

Project acronym	CAMI
Project number	AAL 2014-1-087
Project full name	Artificially intelligent ecosystem for self-management and sustainable quality of life in AAL
Dissemination level	Restricted
Type of deliverable	Report
Contractual Date of Delivery	M10
Actual Date of Delivery	M10
Deliverable Number	D2.2
Deliverable Name	Report on the ecosystem architecture
Work package / Task	WP2 / Task 2.2
Work package responsible / Task responsible	MDH / MDH
Number of Pