Java API, forms the JRE.

JNI The JNI is the interface of the Java to use native applications and libraries written in other programming languages. Consequently, the software loses its benefit of platform-independent. In our case our application is tailored for a specific system environment using Linux and its BlueZ Bluetooth stack. We would need a JNI to use the Antidotes libraries written in C language in order to receive ISO/IEEE 11073 messages via Bluetooth.

Figure 3.6 explains the position of the UniversAAL middleware in the context of OSGi and how it is embedded into the overall system. The lowest layer is the hardware in control of an OS and drivers. JRE runs on top of it like the JNI does, and the OSGi framework is one layer above. In turn the middleware is between the OSGi framework and the applications which are wrapped in bundles. This sequence is also illustrated in Figure 3.4. Both, applications and managers have access to the middleware API. Nevertheless, like it is necessary to have a mailbox to receive letters from the postman, there must be a footprint of the middleware on the node to guarantee a network of AAL-aware nodes that can host AAL-ready software components [69]. They communicate with other applications and resolve nodes with the help of UniversAAL's middleware [69]. To hide the distribution the nodes must be discovered and enabled in order to exchange messages among each other [69]. The middleware uses a unified data representation framework that enables the nodes to exchange data in a way that the heterogeneity of technologies is absorbed [69] and communication takes place between nodes from diverse domains, but with the same data model [69]. Summing up, the main purpose of middleware is its acting as a broker [69].

Context Bus and Service Bus

Figure 3.7 displays the shared middleware within a node. For the exchange of information the middleware uses different kind of buses:

- *UI Bus*: This bus is used to describe the User Interface (UI) and its graphical elements, such as buttons, text fields or labels. It is a bi-directional communication and both, the external control of the UI as well as the user input, use the UI bus.
- *Context Bus*: This bus is event based, which means that only if something changes a context event is published. Not everybody receives all messages from the *Context Bus*. If interested in a specific type of events, first, a context event pattern has to be created.

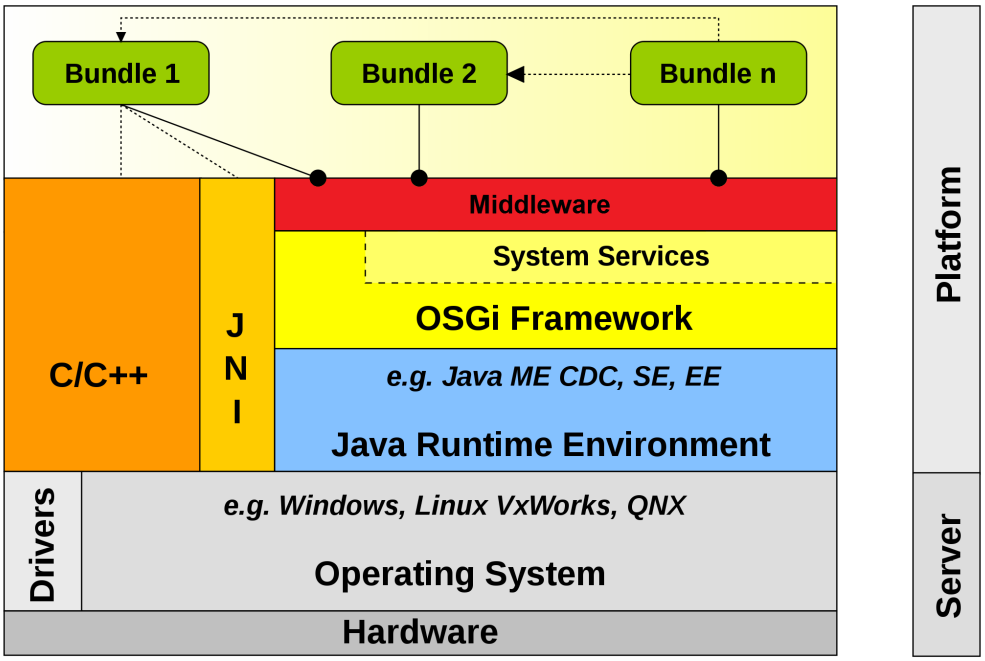


Figure 3.6: UniversAAL Middleware in the context of System Layers including OSGi
Source: Wikipedia - Modified Figure [73]

These patterns formulate the restrictions and the matching context events are forwarded to the respective subscriber. The *Context Bus* is event based, but this does not mean that every event should be published. It is necessary to reduce the amount of data and send only significant changes, but no streaming data. A context event is a fact. This connotes that actual data after a change are published, but not the change event ('The result is 7' is a fact, 'You have to subtract 2' would be a change event.). Not necessarily every subscriber knows the state before the change. Hence this information would be potentially useless. When an event is published, every subscriber receives a message.

- *Service Bus*: The *Service Bus* on the other hand is call based with the primary purpose to get or modify data. A service request describes the service(s) of interest and the *Service Bus* finds the most specialized service that matches the enquiry. The bus principally has the purpose of receiving or modifying data. Such a service request consists of four parts:

  1. Service
  2. Input
  3. Output (multiple outputs possible)
  4. Effect (add, remove, change)

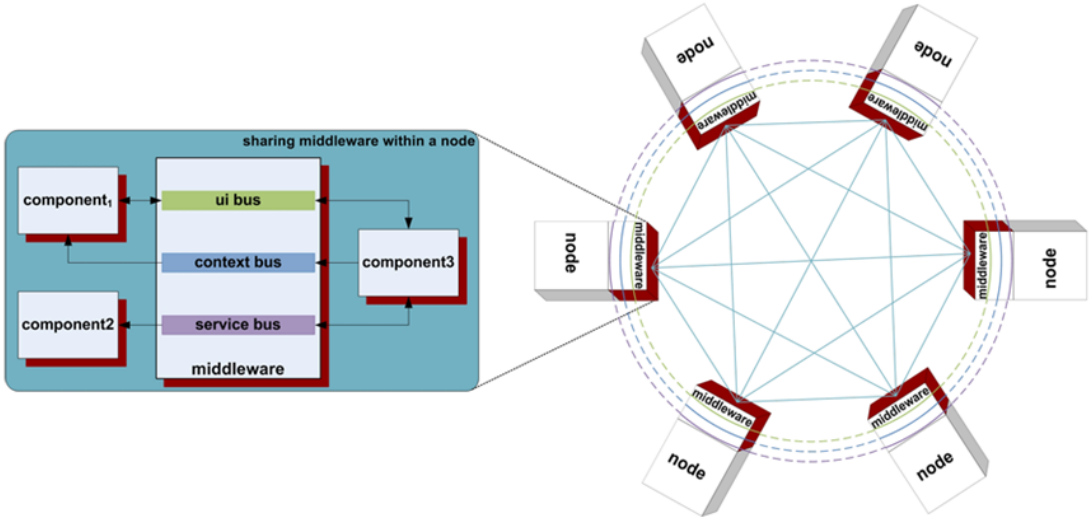


Figure 3.7: Middleware und Data Representation
Source: UniversAAL - Original Figure [69]

Web Ontology Language (OWL) OWL is a knowledge representation language to describe ontologies. It uses Resource Description Framework (RDF) triples and is a pillar of the semantic web [73].

The *Context Bus* and the *Service Bus* both work with patterns and matchmaking. Data are formulated in a common understandable format, namely an ontology. To subscribe to context events or request a service OWL is used for matchmaking.

3.2 Ontologies

Many definitions of the term ontology exist. It has its origin in philosophy [27]. The literal translation is the science of being and is the metaphysical study of the nature of being, existence, or reality [27]. It discusses the questions:

- What characterizes being?
- What is being?
- How should things be classified?

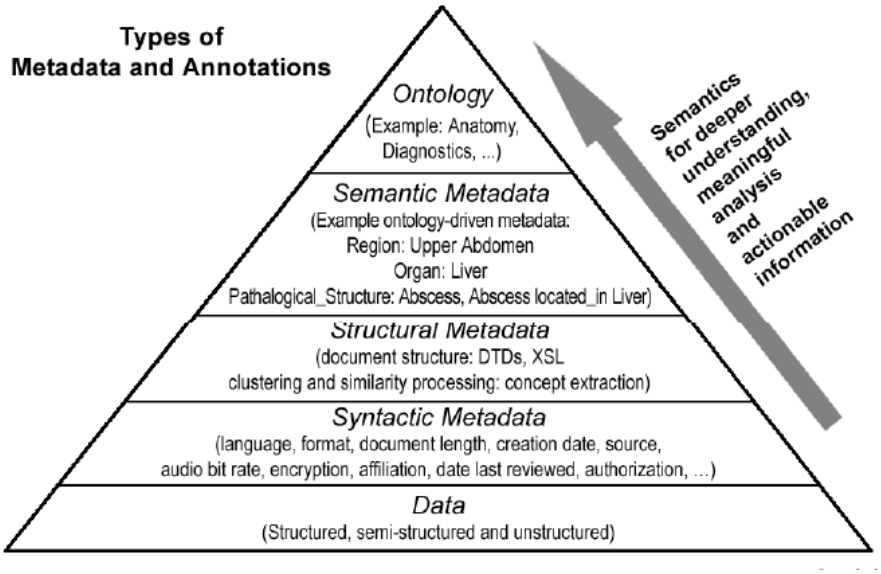


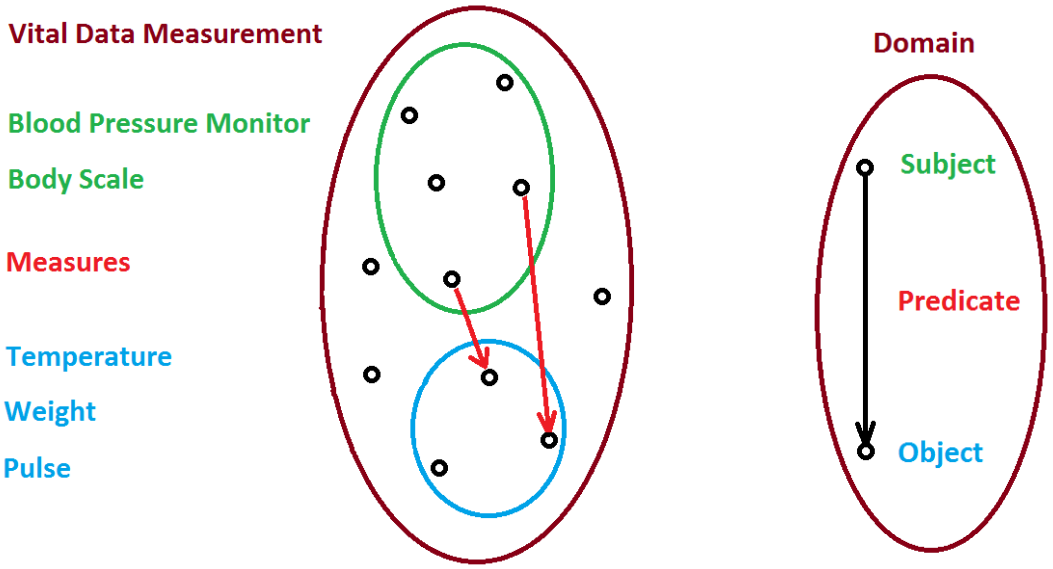
Figure 3.8: Semantics and Ontologies
Source: deri.ie - Original Figure

The study of semantics is the investigation of meaning and related to ontologies. In Figure 3.8 we depict the different types of metadata and annotations. We show the link between conventional (maybe even unstructured) data and ontologies, and this link is the deeper understanding through semantics.

Ontologies in Computer Science

In this section we will discuss the specific meaning of ontologies as they relate to computer science. In 1993 Tom Gruber of Stanford University explained ontologies as a specification of a conceptualization [27]. This statement proclaims that ontologies are descriptions of the concepts of entities in a specific domain including the relationships that exist among them. These entities can be either concrete objects or collections of such objects called classes. An ontology consists of three parts:

- A vocabulary to describe a domain or a particular part of it. E.g.: PHD, CE, pulse, blood pressure monitor
- An explicit specification of the vocabulary's intended meaning. E.g.: A blood pressure monitor is a concept which members are of the kind PHDs.
- Constraints to give a more detailed description and to document additional knowledge about the domain. E.g.: The normal pulse rate at rest is between 60 and 100 for adults.



(a) Example: Measuring Vital Data

(b) RDF Triple

Figure 3.9: Ontology - Description of the Logic
Source: Self-made Figure

So we can interpret ontologies as a set of concepts [27]. Figure 3.9 (a) illustrates an example of entities in a domain named *Vital Data Measurement*. In there are two classes of concepts, PHDs and body functions. All of them are part of our vocabulary. To put meaning into this model we add the relationship between e.g. the *Body Scale* and *Weight*. In ontologies these relationships are expressed as properties between a subject and an object. This general case is illustrated in Figure 3.9 (b). There are two types of properties:

- Datatype Properties: These are specific attributes that can be expressed by a certain value of an already existing data type.

- Object Properties: This kind of property describes the relation to another entity. This object can be either a newly defined data type or a structure. Object properties may have characteristics such as transitivity, (a)symmetry, (ir)reflexivity or being (inverse) functional.

General Purpose and Use in UniversAAL

The next step and the content of this subsection is to understand the purpose of ontologies in UniversAAL. Ontologies describe knowledge in a very general way and with an inner structure and given syntax, such that the result is understandable and interpretable for human and machine. Uschold defined ontologies as a shared understanding of some domain of interests [70]. Consequently, an ontological commitment is an agreement to use a specific consistent vocabulary. The group of people and machines that use this common vocabulary are able to share and reuse knowledge among them. The intention of ontologies is:

- communication among people
- interoperability between software agents
- reusing domain knowledge
- making domain knowledge explicit

It aims to specify meaning in a machine-understandable manner [27]. This idea is not a unique approach. In fact it is simply one method for standardization. More precisely, it is the method chosen in UniversAAL for the exchange of information among several services.

Ontologies are connected to the UniversAAL middleware like the other AAL services. To share information by using such an ontology which is accessible via the *Context Bus* and *Service Bus* of the middleware. Remembering the previous section, context event patterns and service requests are necessary to filter the supply of services and concentrate on the needed context events. In UniversAAL data must be available in the RDF format and such patterns and requests formally are nothing more than an RDF triple consisting of subject, predicate and object. This is the same triple used in Figure 3.9 (b). In UniversAAL such a triple could be a service or a context event for instance.

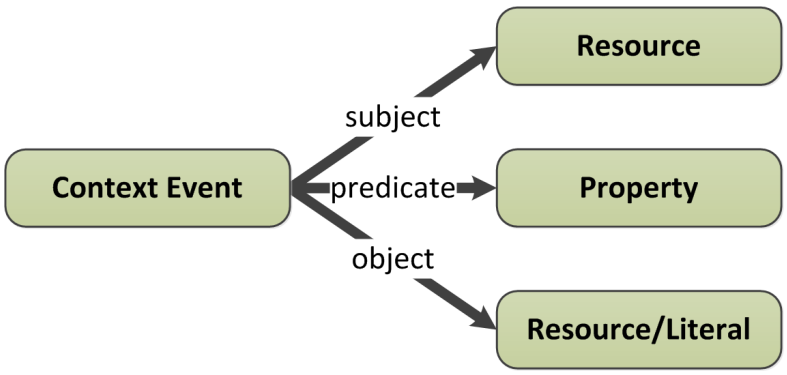


Figure 3.10: RDF Triple of a Context Event
Source: Self-made Figure [69]

Figure 3.10 shows such a triple of a context event. The subject is a resource like a blood pressure monitor for instance. The predicate is a property and is typically a verb or a phrase of verbs such as *is_a* or *measures* like we had it in the example in Figure 3.9 (a). The object is either another resource (object property) or a literal (data type property). The blood pressure monitor would measure the systolic blood pressure or the heart rate.

3.3 Continua Health Alliance

The Continua Health Alliance is a non-profit, open industry organisation of more than 200 healthcare and technology companies joining together in collaboration in order to increase the quality of personal healthcare in a global context [14]. The organisation was founded in June 2006 and, following to first public demonstrations of interoperability in 2008, the first product was certified in the beginning of 2009. The first version of Continua's guidelines provide USB for cable-connection and Bluetooth for wireless device connection. Due to the impractical attributes of Bluetooth regarding the power-consumption and the trend towards wireless communication the Continua Health Alliance version 2 design guidelines introduce low-energy wireless transport protocols. Sensium, ZigBee, ANT+, BLE, BodyLAN, and Z-Wave were considered for the selection of the new flagship transport protocol [50]. Finally, BLE and ZigBee made the running. In August 2012 52 certified products can be found in the product showcase on the Alliance's website [2]. The mentioned design guidelines are free of charge to the public and can be found on the website since the beginning of year 2012 [2].

Figure 3.11: Continua Health Alliance Logo
Source: continuaalliance.org

Continua is working towards establishing industry standards and improve interoperability and compatibility of PHDs [14]. Consequently, systems of interoperable personal telehealth devices and services are in the focus and the Continua Health Alliance describes its scope as the three categories:

- Chronic Disease Management
- Ageing Independently
- Health & Physical Fitness

This can either mean that a new standard has to be developed, but using already existing ones and adapting them on the domain is even more likely. The ISO/IEEE 11073 is Continua's main protocol stack [22]. The goal is to establish a system of connected interoperable personal healthcare solutions and Continua's activities include, additionally to the development of guidelines, the certification of healthcare products following them, and to support a basis for collaboration

in a global network of developers, vendors, users, and government agencies [2].

Concerning the matter of related goals and the big influence the Continua Health Alliance has on the market of personal healthcare solutions we keep track of its trend-setting decisions and use Continua-certified products as our evaluation devices. This improves the performance of our solution due to the wide spreading of these products. In this way we exploit a wider range of users and reduce the barrier for others. Furthermore, the Continua Health Alliance products make use of the ISO/IEEE 11073 PHD Standards [14].

3.4 The ISO/IEEE 11073 Health informatics - Medical/Health Device Communication Standards

The IEEE is dedicated to push on technological progress and is one of the leading standards-making organizations in the world. The IEEE Standards Association (IEEE-SA) is a sub-organisation within the IEEE with the purpose of standards making and maintaining.IEEE standards affect a wide range of industries; Power and energy, biomedical and healthcare, Information Technology (IT) or telecommunications are only some examples. The use of these standards is voluntary and is not a binding regulation like the ones of national standards bodies such as the Deutsches Institut für Normung (DIN) or the American National Standards Institute (ANSI) [36]. The existence of such a norm does not state the single truth and leaves space for alternatives. Nevertheless, for some topics (e.g. IEEE 802 - Local Area Network (LAN)) they hardly exist. The ISO on the other hand is a worldwide federation of such national standards bodies [36]. In the year 2000 a pilot project has been started to develop and maintain a group of ISO/IEEE standards in the field of medical devices [36]. The duty of the IEEE is to develop and maintain these standards with the participation and input from ISO member bodies [36]. The ISO/IEEE 11073 was prepared by IEEE 1073 Committee of the IEEE Engineering in Medicine and Biology Society in 2004 [36].
The IEEE describes the purpose of the standard with the following words [36]:

> ISO/IEEE 11073 standards enable communication between medical devices and external computer systems. They provide automatic and detailed electronic data capture of patient vital signs information and device operational data.

Purpose and Goals

The ISO/IEEE 11073 Personal Health Device Data work group focuses on improving personal telehealth systems and PHDs in particular [58]. The group aims at developing new standards for medical device communication within the family of already existing ISO/IEEE 11073 standards [58]. They emphasis on devices for personal use rather than hospital use. These devices realize the concept of agents and have different demands than conventional medical devices, because especially sensors and other battery-powered devices have a lower computational capability, reduced size and light weight [22]. Therefore, they need a simpler communication

model. With the hypothesis that the managing CE has more capabilities than the agent the protocol places more burden on it and locates most of the intelligence of the systems away from the PHDs and sensors [22]. Each agent is connected to a single manager and this manager possibly communicates with multiple agents. Furthermore, the standards support multiple data types (episodic, streaming, store and forward) and are designed to be transport-agnostic. The message structure is defined, but not the transport. Principally there are two main goals [36]:

- Interoperability: It is aimed to provide a standardized basis for real-time interoperability for PHDs. The connection of the devices is plug-and-play. That means that the system releases the burden of detection and configuration of the user who must not be bothered with such tasks. The communication is provided without any need for human interaction.
- Point-of-Care: The standard facilitates efficient communication of vital signs and medical device data at the point-of-care and independent from the specific healthcare environment.

The ISO/IEEE 11073 PHD supports three domains (with intersections):

- Disease Management: E.g. pulse oximeter, heart rate monitor, blood pressure monitor, thermometer, or glucose meter.
- Health and Fitness: E.g. heart rate monitor, body scale, cardiovascular fitness and activity monitor or strength fitness equipment.
- Independent Living: E.g. disease management devices or medication monitor.

Interoperability is composed of two parts. It involves the ability of two or more systems or components to exchange information [58]. Sending a message is not enough, communication takes place only if the exchanged information can be interpreted [58]. Like in communication sending a message is not enough. The right interpretation is essential. This problem has to be tackled on three layers [58]:

1. Transport Technology: This layer facilitates the basic physical connection between the participating systems and components.
2. Application Profiles: These profiles define how to meet the application's requirements.
3. Data Models: The applications need standardized data models and formats which are developed based on the application profile layer.

For the physical layers there already exist many standards and de-facto standards (e.g. Bluetooth, ZigBee, WiFi) [58], but if we look at the top of this layer model there is still a lot of work to do [58]. For medical imaging data Digital Imaging and Communications in Medicine (DICOM) is popular, and HL7 is a comprehensive set of standards for interoperability of healthcare computer applications rather than PHDs [58]. The Continua Health Alliance is an open industry group with the goal to promote the growth of personal and individual healthcare. Ensuring multi-vendor interoperability is a crucial pre-condition for wide-spread distribution of PHDs [58]. Continua works together with the IEEE and makes use of the ISO/IEEE 11073 PHD standards in order to push the personal telehealth market [58].

The Architecture

Table 3.1 provides an overview of the ISO/IEEE 11073 set of standards [58]. The core standards of the classic ISO/IEEE 11073 are the parts 10101, 10201, 20101 and 30200. For mobile devices the application profile 20601 and wireless transport profiles grow important. In this thesis, if nothing different is explicitly written, we discuss the results of the ISO/IEEE 11073 Personal Health Device Data work group and application profile 20601.

The number of each part already implies its purpose. The first number specifies the layer with which the part can be associated. *Data and Information Definitions* according to the 1yyzz-series provides interoperability on layer 7, the application layer, if talking in terms of the OSI model [58]. If the first digit is a 2 the standard deals with *Application Profiles* and provides specific sets of restrictions to meet the application's requirements [58]. These regulations are independent of any specific device type [58]. The standard 20601 plays a special role for us. The *Optimized Exchange Protocol* is a mechanism to reduce the overhead of exchanging messages and configuration process [58]. This part is especially adapted for battery-powered medical devices with low processing power [58]. The series with higher numbers aim to provide solutions on lower layers regarding the OSI model. They deal with transport issues and target the physical connection itself [58]. The ISO/IEEE 11073 is transport-agnostic, meaning that it does not depend on a specific transport protocol. That the transport layer is the foundation for the other standards can be seen in Figure 3.12.

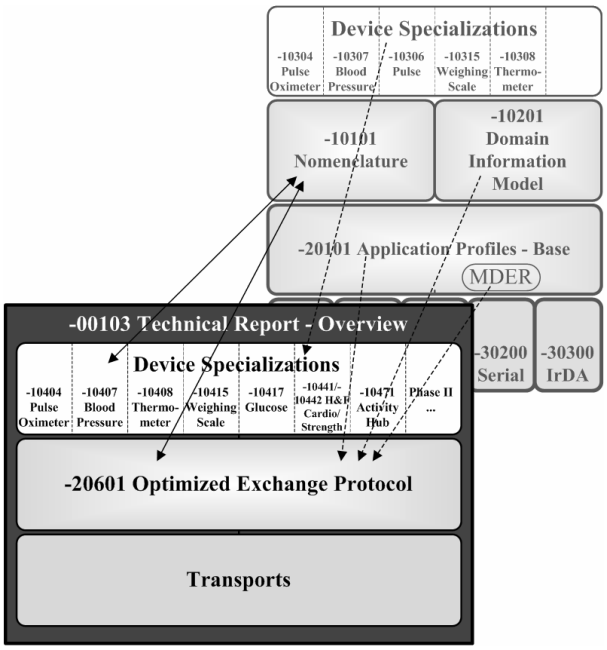


Figure 3.12: Relationship between ISO/IEEE 11073 and ISO/IEEE 11073 PHD
Source: ISO/IEEE 11073-20601 - Original Figure [38]

OSI Layer	Part	Title
	00000	Health Informatics - Point-of-Care Medical Device Communication - Framework and Overview
	00103	Health Informatics - Personal Health Device Communication - Technical Report - Overview
7	**1yyzz**	**Data and Information Definitions**
	10101	Health Informatics - Point-of-Care Medical Device Communication - Nomenclature
	10201	... - Domain Information Model (DIM)
	10300	... - Device Specialization - Framework and Overview
	10307	... - Blood Pressure
	10314	... - Respirator
	10315	... - Weighing Scale
	10400	Health Informatics - Personal Health Device Communication - Device Specialization - Common Framework
	10408	... - Thermometer
	10415	... - Weighing Scale
	10417	... - Glucose Meter
7-5	**2yyzz**	**Application Profiles**
	20101	Health Informatics - Point-of-Care Medical Device Communication - Application Profiles - Base Standard
	20102	... - MIB Elements
	20201	... - Polling Mode Profile
	20202	... - Baseline Profile
	20601	Health Informatics - Personal Health Device Communication - Application Profile - Optimized Exchange Protocol
3-1	**3yyzz**	**Transport and Physical Profiles**
1	**4yyzz**	**Physical Layer Interface**
3	**5yyzz**	**Internetworking Support**
4	**6yyzz**	**Application Gateways**
	9yyzz	**Related Concepts**

Table 3.1: The ISO/IEEE 11073-xyyzz Series of Standards
Source: Schmitt et al. - Self-made Table [58]

Part 10101 defines the standardised nomenclature consisting of numeric codes for the terminology to identify every item in the system and provide the common vocabulary for communication [58]. Generally the standard follows an object-oriented system paradigm and the DIM is used to specify objects, attributes, groups, event attribute reports, and communication services [58]. It describes the device and the physiological data. These elements of the model are used for data exchange and device configuration [58]. The 103zz and 104zz set of standards specifies device specialisations for several medical devices. The list in Table 3.1 is incomplete but enumerates some examples. We focus here on the 104zz-series because they follow the guidelines for PHDs. The modular approach simplifies adding support for new device types.

The hierarchical view on the architecture in Figure 3.12 depicts the relationships between the ISO/IEEE 11073 documents. The new part 20601, called *Optimized Exchange Protocol*, replicates relevant portions of ISO/IEEE 11073-10101 and incorporates new nomenclature codes (Figure 3.13). Additionally it imports information from the parts 20101 and 10201 as normative annexes.

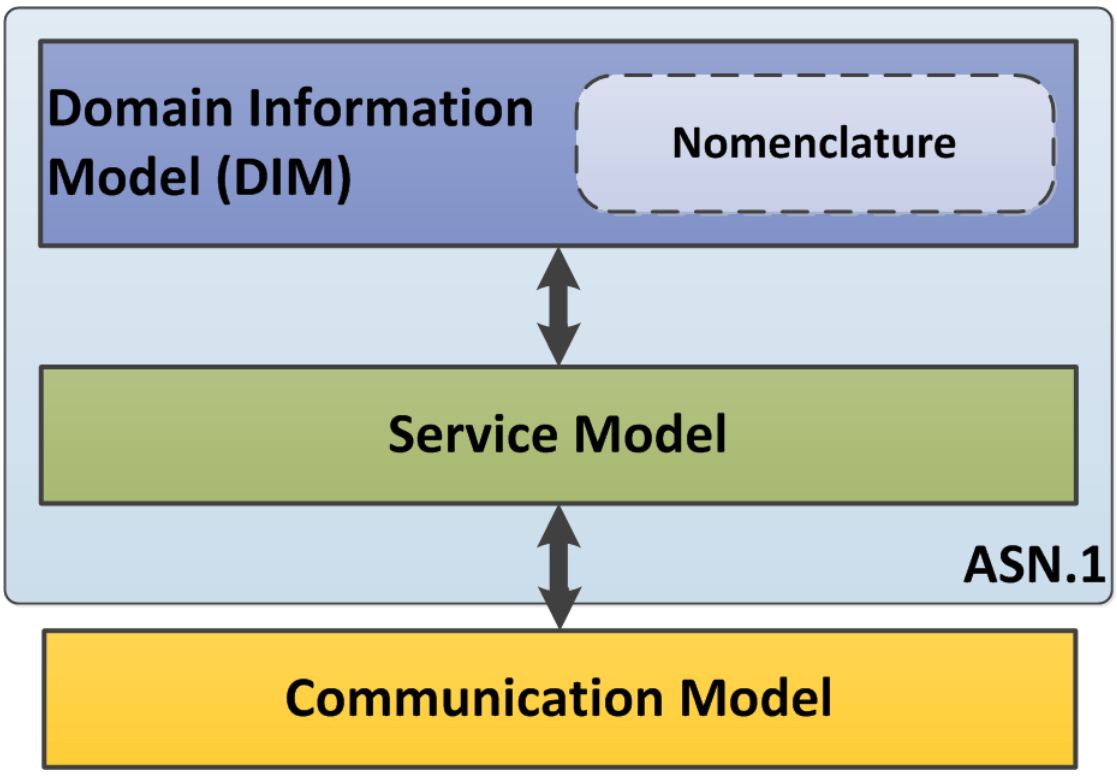


Figure 3.13: ISO/IEEE 11073-20601 Modelling
Source: Self-made Figure [38]

The overall ISO/IEEE system model is divided into three principal components [38]. Their interrelationship to each other is illustrated in Figure 3.13 and described in the following list of the announced three components [38]:

- DIM: As mentioned above this model describes the information of the agent in an object-oriented way [38]. Each object is a class, and consists of methods and attributes that characterise either healthcare information or device configuration [38]. Attributes may be mandatory, optional or conditional [38]. Additionally they can be static or dynamically changing. The attributes are defined in ASN.1. Figure 3.14 illustrates how such a DIM looks like and shows the model of the 11073-10407 blood pressure monitor. Each agent has one Medical Device System (MDS) with general information for identification, reporting its status and providing some other core information, such as the time settings [38]. In our figure we have two objects - *Systolic, Diastolic, Mean Arterial Pressure (MAP)* and *Pulse*. This structure depends on the features and possibilities the specific device type provides [38]. It covers all possibilities a device of this type could facilitate and the concrete dataset depends on the specific model of the PHD and its settings [38] (e.g. a blood pressure monitors does not necessarily provide a measurement of the pulse, but the model provides an object for the case that it does).

- Service Model: Like in the DIM the message format is described in ASN.1. The *Service Model* provides data access primitives that are exchanged between agent and manager in order to transfer DIM data [38]. It offers an *Event Reporting Service* (Config, Data Update), an *Object Access Service* (*Get, Set, Actions*), an *Association Service* (Associate/release request/response, abort) and a *Conversion Service*. The messages separate into three types. Firstly, there are those messages for associating the agent and its manager [38]. Furthermore, the manager can contact the agent to access its DIM in order to gather information like its attributes and to change its settings [38]. The third kind of message is the measurement data that is sent from the agent to the manager [38]. Each command calls for a confirmation and is resent until there occurs a time-out [18]. In this case the association is aborted [18].

ASN.1 ASN.1 is a standard and flexible notation that provides a set of formal rules representing data structures and messages that are independent of machine-specific encoding techniques. In case of health devices the objects are converted into binary data streams using MDER.

- Communication Model: Every agent is connected to its manager via a logical point-to-point connection [38]. The *Communication Model* manages these connections and supports the topology to enable the manager to communicate with one or more agents [38]. The connections are defined by a state machine with the states *Connection*, *Association* , and *Operation* [38].

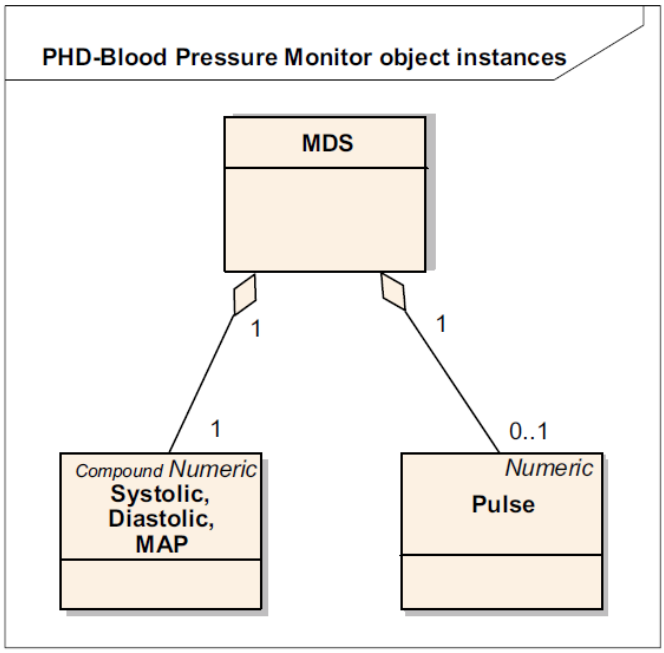


Figure 3.14: ISO/IEEE 11073-20601 Modelling
Source: ISO/IEEE 11073-10407 - Blood Pressure Monitor - Original Figure [37]

The ISO/IEEE 11073 splits the data into network packets called Application Protocol Data Unit (APDU) with a maximum size of 65535 octets [18]. Such an APDU is similar to frames in Ethernet (with the difference that APDUs are bigger) [18]. The splitting process and the reassembling afterwards have to be facilitated by the transport layer [18]. Every kind of PHD has different attributes, every PHD type can be implemented by numerous device models, and a device could offer a range of settings. All these factors influence the inner structure of the data. Therefore, the data are encapsulated with inside the APDU [18]. Furthermore, there are several APDU types depending on the state of the *Communication Model's* state machine [18].

Using the three components, *DIM*, *Service Model*, and *Communication Model*, the ISO/IEEE 11073 PHD facilitates a transport-agnostic way to transfer messages between PHDs and CEs. These so-called agents are connected to their managers and send vital data, typically measurement data of sensors.

3.5 Architecture

In Chapter 2 we evaluated AAL projects, technologies for wireless data exchange and other ISO/IEEE 11073 manager implementations. This was the foundation for fundamental design decisions. In this chapter we will go down to the details of UniversAAL, our evaluation AAL platform, and, hence it is based on the previous chapter. However, all these sections do not end in itself, but have the purpose to gain insight, and consequently, design an architecture for our ISO/IEEE 11073 interface following an underlying concept. At first, this section will conclude Chapter 2 and Chapter 3 in order to develop the architecture of our implementation.

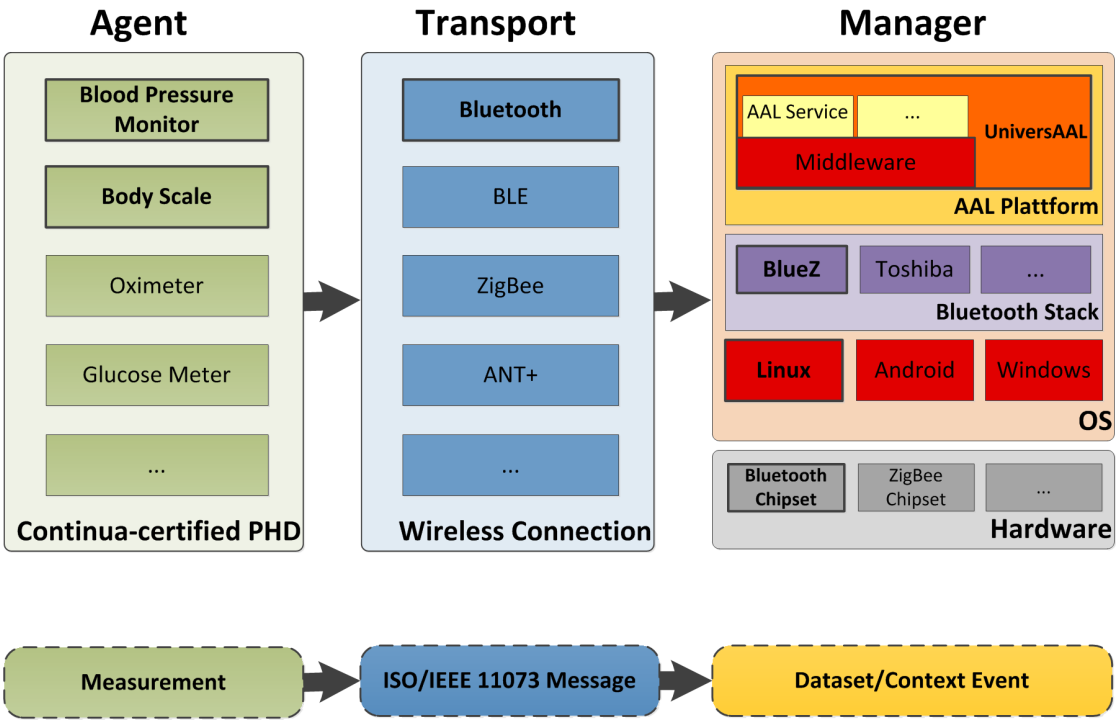


Figure 3.15: Measurement finds its Way to an AAL Platform
Source: Self-made Figure

Using the ISO/IEEE 11073 vocabulary we have agents and managers that are connected either via a cable or a wireless connection. An agent is a PHD and sends its data to a single manager, while this manager is connected to at least this one, but possibly several devices. To stay comparable to other initiatives with similar ambitions, we use Continua-certified PHDs as agents. More precisely, we use a blood pressure monitor and a body scale as our evaluation devices. The interface can be adapted for other Continua-certified PHDs easily, but due to restricted resources it is not test in the context of this work. Furthermore, this thesis is a proof of concept without claiming to be an off-the-box solution for all devices available on the market.

Figure 3.15 illustrates the communication path from the device's side to the UniversAAL platform. There we find the agent on the left side.

The PHD measures and wraps the measurement data into an ISO/IEEE 11073-conform message. Due to reasons of usability the sensor uses a wireless transport protocol to transfer the data (center block in Figure 3.15). We analysed several protocols and decided to use Bluetooth because of its wide spreading in the market of PHDs. Most mobile phones and laptops have an in-built Bluetooth chipset out-of-the-shelf and a new device can be paired easily and fast. For desktop computers Bluetooth dongles are cheap and easy to get and install.

On the right side the figure depicts the manager, which is a CE. UniversAAL is the AAL platform of our choice and in the test environment of ours. We run it on a desktop computer. An other kind of CE is possible, but the OS is a decisive factor. UniversAAL is tested for Linux and Windows. The Bluetooth profile for medical devices is the HDP and is not implemented in every Bluetooth stack. Android ICS 4 provides such a stack out-of-the-box, Linux uses the open source solution BlueZ that implements HDP since 4.80, and for Windows there exists the Toshiba Bluetooth stack, which is not open source. Therefore, it does not meet our requirements. Concluding we use UniversAAL in an Linux environment with the BlueZ Bluetooth stack. Inside our AAL system UniversAAL uses its middleware for communication between AAL services. For the middleware the vital data are represented as a *Context Event*.

The figure explains the design of our interface and the general architecture is the underlying concept of the implementation.

CHAPTER 4

Implementation

EVEN the best preparation has to step aside at some point in favour of its realisation. In this chapter the implementation itself will be described. The need for this implementation is what we described in Chapter 1. What technologies and projects already exist and assemble the environment of our thesis is concluded in the chapter about the state-of-the-art. The previous chapter discussed the design of the solution, and finally this one explains how we put this theory into praxis. We examine the three parts of our interface:

- Antidote IEEE 11073-20601 stack
- Gateway between the incoming vital data and the UniversAAL platform
- Interworking with the UniversAAL middleware

Figure 4.1 illustrates the full stack of our implementation. The lowest layer is the Bluetooth chipset. The managing CE has to be Bluetooth-enabled because the PHDs transfer their data via this wireless transport protocol. In this chapter we are not going further into detail on this requirement. Another requirement is a recent version of the BlueZ. In the first section we will touch upon the Bluetooth stack and the according profiles. However, this is only the basis for Antidote's library that implements the ISO/IEEE 11073-20601 stack. Antidote is written in C language and has only little dependencies. It puts the received ISO/IEEE 11073-conform data on the D-Bus and the gateway reads it in in order to process it further. This gateway between the Antidote library, with its ISO/IEEE 11073 data, and UniversAAL, with its *Context Events* is the core of our implementation. It is realised as an OSGi bundle running in the UniversAAL environment and handles data of both worlds. In the last section of this chapter we will describe the ontology representing the vital data measured by the PHD. Furthermore, every other AAL service on the UniversAAL platform can access the read-in data since they are put on the *Context Bus* as a *Context Event*.

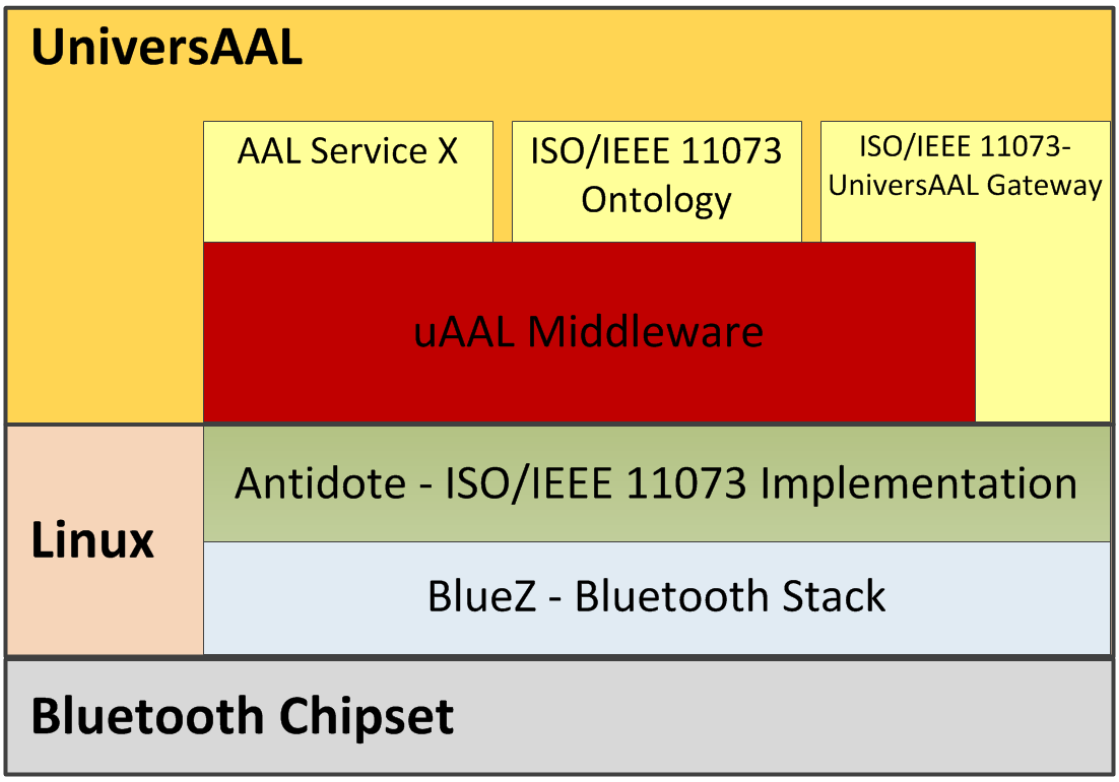


Figure 4.1: Stack of the Implementation
Source: Self-made Figure

4.1 Antidote ISO/IEEE 11073-20601 Stack

In Section 2.3 we examined Antidote's history and purpose, and we paid attention to its architecture. In short, we explained the four parts of the Antidote ISO/IEEE 11073 stack library [6]:

- Core Stack
- Specializations
- Sample transport plug-ins
- Health D-Bus Service

In this section the library will be discussed more in detail, and we will focus on its use for this master thesis. It will describe the reusability of the library and how it interacts with the other parts of the implementation.

Antidote Architecture and Plug-in Description

Antidote fits in our system of modules as receiver for ISO/IEEE 11073 data. Its focus on portability is favourable at this point. The portability is the library's main goal and implied a development in C language with a minimum of dependencies [18]. The existing code with its provided plug-ins is runnable, after its compilation, off-the-box for most systems [18]. Depending on the choice of OS and transport protocol the executable may have some dependencies or needs adoption. Due to its modular concept almost any transport protocol may be used [18]. Apart of the provided plug-ins for Bluetooth, USB, and TCP/IP, most other transport protocols can be implemented as a transport plug-in and added to Antidote [18]. Antidote's modularity also allows to reuse parts of it while discarding others. This feature is not important for our implementation itself, but may be helpful for possible modifications in the future. The internal encapsulation and structure of Antidote is adapted from ISO/IEEE 11073 layering with its three components *DIM*, *Service Model*, and *Communication Model* [18]. Inspecting Figure 4.2 these similarities are revealed. The components of the stack, the *Device Specializations* and ASN.1, PHD, MDER blocks are already familiar for us.

The *Data Encoder* is not part of the ISO/IEEE 11073 model, but releases the user of dealing with MDS types [18]. Antidote encapsulates the data and sends it via IPC [18]. It would be possible to access the MDS directly, but since we have to process the data later on anyway it is much more convenient to handle it in an encoded way [18]. Antidote offers Extensible Markup Language (XML) and JavaScript Object Notation (JSON) encoders. However, due to its modular architecture any other encoder can be added [18]. XML is the default setting and is the one we use in our implementation, because of the possibilities offered by Java for processing XML-encoded information.

The *Communication Plug-ins* are a key feature of Antidote and deal with dependencies of transport protocols and timers. As mentioned previously Bluetooth and TCP/IP are already implemented. For our implementation it is not necessary to contribute to the existing plug-ins, but if other protocols, such as ZigBee or BLE, grow important in the future it might be demanded. If other transport protocols are needed a new standard C interface has to be developed which adds following services, regarding the desired protocol, to the Antidote stack [18]:

- Manage Connections: Open and Close connections and notify the stack about it.
- IDs: Each connection of each device gets an unique ID that has to be created and maintained by the communication plug-in.
- Communication: Notify stack about new APDUs and other way round it has to accept and forward the ones sent by the stack.
- Asynchronous Timers: Antidote is transport-agnostic and cannot care about asynchronous operations. The plug-in handles the asynchronous timers for the stack and chooses the techniques to achieve this kind of operation.

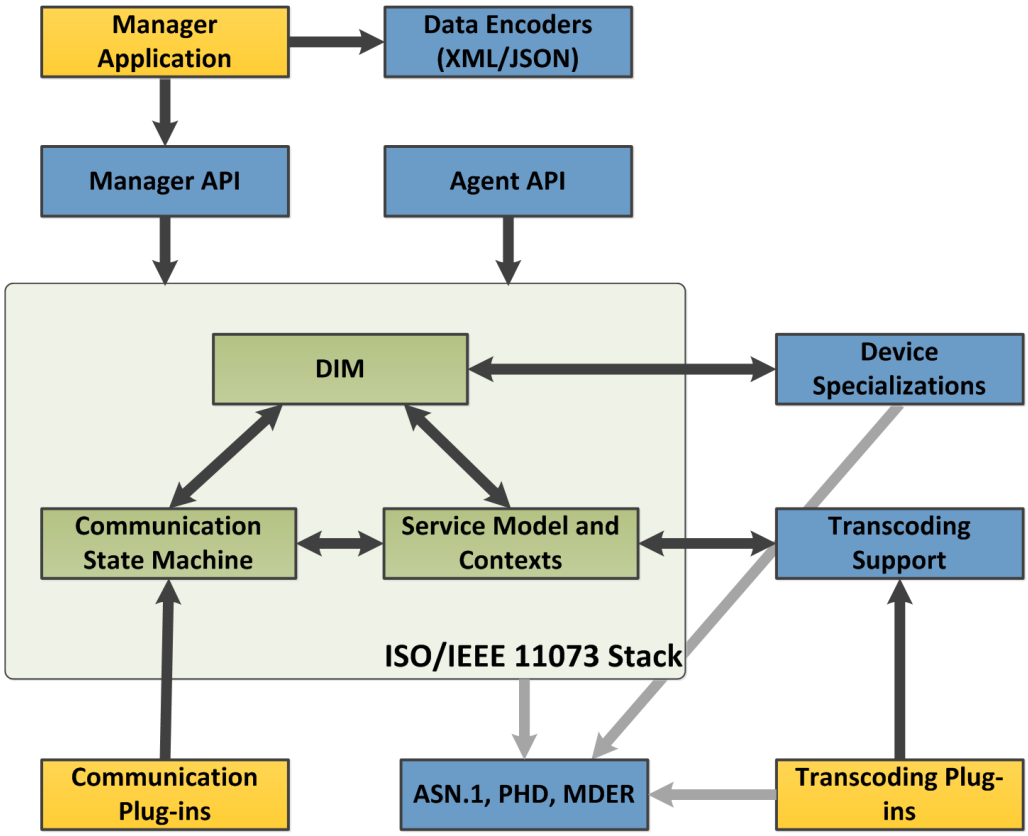


Figure 4.2: Antidote Architecture
Source: Antidote Program Guide - Modified Figure [18]

- Manage Context: The *Communication Context* is the most important structure in Antidote. All device-related information is an attribute of the *Context* and it represents the communication channel with the device. The plug-in's duty is to protect these *Contexts* if multiple threads are used.

For communication with non-Continua-certified products the library supports transcoding [18]. If such a device should be added a *Transcoding Plug-in* has to be developed first. Obviously, every device model demands a different plug-in on its own [18]. It is analog to the *Device Specifications* for ISO/IEEE 11073 devices which enable Antidote to deal with standard configurations. The delivered plug-ins are sufficient for us, since they include the specifications for blood pressure monitors and body scales. However, possibly not every device connected in the future will be included and in these cases a plug-in has to be added to the *Device Specifications*.

The *ASN.1, PHD, MDER* module implements an API to facilitate the development of new *Device Specialisations* or plug-ins for transcoding [18]. It is the counterpart of the ISO/IEEE 11073 structure including its ASN.1 primitives and PHDs in a C *typedef* [18].

Antidote is a library, but is bundled with *healthd*, a full-featured manager application [18]. It is a complete example of an Antidote-based application running on the command-line and publishes its results on the D-Bus. We decided to use this manager instead of reimplementing it. Consequently we never link directly to the Antidote library, but use the included manager. The benefit of this design decision is additional robustness against new Antidote versions and the avoidance of any contact to Antidote's C API. Consequently the adoption of our results to other projects, that use other AAL platforms, and maybe even other programming languages, is much easier due to the unambiguous separation of tasks and smaller reprogramming and translation effort. The draw-backs are that we have to start *healthd* before we can start our bundle in UniversAAL and we have some dependencies. Namely, we need BlueZ version 4.90 or better, Kernel 2.6.36 or better, D-Bus 1.4 or better, LibUSB 1.0, D-Bus Glib and Glib 2.0 [18]. Since we chose to use BlueZ as our Bluetooth stack due to its availability and wide spreading on Linux this dependency has no influence on us. We decided to run Linux and a recent version of Ubuntu fulfils most of the requirements on delivery, and to install the remaining libraries is straightforward and explained in Appendix A. In a nutshell, the dependencies are easy to be met and do not restrict us since they meet with dependencies we already have for some other reasons. The second OS with HDP support is Android. Theoretically, Antidote also runs on Android, using the Native Development Kit (NDK) [18]. Even though Android has no native access to the D-Bus and, therefore, no direct access to BlueZ, other alternatives can be used [18]. For such systems Antidote depends on Java-level support to communicate via HDP instead [18] and consequently, access the vital data sent by the PHDs [18]. Practically, we did not test it due to the focus on the UniversAAL platform. Nevertheless, this feature keeps doors open for further developments and could be handy in the future.

Code Structure and Repository

After analysing the theoretical modules we continue with inspecting the implementation. Table 4.1 reveals and describes Antidote's code structure, which is similar to the module structure in Figure 4.2. The main source code folder is *src/* and its subfolders. In this file folder there is the code of the library itself including its plug-ins and the test programs including the *healthd* manager application.

The library supports unit-testing and has been tested by Signove with a wide range of real-world devices [18], either Continua-certified and non-Continua-certified products. The source code is documented in Doxygen and is available under LGPL [18]. The documentation including the mailing list, and release candidates of the source code can be found at

HTTP://OSS.SIGNOVE.COM/INDEX.PHP/ANTIDOTE:_IEEE_11073-20601_LIBRARY

Path				Description
src/				Source Code of the Library and Sample Applications
	asn1/			ASN.1 and PHD Types
	util/			Utility Modules
	communication/			Communication Model and Service Models
		parser/		MDER Codecs for the PHD and Error Detection
		plugin/		Communication Plug-ins
			android/	Android Plug-in
			bluez/	Linux/Bluetooth/HDP Plug-in
			trans/	Dummy Transcoding Plug-in
			usb/	USB PHDC Plug-in (Linux)
	dim/			DIM
	trans/			Transcoding Support Engine
		plugin/		Transcoding Plug-ins
	specializations/			Device Specializations
	resources/			Non-code Resources (e.g. D-Bus service descriptor)
	api/			XML and JSON Data Encoders
tests/				Unit tests
doc/				Doxygen generated Documentation
debian/				Debian Package Build Files

Table 4.1: Code Structure of Antidote
Source: Antidote Program Guide [18]

Furthermore, there exist a Git project page at

HTTPS://GITORIOUS.ORG/ANTIDOTE

In our implementation we used Antidote version 2.0. In order to stay independent and for ease of updates we hardly touched Antidote's source code and in the final version we used an original Antidote installation without any changes at all. We kept Antidote as a module in our implementation and use the library as a black box. Even though we analyse its inner structure here this knowledge is not crucial for understanding our software itself. Nevertheless, this part is essential in order to understand the sequence of events of the entire implementation.

Encoded Data

Antidote delivers two plug-ins to encode data [18]. We prefer the default mode and use the XML coding rather than the JSON coding. Antidote translates the hierarchic ISO/IEEE 11073 messages to *DataLists* [18]. Each XML document is nothing more than a single *DataList* containing any number of possibly nested *DataEntry* children. Each of these entries may contain meta-data as a description [18]. There exist two types of entries [18]:

- Simple Entry: This kind of entry contains exactly the three attributes *name*, *type* and *value*.

```
<data-list>
  <entry>
    <meta-data>
      <meta name='unit-code'>544</meta>
      <meta name='unit'>%</meta>
    </meta-data>
    <simple>
      <name>Basic-Nu-Observed-Value</name>
      <type>float</type>
      <value>29.00000</value>
    </simple>
  </entry>
</data-list>
```

- Compound Entry: In difference to *Simple Entries* this type contains other entries and provides the basis for nested structures.

```
<data-list>
  <entry>
    <meta-data>
      <meta name='unit-code'>3872</meta>
      <meta name='unit'>mmHg</meta>
    </meta-data>
    <compound>
      <name>Compound-Basic-Nu-Observed-Value</name>
      <entries>
        ... three simple entries meaning systolic,
        diastolic and MAP blood pressures ...
      </entries>
    </compound>
  </entry>
</data-list>
```

Using these two types of entries every ISO/IEEE 11073 message is translated to a tree-like set of parents, children and siblings on one or more hierarchic levels [18]. Due to their length we are not discussing a whole actual *DataList* here . However, we will go further into detail on the different types of *DataLists* and their inner structure.

A recurring attribute for distinction of *DataLists* of the same kind is the *HANDLE*-entry in the meta-data. Previously, in Figure 3.14 we gave an example of a blood pressure monitor object instance. The main object *MDS* has two children *Systolic, Diastolic, MAP* and *Pulse*. Each of these three is a *DataList* on its own. However, they have a different *HANDLE*-number and thus can be identified.

ISO/IEEE 11073 message can have three purposes [18]:

- Configuration Data: It is necessary to identify the type of device and thus the kind of measurement data to be able to adapt to the specific layout of the MDS objects. Interpreting the configuration reveals this information.

- Measurement Data: In a nutshell, these data are what everything is about and contain the actual measurement data of the sensor. Nevertheless, configuration and device attributes are our tools to enable interpretation.

- Association Data and MDS Attributes: Right after the association a *DataList* with some basic information is sent. However, to gain detailed information about the device, such as the manufacturer or the model number, this MDS attributes have to be requested.

The XML documents containing the regarding *DataLists* are fed to the D-Bus whenever needed. The association data are published right after the association state is entered, the measurement data after received from the PHD and the device attributes if requested. *healthd* encodes and forwards the incoming data from the PHD using the D-Bus. Furthermore, this Antidote manager application handles requests of UniversAAL.

4.2 Gateway: ISO/IEEE 11073 to UniversAAL Platform

The main part of our programming effort was the development of a gateway to the UniversAAL platform. Basically, this gateway is an OSGi bundle running in UniversAAL which is connected to the D-Bus and the middleware at the same time. The D-Bus is the entry point for the ISO/IEEE 11073 vital data sent by Antidote's *healthd* manager application. In this context the gateway is an agent. This should not be confused with the PHDs which we called agents in the context of exchanging data with the CE. The managers and agents we discuss in this sections are both part of the CE's stack. Furthermore, the gateway publishes the gathered information on the Context Bus of the UniversAAL's middleware. In this section these two parts of the bundle will be investigated and we will discuss how they are connected to each other.

Read in Vital Data

Before we can process the vital data we have to read it in from the D-Bus. The D-Bus is a system for Linux IPC and is supported by all major programming languages including Java [18]. Even though we use Linux and, therefore, we do not have to make any modifications, the use of a different OS would not be an issue. Like the translation for Android, the basic idea is simple and could be easily translated to other platforms and other IPC frameworks [18]. Information is not sent to an application, but data are exchanged by routing messages to specific objects with a dedicated addresses. Regarding this matter of fact the two or more parts of communication do not interact directly with each other, but use these objects as their pigeon hole. The *healthd* D-Bus API enumerates three interfaces [18]:

- Manager Interface (hereinafter: *Manager*) - com.signove.health.manager
- Agent Interface (hereinafter: *Agent*) - com.signove.health.agent
- Device Interface (hereinafter: *Device*) - com.signove.health.device

We adapted this structure to our gateway. The package *com.signove.health* contains the three classes *Agent*, *Device* and *Manager* to read from the respective D-Bus object. Figure 4.3 reveals the class diagram of the package including its public methods.

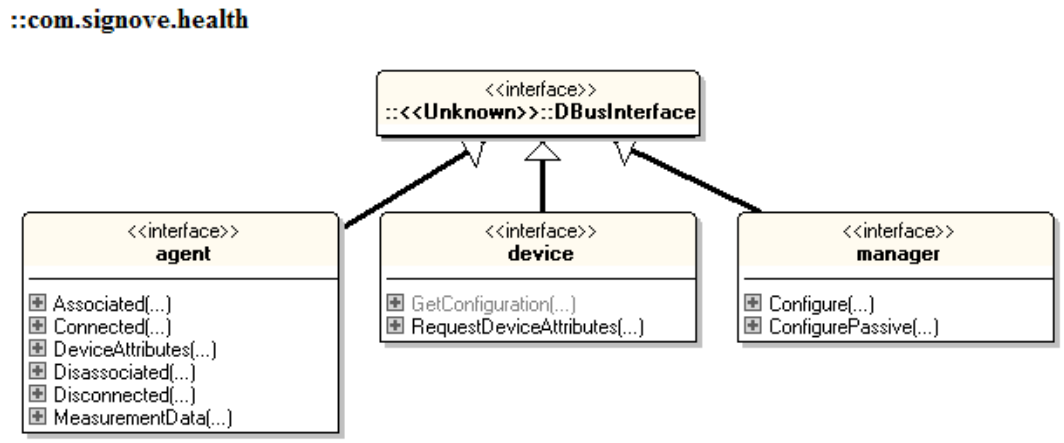


Figure 4.3: Class Diagram of com.signove.health
Source: Self-made Figure

The *Manager* implements the two methods *ConfigurePassive* and *Configure*. Table 4.2 lists them both and explains their purpose [18]:

void *ConfigurePassive*(object *Agent*, int data_types[]) Configured passively the health service reacts on connections from the *Agent* and is in listening mode. This is the configuration we use in our implementation. The *Agent* is another D-Bus object and *data_types* is an array containing the HDP types that should be accepted.
void *Configure*() *Configure* would make the health service initiate the connection with the agent. This Method is not yet implemented in *healthd*.

Table 4.2: Methods of the *Manager*

The *Device* object is of big importance for the *Agent* and is present in all of the methods we implemented. Every *Device* object is only a channel to an actual device and such a PHD possibly uses more than one channel to communicate [18]. Consequently one actual device possibly is represented by more than one D-Bus *Device* object. Such an object can be 'recycled' and reused if *healthd* associates with the same device again [18]. Table 4.3 lists the two methods we implemented in our class and explains their purpose [18].

void RequestDeviceAttributes() Whenever this method is called the *Agent* receives the attribute information via its method *DeviceAttributes*.
string GetConfiguration() *GetConfiguration* returns a string that contains the device's configuration encoded in XML. The configuration influences the structure of the MDS objects.

Table 4.3: Methods of the *Device*

As mentioned above this *Agent* should not be confused with the devices. The *Agent* is the object that interacts indirectly with the PHD. It receives events with the measured vital data and is a representation of the state machine of the *Communication Model* with its states *Connected* and *Associated*. Table 4.4 lists the methods and explains their purpose [18].

To contribute the data to UniversAAL the gateway is realized as an OSGi bundle within the platform. When such a bundle is started first the class *Activator* is executed. This *Activator* initializes the class *ISO11073DBusServer* as it is illustrated in Figure 4.4. This class is responsible for gathering the vital data. First we connect to the D-Bus. To do so we use libraries of the freedesktop.org project, an open source software project with the goal to improve interoperability for X Windows System desktops, such as GNOME or KDE. One output of this project is a set of libraries that facilitates accessing the D-Bus with Java.

void *Associated*(object *Device*, string xmldata) This method is called whenever the state *Associated* is entered. The *Device* contains the path of its object and is identified by its MAC address. The second parameter is the association APDU transformed in a XML stream.
void *Connected*(object *Device*, string address) Whenever a device connects this method is called and delivers the device object and the MAC address. Possibly there exist two or more D-Bus device objects for one actual PHD, because the term is a bit misleading and implements only a single data channel with the device. If more than one communication channel is established then this method is called several times and several device objects are generated for one actual PHD with one MAC address.
void *DeviceAttributes*(object *Device*, string xmldata) As a response to a request for the device's attributes this method is called and delivers a string containing these attributes encoded in XML. These attributes include e.g. the model of the device, its manufacturer, MAC address and configuration.
void *Disassociated*(object *Device*) Called when device enters the Disassociated state. This method has a special meaning in our gateway, because we interpret the call of this method as a signal for a finished measurement. So we start with processing the incoming data.
void *Disconnected*(object *Device*) Should usually be called right after the disassociation and reports the disconnection of a *Device* object.
void *MeasurementData*(object *Device*, string xmldata) This method is called back upon a measurement event report and the second parameter, *xmldata*, contains the MDS objects encoded in XML.

Table 4.4: Methods of the *Agent*

```
DBusConnection conn = null;
try {
        conn = DBusConnection.getConnection(DBusConnection.
            SYSTEM);
        System.out.println("DBusConnection␣conn:␣" +conn);
} catch (DBusException DBe) {
        System.out.println("Impossible␣doing␣D-Bus␣connection")
            ;
}
```

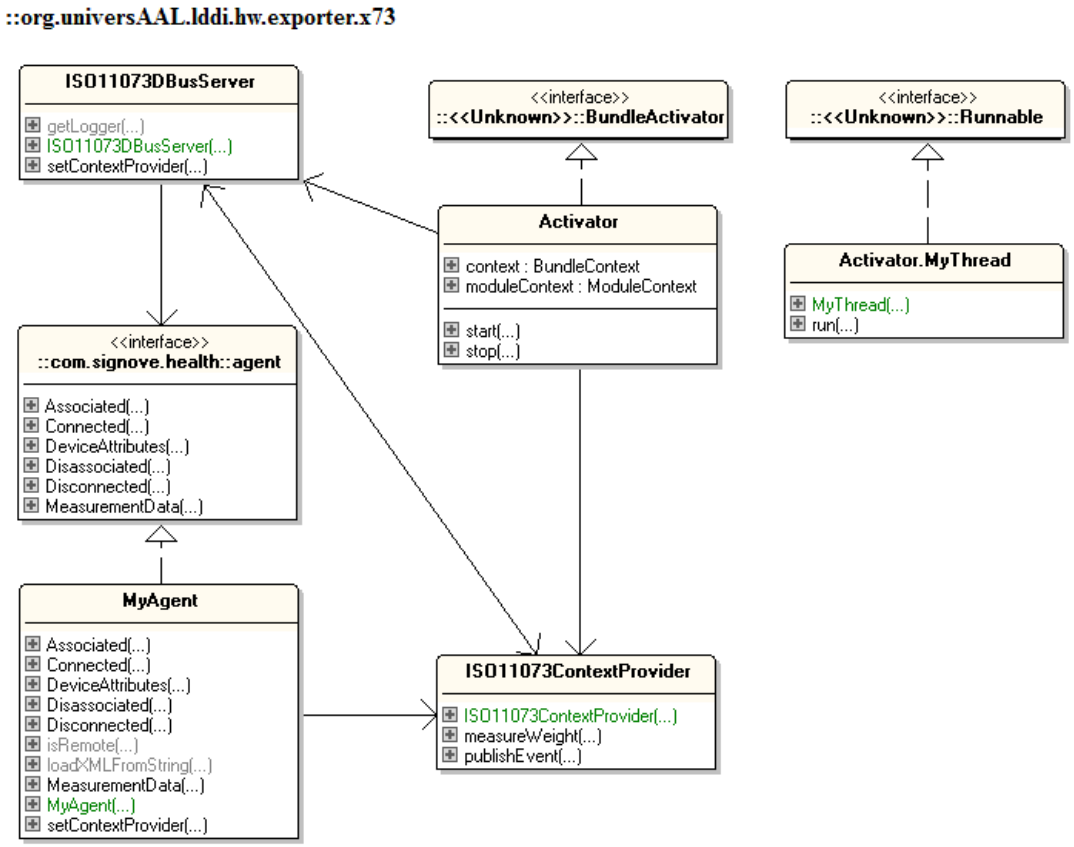


Figure 4.4: Class Diagram of org.universAAL.lddi.hw.exporter.x73
Source: Self-made Figure

The next step of *ISO11073DBusServer* is to create a remote object of the D-Bus object *Manager*.

```
manager remoteObject;
remoteObject =
(manager) conn.getRemoteObject("com.signove.health", "/com/
    signove/health", manager.class);
```

The class *MyAgent* implements the *Agent*, and gets a callback whenever data are incoming. This applies for measurement data as well as for the device attributes (after a request) or association data. Furthermore, the class implements methods to parse the strings to XML documents.

```
String agent_pid = "/com/signove/health/agent/" + ((int) (1 +
    Math.random() * 2000000000));
agt = new MyAgent(conn, contextProvider);
conn.exportObject(agent_pid, agt);
```

Then the manager is set to listening mode. It reacts on connections from the Agent without proactive acting. We do not provide any methods to use the *Object Access Services* of the ISO/IEEE 11073. Our implementation targets sensors that are configured using the device itself without any manipulations by the CE. They have to be turned on by the user to take measurements, which is alike the real-world workflow.

```
int data_types[] = {0x1004, 0x1007, 0x1029, 0x100f};
remoteObject.ConfigurePassive(agt, data_types);
```

Implemented are three different ISO/IEEE 11073 HDP data types:

- 0x1004: Pulse oximeter 11073-10404
- 0x1007: Blood pressure monitor 11073-10407
- 0x100f: Body weight scale 11073-10415

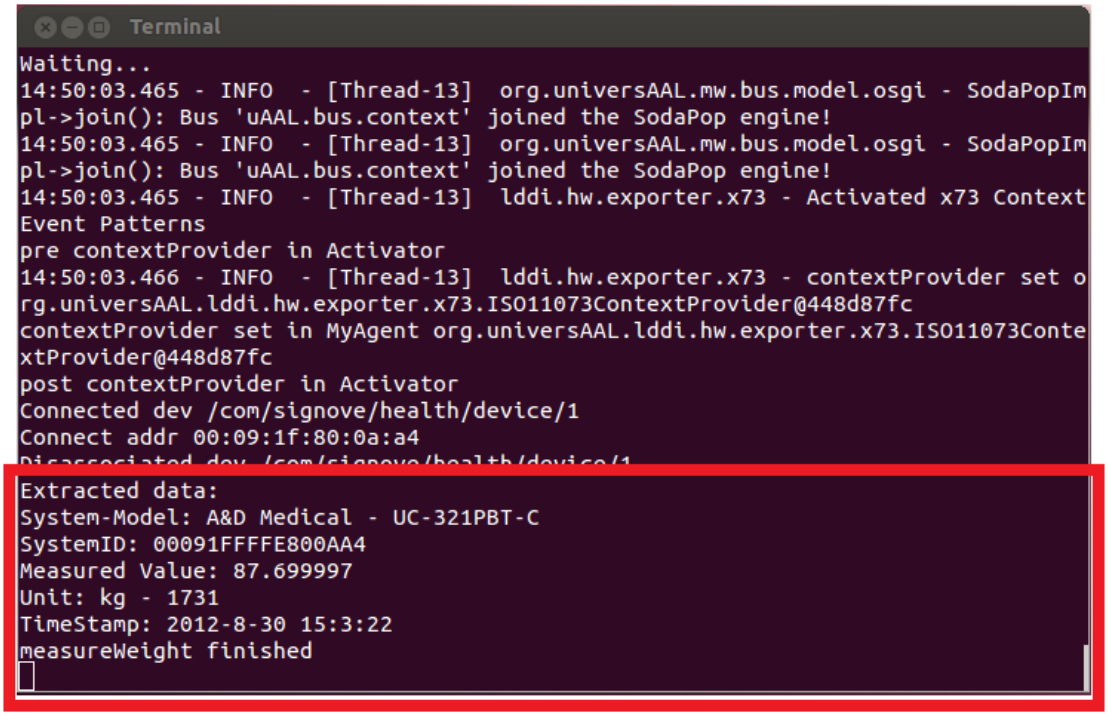


Figure 4.5: *MyAgent* receives Vital Data of a Measurement

We use as evaluation devices a blood pressure monitor and a body scale. For other types of PHDs than the three that are enumerated above it is necessary to add a plug-in for the device specification to Antidote. In Figure 4.5 the agent receives the data of a measurement. The system model is a A&D Medical UC-321PBT-C which is a body scale. The output is only a part

of the received data and consists of information retrieved from the device attributes and of the measurement data set.

After the initialisation of the *ISO11073DBusServer*, the *Activator* starts the thread *MyThread* which runs the *ISO11073ContextProvider* in order to publish received data (e.g. measured vital data) on the *Context Bus*.

Feed the Vital Data to the Context Bus

To contribute incoming data to other AAL services running on the UniversAAL platform the class *ISO11073ContextProvider* publishes it on the *Context Bus* of UniversAAL's middleware. All services that subscribe this kind of events will receive the measured data wrapped as a *Context Event*. Therefore, the data has to be in RDF format.

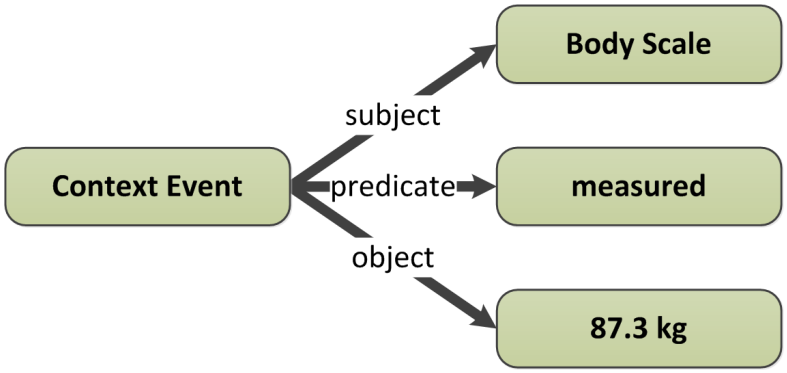


Figure 4.6: RDF Triple of a Body Scale
Source: Self-made Figure

Figure 4.6 depicts an example for such an RDF triple. In this example we measure 87.3 kg with a body scale. Therefore, the subject is the *body scale*, *measured* is the predicate and *87.3 kg* is our object. It can be read like the sentence 'The body scale measured 87.3 kg'. For every piece of data a property has to be changed. Hence, there are numerous *Context Events* for every measurement, including the system ID, the device model, and the unit code. Different device models produce different data. Already implemented are methods to handle the A&D UC-321PBT-C bathing scale and the A&D UA-767PBT-C blood pressure monitor. For other models with the same handling the model description has to be added to the if-clause in the Disassociated-method of *MyAgent*. To deal with completely new devices or adapt their treatment a publishing method has to be added to the class *ISO11073ContextProvider*. This method publishes the properties of a specific device. Following lines publish the properties of the ontology class WeighingScale:

```
WeighingScale ws = new WeighingScale(WeighingScale.MY_URI +
    systemId);
ws.setSystemId(systemId);
```

```
cp.publish(new ContextEvent(ws, WeighingScale.PROP_SYSTEM_ID));
ws.setSystemTypeSpecList(typeSpecList);
cp.publish(new ContextEvent(ws, WeighingScale.
    PROP_SYSTEM_TYPE_SPEC_LIST));
ws.setSystemModel(sm);
cp.publish(new ContextEvent(ws, WeighingScale.PROP_SYSTEM_MODEL
    ));
ws.setHasMeasuredWeight(bw);
cp.publish(new ContextEvent(ws, WeighingScale.
    PROP_HAS_MEASURED_WEIGHT));
```

The properties PROP_SYSTEM_ID and PROP_SYSTEM_TYPE_SPEC_LIST are datatype properties, while PROP_SYSTEM_MODEL is an object property linking to the ontology class *SystemModel* and PROP_HAS_MEASURED_WEIGHT to *BodyWeight*. The setter-method changes the property.

```
public void setHasMeasuredWeight(BodyWeight newPropValue) {
  if (newPropValue != null)
    changeProperty(PROP_HAS_MEASURED_WEIGHT, newPropValue);
}
```

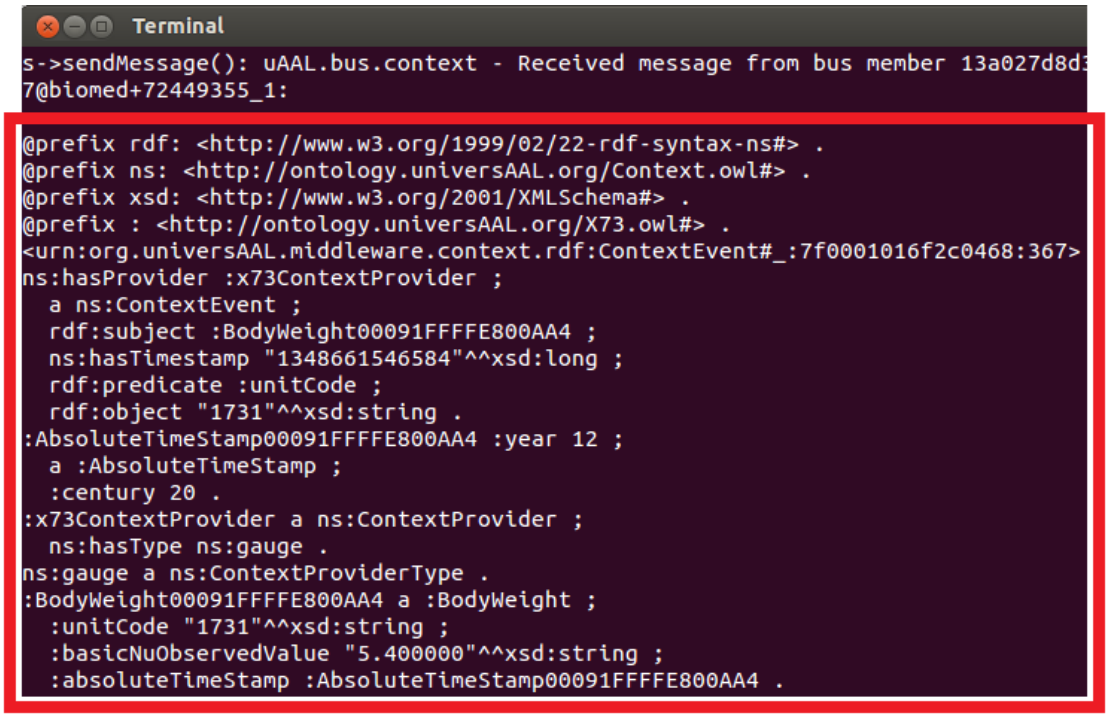


Figure 4.7: Measurement Data are published on the *Context Bus*

Figure 4.7 illustrates the *Context Event* of this measurement published on the *Context Bus*. The depicted output is only a *Context Event* that is feed to the bus after measuring the weight and it provides the unit code. The first lines after the prefix are the RDF triple with *BodyWeight* as subject, *unitCode* as predicate and the value '*1731*' as object. This number is the code for 'kg'.

To access the source code it is necessary to have a login of the UniversAAL forge and the permissions to enter the *Local Device Discovery and Integration (LDDI)* project. The gateway code is open source and can be found at

HTTP://FORGE.UNIVERSAAL.ORG/GF/PROJECT/LDDI/SCMSVN/?ACTION=BROWSE
&PATH=%2FTRUNK%2F

and on the Apache Subversion (SVN) in the folder /trunk/lddi.hw.exporter.x73 of the *LDDI* project.

4.3 UniversAAL Middleware and Ontology Integration

The following section reveals the structure that we use to depict vital data in UniversAAL. Its middleware is the broker and provides the *Context Bus* where measurements are published as *Context Events*. They refer to an ontology and only with knowledge about the relations and attributes it is possible to extract the actual information from these messages. The ontology is an AAL service and is deployed like any other service as an OSGi bundle in UniversAAL and typically implements other delivered basic ontologies.

The name of our ontology is X73 and is our representation of a measurement. An ontology module for UniversAAL consists of four parts [28]:

- Module Activator: As the ontology is an OSGi bundle we need a module activator. Such a module possibly implements a number of ontologies. However, we need only one for the measurement data we receive. The activator has a couple of constraints:
  - It should be in the package *org.universAAL.ontology*.
  - The activator must create, register, and unregister the ontology in the middleware.
  - The class name has to end with 'Activator' the ontology's name should be put in front (e.g. X73Activator).
  - The interface *org.universAAL.middleware.container.uAALModuleActivator* must be implemented.
- Factory: The factory's purpose is to instantiate non-abstract ontologies.
  - The factory should be in the package *org.universAAL.ontology*.
  - It should be named '<ontologyname>Factory' with the ontology's name in front (e.g. X73Factory).

  - It has to extend *org.universAAL.middleware.rdf.impl.ResourceFactoryImpl*.

- Ontology Definition: The definition is used for the creation of the ontology and defines the properties, relationships and restrictions and defines all information about the ontology.
  - The ontology definition should be in the package *org.universAAL.ontology.<ontologyname)* (e.g. org.universAAL.ontology.X73Activator).
  - Its name should be the ontology's name and 'Ontology' in the end (e.g. X73Ontology)
  - The class must extend *org.universAAL.middleware.owl.Ontology*.
  - It has to define the namespace like 'http://ontology.universaal.org/<ontologyname>.owl#' (e.g. http://ontology.universaal.org/X73.owl#).
  - The definition must overwrite the method *create()*.

- Set of Ontology Classes: Every class in the ontology extends another ontology. The top level classes of an ontology implement interface classes *Service* or *ManagedIndividual* which are delivered by UniversAAL.
  - Each class has to extend another UniversAAL ontology.
  - They should define the Uniform Resource Identifier (URI) which starts with the ontology namespace and puts the name of the particular class in the end (e.g. X73Ontology.NAMESPACE + 'BloodPressureMonitor').
  - They must overwrite the method *getClassURI()*.

Some of these constraints are only for a common layout and are not influencing the syntactical and semantic correctness of the ontology module. Nevertheless, it is prudent to stick to this guidelines due to better legibility.

The relationship of our set of ontology classes is illustrated in Figure 4.8. On the left upper side of the figure we have the ontology definition with the namespace of the ontology. Furthermore, the definition calls the factory in order to instantiate the ontology.

Additionally, we have the class *X73* which extends the ontology *Service* and has a property named *PROP_CONTROLS* which connects it logically with the *MDS* class.

On the other hand *MDS* is an abstract ontology class that has to be extended by all classes representing PHDs. It has some basic properties that can be found in every device, such as an ID and a model number.

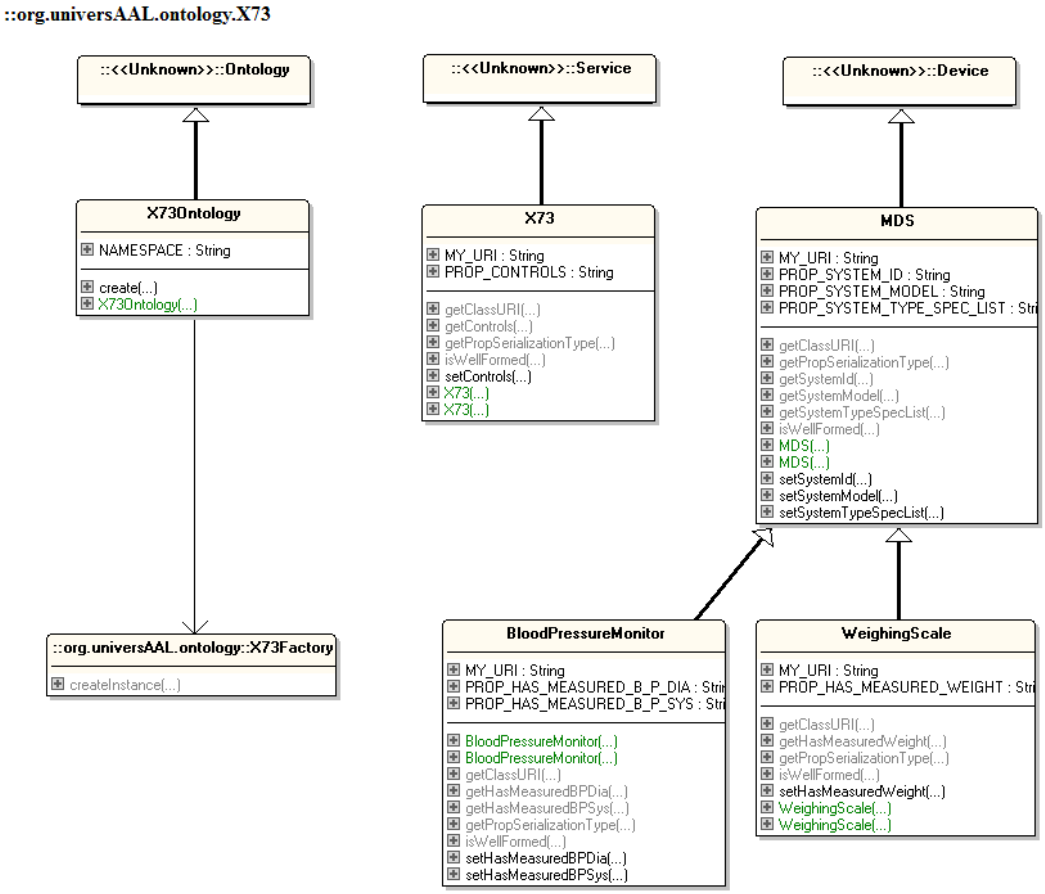


Figure 4.8: Class Diagram of org.universAAL.ontology.X73 - Basic Structure
Source: Self-made Figure

Our evaluation devices are represented by the classes *BloodPressureMonitor* and *WeighingScale*. They adapt the respective special cases and attributes of the incoming ISO/IEEE 11073 messages. They contain not only datatype properties, but object properties as well. These object properties are other ontology classes and are displayed in Figure 4.9.

There we have the class MDSAttribute as a counterpart of MDS. It implements properties every attribute and every measurement has, such as a timestamp, a unit and the measuring. The measuring can be either a single value (*PROP_BASIC_NU_OBSERVED_VALUE*) or a set of values (*PROP_COMPOUND_BASIC_NU_OBSERVED_VALUE*).

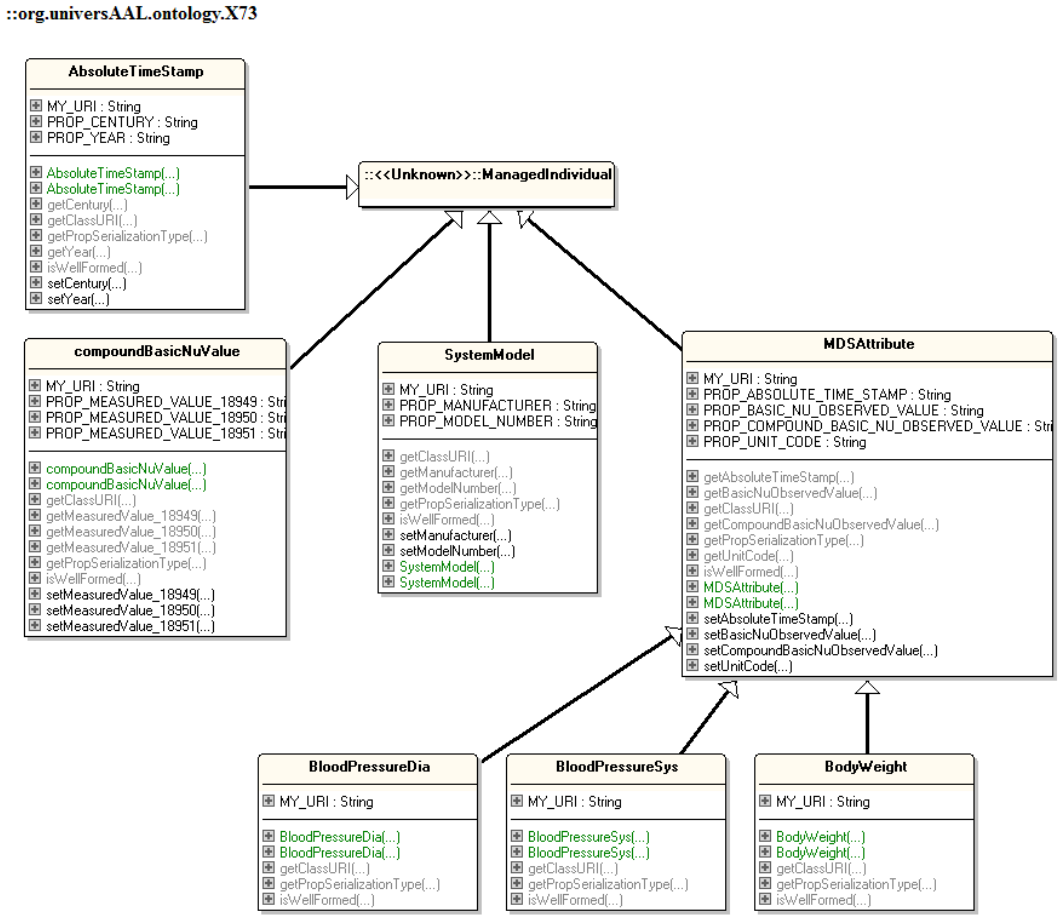


Figure 4.9: Class Diagram of org.universAAL.ontology.X73 - Object Properties
Source: Self-made Figure

Like before the source code can be accessed via the UniversAAL forge. The ontology is open source and can be found at

HTTP://FORGE.UNIVERSAAL.ORG/GF/PROJECT/LDDI/SCMSVN/?ACTION=BROWSE
&PATH=%2FSANDBOXES%2FPATRICKSTERN%2FORG.UNIVERSAAL.ONTOLOGY.X73%2F

and on the SVN in the folder /sandboxes/patrickstern/org.universAAL.ontology.x73 of the *LDDI* project.

CHAPTER 5

Conclusion and Future Prospects

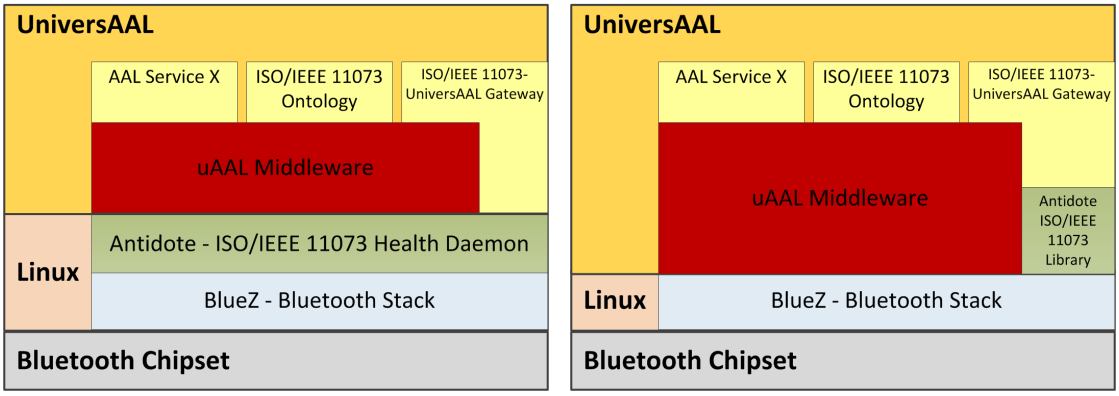
LIFE expectancy rises continuously and the consequence is a worldwide trend of an ageing population [57]. This development affects the balance between older citizens and their caregivers [41]. AAL attends to this shift by using technological aid in order to facilitate older people to handle daily life activities in an independent way and support caretakers in their job [16].

AAL is a young research topic with a diverse field of contributors [16]. Interoperability and compatibility of PHDs of different manufacturers is a major issue of the branch [17]. The Continua Health Alliance is an industry organisation of healthcare and technology companies with the goal to tackle this problem and improve interoperability with the use of existing standards (e.g. the ISO/IEEE 11073) [14]. The UniversAAL project aims to develop an open AAL platform for the purpose of making it technically realisable for software engineers to design, code and field an AAL solution with reasonable costs [68].

We pursued the goal to provide a reference implementation of a standardized connection between PHDs to an open AAL platform. More precisely, we managed to send measurement data of a Continua-certified bathing scale and a Continua-certified blood pressure monitor to the UniversAAL AAL platform using the ISO/IEEE 11073 standards. Therefore, we developed a gateway service that runs on the platform and reads in the vital information and processes it.

We use Bluetooth to transfer the data from the PHD to the CE. In the second version of the Continua Health Alliance design guidelines BLE and ZigBee are selected to be the new flagship protocols [50]. Due to their advantageous lower power consumption [48] [59] these protocols will be used increasingly in the future. However, the current implementation cannot handle them. Consequently, an update is advisable. Therefore, a plug-in for the Antidote has to be developed [18]. Antidote is an ISO/IEEE 11073-20601 stack and is used to retrieve the data sent via Bluetooth [6]. It uses the BlueZ Bluetooth stack for Linux which is open source and implements the Bluetooth HDP [6]. For Windows Toshiba's Bluetooth stack implements

the HDP. However, it is not open source. We need this profile and in 2012 Linux and Android are the only suitable OS. Antidote runs on Android as well [18]. Therefore, once the UniversAAL platform runs on this OS the gateway can be used out-of-the-box. Nevertheless, this is a theoretical statement and was not tested in the course of this master thesis. Our partners in Spain, TSB Tecnologias, are working on a solution running on Linux and Windows systems. In order to adapt our implementation to Windows it would be necessary to reimplement the services provided by Antidote. Furthermore, the implementation would be more independent if



(a) Realized Stack using the Antidote Health Daemon

(b) Possible Stack using the Antidote Library

Figure 5.1: Implementation Stack
Source: Self-made Figure

Antidote would be used as a library instead of using the Antidote health daemon. Figure 5.1a depicts the stack of our implementation while Figure 5.1b contrasts the architectural change of including the routines of the daemon in the gateway and use only the Antidote library instead. Currently, the daemon feeds the vital information encoded in XML to the IPC bus D-Bus [18]. The gateway service reads in the data and transforms the information into *Context Events*. These *Context Events* are RDF triples for every piece of information [28]. A single measurement with the bathing scale produces not only a message for the body weight. Moreover, there are *Context Events* representing the PHD's model number, its unique address, the manufacturer, the timestamp and more. The reason is the ontology-based bus system of UniversAAL's middleware [28]. All other AAL services running on the UniversAAL platform can use a *Context Event* pattern and subscribe to a specific kind of such messages [28]. Using a pattern matching with our messages other services can filter measurement data of one or more PHDs [28].

The developed implementation is a proof of concept and is useful to guide the first steps of other software developers with the ambition to exchange data between PHDs and CEs. Both, ontology and gateway, are designed only for a rudimentary use of a specific device model of a bathing scale and a blood pressure monitor. Its purpose is to help in understanding and provide a explanatory implementation and a template for future developers.

APPENDIX A

Installation Guide

THE outcome of this thesis is an interface between PHDs and CEs. The implementation facilitates the UniversAAL community to develop AAL solutions using ISO/IEEE 11073 conform devices. In order to support future users in reproducing our results Appendix A documents the used environment and explains how to set up such an interface from the scratch.

A.1 System Properties

First of all we have to put the system properties of the evaluation system on record. Table A.1 demonstrates the characteristics of the CE that is the sink in our example. A sink is the receiver in Bluetooth communication. The desktop computer we use has a dual core processor with a clocking of 2.53 GHz each. It has a 64-bit architecture, but this is not part of the system requirements. Even though we used a 64-bit system for evaluation the implementation would run on systems with a 32-bit architecture as well.

Central Processing Unit	Intel Xeon CPU W35
	2.53 GHz x 2
	64 Bit
Working Memory	5.8 GB
Hard Disk	243.3 GB

Table A.1: Evaluation System's Properties

The additional installation files are less than 20 MByte in size and is negligible small, compared to the disc space that is required by the OS and the Eclipse Integrated Development Environment (IDE). The used OS is the 64-bit version of Ubuntu 12.04, as indicated by Table A.2.

Ubuntu can be downloaded at

HTTP://WWW.UBUNTU.COM/DOWNLOAD/DESKTOP

BlueZ is the default Bluetooth Stack in Linux and supports the Bluetooth HDP. The used version 4.98 is included when installing Ubuntu 12.04.

Operating System	Ubuntu 12.04 LTS
	64 Bit
Kernel	3.2.0-26-generic x86_64

Table A.2: Evaluation System's Operating System

A.2 Setup the Evaluation System

Once the OS is running the system environment has to be set up. The first step is to fulfil Antidote's requirements and get it running. The dependencies of the Antidote library are listed in Table A.3. This table is valid for Antidote 2.1, which is the most recent version in July 2012. Whenever we make use of the term *Antidote* in this chapter we talk specifically about version 2.1.

Software	Minimal Requirement	Evaluated Version
BlueZ Bluetooth Stack	4.80 or better (because this version has mature HDP support)	4.98-2ubuntu7
D-Bus	1.4.0 or better (because HDP API uses file descriptor passing, a novel feature of D-Bus)	1.4.18-1ubuntu1
Linux Kernel	2.6.36 or better (because of ERTM, a Bluetooth feature that HDP demands)	3.2.0-26-generic x86_64

Table A.3: Dependencies of Antidote 2.1

We use Ubuntu 12.04 which automatically installs an adequate version of BlueZ and D-Bus. Nevertheless, some packages are missing. Following packages have to be installed by using Synaptic or apt-get (access authorization of root is necessary):

- automake (1:1.11.1-1ubuntu1)
- dbus-1-dbg (1.4.14-1ubuntu1)
- libdbus-1-dev (1.4.14-1ubuntu1)

- libdbus-glib-1-dev (0.94-4)
- libltdl-dev (2.4-2ubuntu1)
- libusb-1.0-0-dev (2:1.0.8-4)
- build-essential

There are no remaining dependencies anymore and the system is ready for the installation of Antidote. The most recent version of Antidote can be downloaded at

HTTPS://GITORIOUS.ORG/ANTIDOTE/ANTIDOTE

On Antidote's project website current information, older releases, samples and a documentation document can be found at

HTTP://OSS.SIGNOVE.COM/INDEX.PHP/ANTIDOTE:_IEEE_11073-20601_STACK

To install the library following commands have to be executed in Antidote's main folder. For additional information about the installation and individual configuration read the *INSTALL* file that appears after execution of *autogen.sh*.

```
./autogen.sh
sudo ./configure
sudo make
sudo make install
sudo cp apps/resources/healthd.conf /etc/dbus-1/system.d
sudo service dbus reload
```

Continua decided to use Bluetooth as its flagship transport protocol. In order to communicate with Contina-certified PHDs it is necessary to use a CE that supports this transport protocol. Dependent from your system additional steps for the installation of a Bluetooth chipset may have to be taken. We used the *hama Bluetooth USB-Adapter Nano*. For Ubuntu it is plug-and-play and there is no need for additional steps for the installation. *Blueman* is software that is not needed necessarily for dealing with Bluetooth devices in Linux, but perhaps it could be useful.

In order to verify that your Bluetooth chipset is working fine and BlueZ is running the command

```
hciconfig
```

should find *hci0* like it is pictured in Figure A.1.

If Antidote is installed and the system is capable of establishing Bluetooth connections it is possible to run first tests to check if Antidote is working properly. Details are described in the first two subsections of the subsequent section. However, to get the Java interface running it is necessary to install *Matthew's Java libraries* in order to exploit access to the D-Bus for Java.

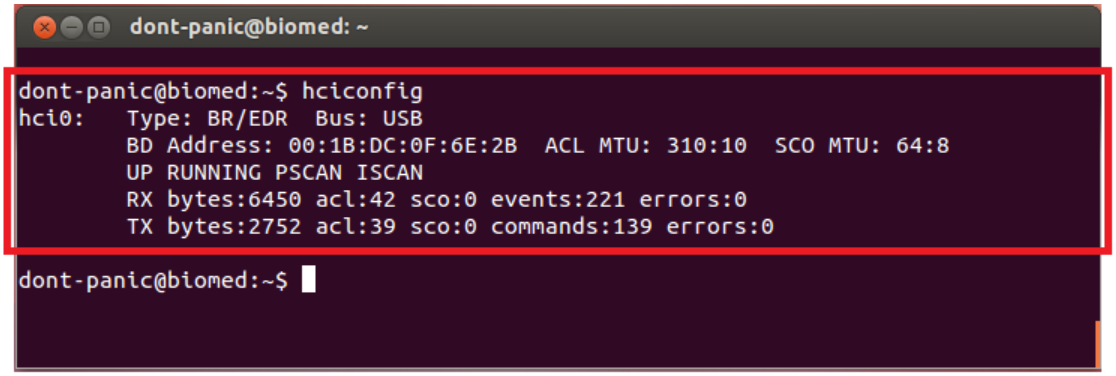


Figure A.1: Test Bluetooth chipset

Therefore, the Java Development Kit (JDK) has to be installed. It is not running with JDK 7 so we used OpenJDK 1.6.0_24. To check the installed Java version enter

```
java -version
```

in the terminal. Furthermore, it is mandatory to set the variable *JAVA_HOME*. To check if it is set enter

```
echo $JAVA_HOME
```

If a path, such as */usr/lib/jvm/java-6-openjdk-amd64* is printed on the screen the variable is set and the upcoming lines can be skipped. If the output is an empty line the following code has to be executed. The path has to be adapted following local conditions.

```
JAVA_HOME=/usr/lib/jvm/<jdk-directory>
export JAVA_HOME
PATH=$PATH:$JAVA_HOME/bin
export PATH
```

Once JDK is running we can go further to install *Matthew's Java libraries*. They can be found at his website

HTTP://WWW.MATTHEW.ATH.CX/PROJECTS/JAVA/

The installation of the library is quite straight-forward. It is only necessary to run

```
make
make install
```

The last issue is to add the D-Bus library to the Java build path of the Java project. We presuppose that Eclipse 3.7.2 and the UniversAAL platform are working fine and running. A

detailed installation description can be found in the first three chapters of the UniversAAL Developers Handbook. The guide is online available at

HTTP://WWW.UNIVERSAAL.ORG/IMAGES/STORIES/DELIVERABLES/D2.4-B.PDF

The necessary libdbus-java-2.7.jar has been developed by the freedesktop.org-project and can be found at their website

HTTP://FREEDESKTOP.ORG/

After this final step the system will receive the vital data from the PHD via the Bluetooth HDP. Antidote's manager software handles the incoming data and puts them on the D-Bus such that the Java application can read it in and process it further.

Our developments can be found in the *UniversAAL forge* in the LDDI project. The source code is open source and the project's homepage is

FORGE.UNIVERSAAL.ORG/GF/PROJECT/LDDI

A.3 Run the System

In this section we want to describe the recurring process of running the system and taking measurements. This starts with pairing, goes further with testing the Antidote environment and finally measuring vital data.

Pairing PHD and CE

First of all we have to connect the PHD and the CE. Using Bluetooth terminology this is called pairing. We make use of two evaluation devices that are listed in Table A.4. On the one hand we use a blood pressure monitor and on the other hand a body scale. Both of them by the manufacturer *A&D Medical* (hereinafter: *A&D*).

Purpose	Manufacturer	Device
Body Scale	A&D Medical	UC-321PBT-C Precision Scale
Blood Pressure Monitor	A&D Medical	UA-767 Plus BT-C

Table A.4: Evaluation Devices

Due to some troubles with PHDs by this manufacturer we have to pair the devices every time after a reboot. The first step is to delete the existing connection if present. Therefore, use the BlueZ device manager. Then the SSP mode has to be disabled.

```
sudo hciconfig hci0 sspmode 0
```

The next step is the pairing itself. A&D devices are visible for two minutes after a reset. To reset the device the batteries have to be removed temporarily for some seconds. The Bluetooth device manager is not capable of an individual PIN like these devices have. The PIN is 123456. Hence the terminal has to be used for pairing.

```
bluez-simple-agent hci0 00:09:1F:80:02:7D #blood pressure
```

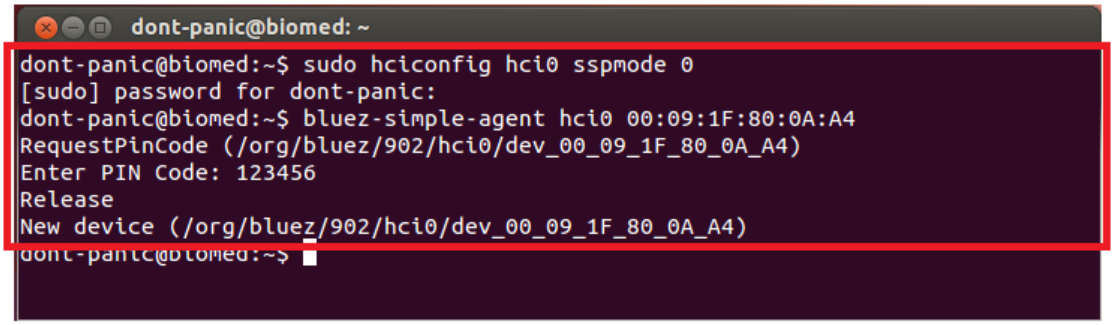


Figure A.2: Pair a PHD

Figure A.2 illustrates the pairing process. First we disable the SSP mode and afterwards we pair the device and enter the PIN. After this procedure the PHD is paired and ready for operation.

Test the Antidote Library

Antidote provides a manager called *healthd* and test scripts as well. To check if the system is running and eliminate sources of error we run this test first to ensure that the foundation of our Java application is stable and running. Therefore, we need two terminals and run the following commands in either one of them. The two scripts can be found in the *apps* directory in the Antidote main folder.

```
sudo ./healthd
sudo ./test_healthd.py --mds
```

The output of this test is demonstrated in Figure A.3. In the right terminal where the agent was running we see the incoming vital data from the body scale. The data are structured in XML and the option *–mds* of the test script causes that this xml data are saved to files as well.

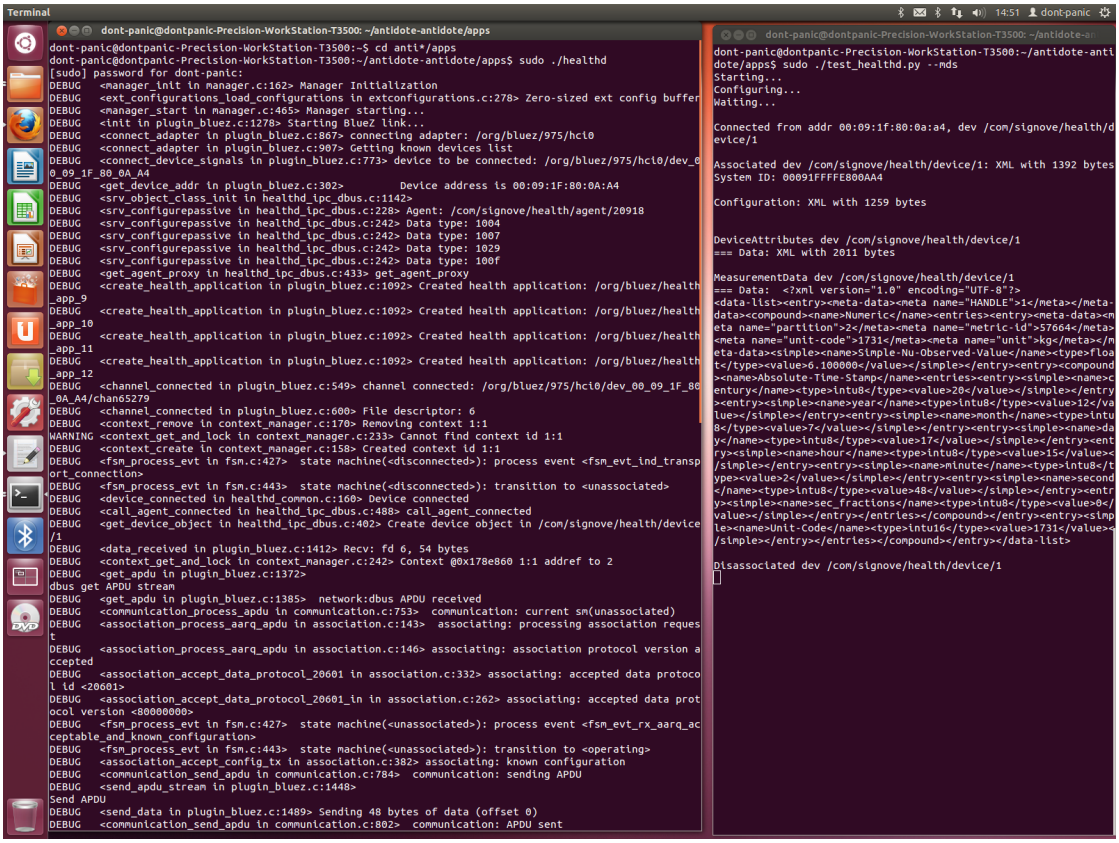


Figure A.3: Run the Python test script of Antidote

Measure Data with the Java Agent

Finally we take measurements and import it to the Java application. Therefore, *healthd* is started like in the test of the previous subsection. This installation guide assumes that the UniversAAL platform is already running on the system, and the bundles of the gateway and the X73 ontology are embedded and started. The platform is started by executing run.sh in the launches-folder of the gateway. The command lb lists the running bundles. Figure A.4 is a screenshot of the command's output.

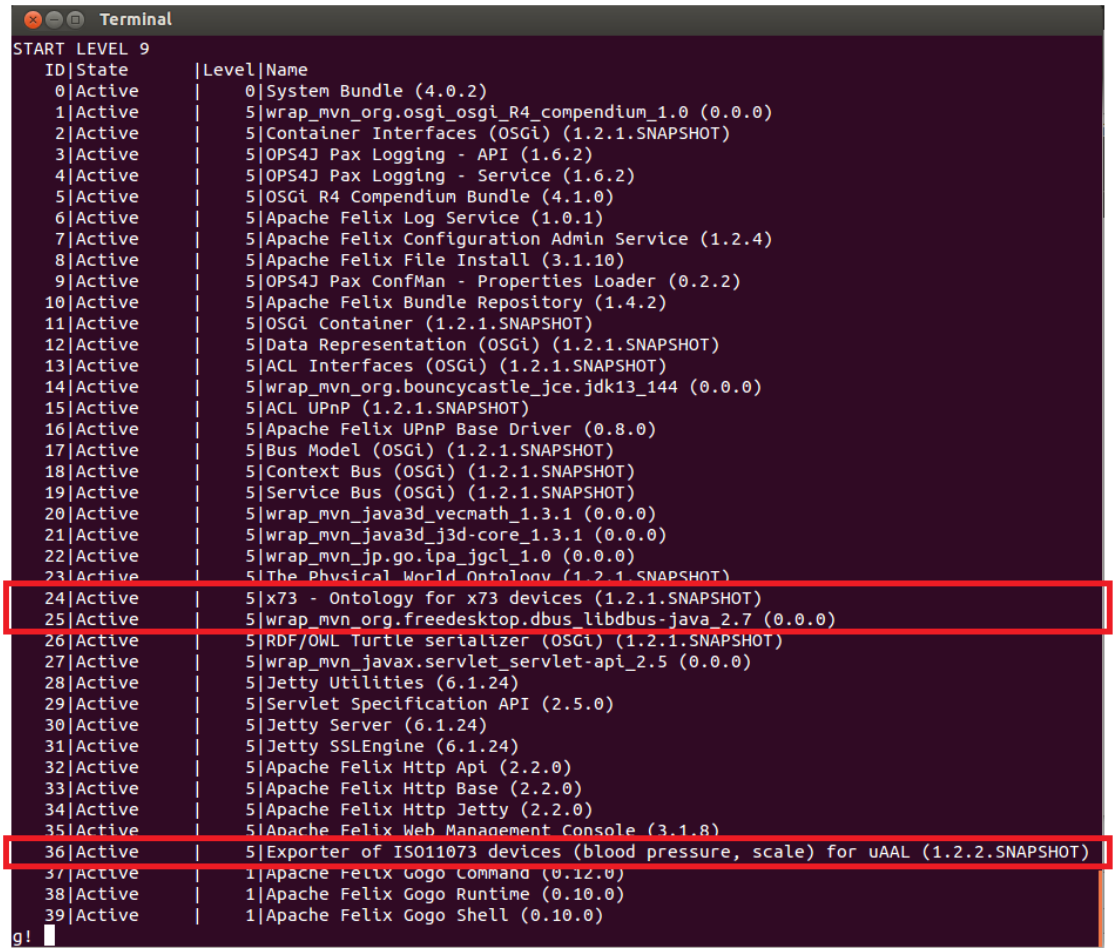


Figure A.4: The Bundles in our OSGi Framework

After this step the framework is running and waiting PHDs to connect and send data. Figure A.5 displays the output of the gateway after such a measurement.

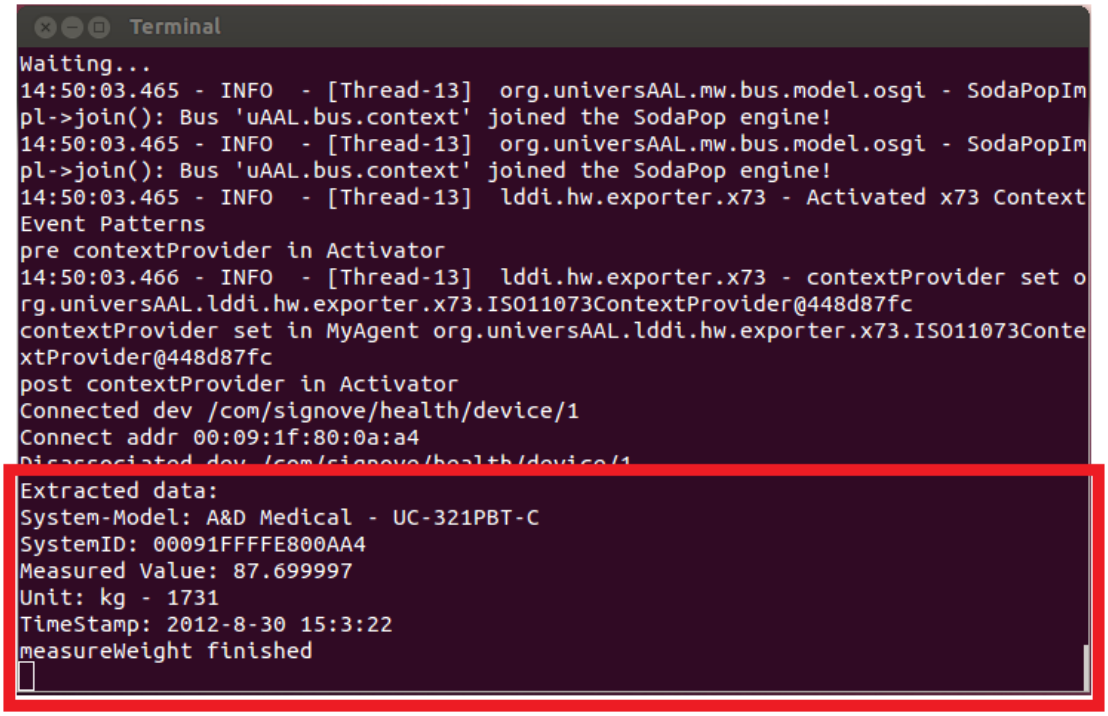


Figure A.5: Measurement Data displayed in OSGi Framework

The output of the Java implementation in Figure A.5 is an extraction of information from the XML-data of the Python test script. Like in the test script the data are saved in XML-files in the folder of the Java project. Even more important is that other AAL services that run on the UniversAAL platform are enabled to use this data because it is feed to the *Context Bus* of UniversAAL's middleware as we have depicted it earlier in Figure 4.7.

Glossary

AAL Ambient Assisted Living. 1, 6, 8–18, 20–24, 27–35, 42, 49, 52, 55–61, 66, 75–77, 81, 90, 92, 97–99, 107, 114, 115

AES Advanced Encryption Standard. 48

AI Artificial Intelligence. 21

AIT Austrian Institute of Technology. 24, 26, 34

AmI Ambient Intelligence. 21, 22, 25

ANSI American National Standards Institute. 68

APDU Application Protocol Data Unit. 74, 79, 87

API Application Programming Interface. 34, 49, 51, 53, 61, 81, 85

ARC Austrian Research Centers. 24

ASN.1 Abstract Syntax Notation One. 53, 73, 79, 81, 82

BAN Body Area Network. 8, 9, 36, 42, 48, 52

BLE Bluetooth Low Energy. 37, 40, 45–48, 53, 67, 79, 97, 117

CE Computational Engine. 9, 14, 15, 17, 27, 36, 44, 45, 49, 52, 54, 64, 69, 74, 76, 77, 84, 89, 97–99, 101, 103, 113

CRC Cyclic Redundancy Check. 48

D-Bus Desktop Bus. 54, 77, 81, 82, 84–88, 98, 100, 101, 103

DICOM Digital Imaging and Communications in Medicine. 69

DIM Domain Information Model. 50, 53, 72–74, 79, 82

DIN Deutsches Institut fur Normung. 68

DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik. 9

DoW Description of Work. 56, 57

E&AR Experience and Application Research. 19

EU European Union. 11

FLOSS Free/Libre Open Source Software. 52

FTP File Transfer Protocol. 38

GSyC Grupo de Sistemas y Comunicaciones. 52

HAN Home Area Network. 8, 9, 36

HDP Health Device Profile. 14, 38, 39, 48, 49, 52–54, 76, 81, 86, 89, 97, 98, 100, 103

HL7 Health Level 7. 50, 69

ICT Information and Communication Technology. 8, 21, 24, 29, 36

IDE Integrated Development Environment. 99

IEEE Institute of Electrical and Electronics Engineers. 13–16, 18, 37–40, 42, 45, 49, 50, 53–55, 61, 67–70, 72–81, 83, 84, 89, 94, 97, 99, 115

IEEE-SA IEEE Standards Association. 68

IHE Integrating the Healthcare Enterprise. 50, 52

IP Internet Protocol. 51

IPC Inter-Process Communication. 54, 85, 98

IrDA Infrared Data Association. 36, 37

ISM Industrial, Scientific and Medical radio frequency band. 37, 40, 42, 44, 45

ISO International Organization for Standardization. 13–16, 18, 38–40, 42, 49, 50, 53–55, 61, 67–70, 72–81, 83, 84, 89, 94, 97, 99, 115

IT Information Technology. 68

JDK Java Development Kit. 102

JNI Java Native Interface. 54, 61

JRE Java Runtime Environment. 61

JSON JavaScript Object Notation. 79, 82, 83

JVM Java Virtual Machine. 61

L2CAP Logical Link and Adaptation Protocol. 39

LAN Local Area Network. 68

LDDI Local Device Discovery and Integration. 92, 95, 103

LGPL GNU Lesser General Public License. 54, 81

LIN Lamprey Networks Inc.. 49

MAC address Media Access Control address. 40, 87

MAP Mean Arterial Pressure. 73

MCAP Multi-Channel Adaptation Protocol. 39

MCU Microcontroller Unit. 42

MDER Medical Device Encoding Rules. 50, 53, 73

MDS Medical Device System. 73, 79, 84, 86, 87

NDK Native Development Kit. 81

NSA National Security Agency. 48

OASIS Open Architecture for Accessible Services Integration and Standardisation. 20, 21, 23, 26, 34, 56

OS Operating System. 53, 54, 61, 76, 79, 81, 85, 98–100

OSGi Open Services Gateway initiative. 15, 51, 52, 54, 59–61, 77, 84, 86, 92

OSI Open Systems Interconnection. 40

OWL Web Ontology Language. 63

PERSONA PERceptive Spaces prOmoting iNdependent Aging within dynamic ad-hoc Device Ensemble. 34, 56, 57

PHD Personal Health Device. 9, 11, 13–15, 17, 18, 27, 34, 36, 39, 42, 48–53, 64, 65, 67–69, 72–77, 81, 82, 84, 86, 87, 89, 93, 97–99, 101, 103, 104, 107, 113

PHMR Personal Healthcare Monitoring Report. 50

R&D Research and Development. 19

RDF Resource Description Framework. 63, 66, 90, 98

SDP Service Discovery Protocol. 39

SME Small and Medium Enterprise. 20

SOA Service Oriented Architecture. 18, 50

SODA Service Oriented Device Architecture. 50, 51

SOPRANO Service-oriented Programmable Smart Environments for Older People. 18–20, 26, 34, 56

SVN Apache Subversion. 92, 95

TCP/IP Transmission Control Protocol/Internet Protocol. 54, 79

UI User Interface. 61

UniversAAL UNIVERsal open platform and reference Specification for Ambient Assisted Living. 14, 16, 17, 26–29, 31–35, 55–58, 61, 63, 66, 75–77, 81, 84, 86, 90, 92, 93, 95, 97–99, 102, 103, 105, 107, 114, 115

URI Uniform Resource Identifier. 93

USB Universal Serial Bus. 36, 42, 51, 67, 79

WAN Wide Area Network. 50, 54

WPAN Wireless Personal Area Network. 36, 37, 42, 45, 48

WSN Wireless Sensor Network. 15, 42, 44, 48

XML Extensible Markup Language. 79, 82–84, 86–88, 98, 107

List of Figures

List of Tables

Bibliography

[1] Gartner Newsroom 2011. Gartner says worldwide smartphone sales soared in fourth quarter of 2011 with 47 percent growth. http://www.gartner.com/it/page.jsp?id=1924314. Accessed: 2012-02-14.

[2] Continua Health Alliance. Continua health alliance. http://www.continuaalliance.org. Accessed: 2012-08-22.

[3] ZigBee Alliance. *ZigBee Wireless Sensor Applications for Health, Wellness and Fitness*. ZigBee Alliance, 2009. https://docs.zigbee.org/zigbee-docs/dcn/09-4962.pdf.

[4] ZigBee Alliance. *ZigBee and Wireless Frequency Coexistence*. ZigBee Alliance, 2011. https://docs.zigbee.org/zigbee-docs/dcn/07-5219.PDF.

[5] ANT. Ant - the power of less. http://www.thisisant.com; Accessed: 2012-07-25.

[6] Signove Antidote. http://oss.signove.com/. Accessed: 2012-08-12.

[7] Apple. Apple iphone - app store. http://www.apple.com/iphone/apps-for-iphone/. Accessed: 2012-02-14.

[8] Statistik Austria. *Demographisches Jahrbuch 2010*. Bundesanstalt Statistik Oesterreich, 2011.

[9] N. Baker. Zigbee and bluetooth strengths and weaknesses for industrial applications. *Computing Control Engineering Journal*, 16(2):20 –25, april-may 2005.

[10] BKA. Bundesgesetz ueber den schutz personenbezogener daten. http://www.ris.bka.gv.at. Accessed: 2012-02-14.

[11] M.T. Camp. Development of the bluetooth version 1.0 specification. In *IEEE Emerging Technologies Symposium: Broadband, Wireless Internet Access*, page 7 pp., 2000.

[12] Austrian Data Protection Commission. Website of the austrian data protection commission. http://www.dsk.gv.at/site/6295/default.aspx. Accessed: 2012-02-14.

[13] CompanionAble. Companionable - integrated cognitive assistive & domotic companion robotic systems for ability & security. http://www.companionable.net; Accessed: 2012-08-10.

[14] Continua. *Continua Health Alliance: The Next Generation of Personal Telehealth is Here*. Continua Health Alliance, 2012.

[15] J.D. Day and H. Zimmermann. The osi reference model. *Proceedings of the IEEE*, 71(12):1334 – 1340, dec. 1983.

[16] DKE. *Deutsche Normungs-Roadmap AAL*. Deutsche Kommission Elektrotechnik Elektronik Informationstechnik im DIN und VDE, 2011.

[17] M. Eichelberg. Die bedeutung von interoperabilitaet fuer aal. BMBF/VDE Innovationspartnerschaft AAL - AG Schnittstellenintegration und Interoperabilitaet, April 2010.

[18] Elvis Pf"utzenreuter et al. *Antidote: Program Guide - Documentation for developers of applications based on Antidote IEEE 11073 library*. Signove Antidote, 2012.

[19] J. Oeppen et al. Broken limits to life expectancy. *Science*, 296 no. 5570:1029–1031, May 2002.

[20] Jeff Drake et al. *Energy Efficiency Comparisons of Wireless Communication Technology Options for Smart Grid Enabled Devices*. General Electric Company, 2010. http://www.brymercreative.com/geal_2010/images/120910_zigbee.pdf.

[21] A. Fioravanti, G. Fico, M.T. Arredondo, D. Salvi, and J.L. Villalar. Integration of heterogeneous biomedical sensors into an iso/ieee 11073 compliant application. In *2010 Annual International Conference of the IEEE on Engineering in Medicine and Biology Society*, pages 1049 –1052, 31 2010-sept. 4 2010.

[22] M. Galarraga, L. Serrano, I. Martinez, P. de Toledo, and M. Reynolds. Telemonitoring systems interoperability challenge: An updated review of the applicability of iso/ieee 11073 standards for interoperability in telemonitoring. In *29th Annual International Conference of the IEEE on Engineering in Medicine and Biology Society*, pages 6161 –6165, aug. 2007.

[23] R.G. Garroppo, L. Gazzarrini, S. Giordano, and L. Tavanti. Experimental assessment of the coexistence of wi-fi, zigbee, and bluetooth devices. In *2011 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1 –9, june 2011.

[24] ARS Software GmbH. Bluetooth - bluetooth health device profile (hdp). http://www.ars2000.com/Bluetooth_HDP.pdf. Accessed: 2012-07-17.

[25] ARS Software GmbH. Bluetooth - ein standard fuer drahtlose kommunikation im nahbereich. http://www.ars2000.com/Bluetooth-drahtlose-Komm.pdf. Accessed: 2012-07-17.

[26] H.-M. Gross, C. Schroeter, S. Mueller, M. Volkhardt, E. Einhorn, A. Bley, C. Martin, T. Langner, and M. Merten. Progress in developing a socially assistive mobile home robot companion for the elderly with mild cognitive impairment. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2430–2437, sept. 2011.

[27] Thomas R. Gruber. Toward principles for the design of ontologies used for knowledge sharing. *International Journal Human-Computer Studies*, 43:907–928, aug. 2003.

[28] UniversAAL Ontology Guidelines. http://forge.universaal.org/wiki/ontologies:guidelines. Accessed: 2012-08-31.

[29] Open Health. The openhealth floss implementation of the iso/ieee 11073-20601 standard. Workshop OSEHC 2010; http://androidfloss.libresoft.es/sites/androidfloss.libresoft.es/files/WorkshopOSEHC_2010_6.pdf; Accessed: 2012-08-11, 2010.

[30] Dr. Sumi Helal. Mobile and pervasive computing laboratory, university of florida. http://www.harris.cise.ufl.edu/opensource.htm. Accessed: 2012-08-11.

[31] Sumi Helal, Raja Bose, Chao Chen, Andy Smith, Scott de Deugd, and Diane Cook. Stepstone: An intelligent integration architecture for personal tele-health. *Journal of Computing Science and Engineering*, 5(3):269–281, sept. 2011.

[32] Claire Huijnen, Atta Badii, Herjan van den Heuvel, Praminda Caleb-Solly, and Daniel Thiemert. Maybe it becomes a buddy, but do not call it a robot - seamless cooperation between companion robotics and smart homes. In David Keyson, Mary Maher, Norbert Streitz, Adrian Cheok, Juan Augusto, Reiner Wichert, Gwenn Englebienne, Hamid Aghajan, and Ben Kroese, editors, *Ambient Intelligence*, volume 7040 of *Lecture Notes in Computer Science*, pages 324–329. Springer Berlin / Heidelberg, 2011. 10.1007/978-3-642-25167-2_44.

[33] Lamprey Networks Inc. Oxplib whitepaper. http://www.lampreynetworks.com/assets/ documents/OXPlib_whitepaper.pdf. Accessed: 2012-08-11.

[34] Dynastream Innovations. *ANT Message Protocol and Usage*. ANT Alliance, 2011. thisisant.com/images/Resources/PDF/1204662412_ant_message_protocol_and_usage.pdf; Accessed: 2012-10-15.

[35] Texas Instruments. *2.4-GHz Bluetooth low energy and Proprietary System-on-Chip CC2541*. Texas Instruments, 2012. http://www.ti.com/product/cc2541.

[36] ISO/IEEE. Iso/ieee health informatics - point-of-care medical device communication - part 10101: Nomenclature. *ISO/IEEE 11073-10101:2004(E)*, pages 1–492, 2004.

[37] ISO/IEEE. Health informatics-personal health device communication part 10407: Device specialization - blood pressure monitor. *IEEE STD 11073-10407-2008*, pages c1 –41, 2008.

[38] ISO/IEEE. Iso/iec/ieee health informatics–personal health device communication–part 20601: Application profile–optimized exchange protocol. *ISO/IEEE 11073-20601:2010(E)*, pages 1 –208, 1 2010.

[39] A. Johansson, Wei Shen, and Youzhi Xu. An ant based wireless body sensor biofeedback network for medical e-health care. In *2011 7th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM)*, pages 1 –5, sept. 2011.

[40] K. Kalogirou and G. Telkamp. An ontological framework for the elderly to control their home environment. In *IST-Africa, 2010*, pages 1–7, may 2010.

[41] Harald Kuenemund. Changing welfare states and the 'sandwich generation' : Increasing burden for the next generation? *International Journal of Aging and Later Life*, pages 11–29, 2006.

[42] Jin-Shyan Lee, Yu-Wei Su, and Chung-Chou Shen. A comparative study of wireless protocols: Bluetooth, uwb, zigbee, and wi-fi. In *33rd Annual Conference of the IEEE Industrial Electronics Society, 2007*, pages 46 –51, nov. 2007.

[43] Madrid LibreSoft Group, Rey Juan Carlos University. Openhealth website. http:// openhealth.libresoft.es. Accessed: 2012-08-11.

[44] A. Living. The aaloa manifesto. *AALOA*, 2006.

[45] Fraunhofer-Allianz Ambient Assisted Living. Persona. http://www.aal.fraunhofer.de /projects/persona.html. Accessed: 2012-06-29.

[46] M. Martinez-Espronceda, L. Serrano, I. Martinez, J. Escayola, S. Led, J. Trigo, and J. Garcia. Implementing iso/ieee 11073: Proposal of two different strategic approaches. In *30th Annual International Conference of the IEEE on Engineering in Medicine and Biology Society*, pages 1805 –1808, aug. 2008.

[47] P. McDermott-Wells. What is bluetooth? *IEEE Potentials*, 23(5):33 – 35, 2004-jan. 2005.

[48] I.H. Mulyadi, E. Supriyanto, N.M. Safri, and M.H. Satria. Wireless medical interface using zigbee and bluetooth technology. In *Third Asia International Conference on Modelling Simulation, 2009*, pages 276 –281, may 2009.

[49] United Nations. United nations statistics division. http://unstats.un.org/unsd/default.htm. Accessed: 2012-02-14.

[50] Mobi Health News. Which technology should continua pick? http://mobihealthnews.com/1053/which-technology-should-continua-pick/. Accessed: 2012-08-11.

[51] T. Norgall. Telemedizin und ambient assisted living (aal) - eine bestandsaufnahme 2009. DGTelemed Fachkongress: Telemedizin 2009 - Geschaefts- und Versorgungsmodelle im klinischen Alltag, Nov. 2009.

[52] Jad Noueihed, Robert Diemer, Samarjit Chakraborty, and Stefanie Biala. Comparing bluetooth hdp and spp for mobile health devices. In *2010 International Conference on Body Sensor Networks (BSN)*, pages 222 –227, june 2010.

[53] OASIS. Oasis project presentation. http://www.oasis-project.eu/docs/OFFICIAL_ DELIVERABLES/ SP5/OASIS_D5.1.3_Project_Presentation.pdf; Accessed: 2012-08-08.

[54] Council of the European Communities Data Protection Working Party. Council directive 93/42/eec concerning medical devices. http://eur-lex.europa.eu/ LexUriServ/LexUriServ.do?uri=CONSLEG:1993L0042:20071011:EN:PDF. Accessed: 2012-06-24.

[55] Council of the European Communities Data Protection Working Party. Opinion 4/2007 on the concept of personal data. http://ec.europa.eu/justice/. Accessed: 2012-02-14.

[56] Chan-Yong Park, Joon-Ho Lim, and Soojun Park. Iso/ieee 11073 phd standardization of legacy healthcare devices for home healthcare services. In *IEEE International Conference on Consumer Electronics (ICCE)*, pages 547 –548, jan. 2011.

[57] Berlin Robert Koch-Institut. *Gesundheit und Krankheit im Alter*. Statistisches Bundesamt; Deutsches Zentrum fuer Altersfragen; Robert Koch-Institut, 2009.

[58] L. Schmitt, T. Falck, F. Wartena, and D. Simons. Novel iso/ieee 11073 standards for personal telehealth systems interoperability. In *Joint Workshop on High Confidence Medical Devices, Software, Systems and Medical Device Plug-and-Play Interoperability (HCMDSS-MDPnP)*, pages 146 –148, june 2007.

[59] M. Siekkinen, M. Hiienkari, J.K. Nurminen, and J. Nieminen. How low energy is bluetooth low energy? comparative measurements with zigbee/802.15.4. In *Wireless Communications and Networking Conference Workshops (WCNCW), 2012 IEEE*, pages 232 –237, april 2012.

[60] Signove. Signove website. http://www.signove.com/. Accessed: 2012-08-11.

[61] Andrew Sixsmith, Sonja Meuller, Felicitas Lull, Michael Klein, Ilse Bierhoff, Sarah Delaney, and Robert Savage. Soprano - an ambient assisted living system for supporting older people at home. In Mounir Mokhtari, Ismail Khalil, Jeremy Bauchet, Daqing Zhang, and Chris Nugent, editors, *Ambient Assistive Health and Wellness Management in the Heart of the City*, volume 5597 of *Lecture Notes in Computer Science*, pages 233–236. Springer Berlin / Heidelberg, 2009. 10.1007/978-3-642-02868-7_30.

[62] SOPRANO. Soprano - service-oriented programmable smart environments for older people. http://www.soprano-ip.org; Accessed: 2012-08-08.

[63] Stollmann. Stollmann - ieee 11073. http://www.stollmann.de/. Accessed: 2012-08-11.

[64] H.P. Tews. *Von der Pyramide zum Pilz. Demographische Veraenderungen in der Gesellschaft*. Funkkolleg Altern 1. VS Verlag fuer Sozialwissenschaften, 1999.

[65] WIPRO Applying Thought. Towards a continua compliant tomorrow. www.wipro.com/ industries/medical-devices/continua-toolkit.aspx. Accessed: 2012-08-11.

[66] Open Health Tools. Open health tools - stepstone website. https://www.projects. openhealthtools.org/sf/projects/stepstone/. Accessed: 2012-08-11.

[67] B.Q. Tran. Home care technologies for promoting successful aging in elderly populations. In *Proceedings of the Second Joint of the 24th Annual Conference and the Annual Fall Meeting of the Biomedical Engineering Society on Engineering in Medicine and Biology Conference*, volume 3, pages 1898 – 1899 vol.3, oct. 2002.

[68] UniversAAL. http://www.universaal.org/. Accessed: 2012-02-14.

[69] UniversAAL. Universaal reference architecture d 1.3-c. http://www.universaal.org/images/stories/deliverables/D1.3-C.pdf. Accessed: 2012-02-14.

[70] Mike Uschold, Michael Gruninger, Mike Uschold, and Michael Gruninger. Ontologies: Principles, methods and applications. *Knowledge Engineering Review*, 11:93–136, 1996.

[71] G. Virone and A. Sixsmith. Monitoring activity patterns and trends of older adults. In *30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2008*, pages 2071–2074, aug. 2008.

[72] WhatIs.com. Whatis.com - find a tech definition. http://whatis.techtarget.com/; Accessed: 2012-08-08.

[73] Wikipedia. Wikipedia. wikipedia.org; Accessed: 2012.

[74] Wiktionary. Wiktionary. en.wiktionary.org; Accessed: 2012.

[75] J. Yao, S. Simmons, and S. Warren. Ease of use considerations for wearable point-of-care devices in home environments. In *1st Transdisciplinary Conference on Distributed Diagnosis and Home Healthcare (D2H2)*, pages 8 –11, april 2006.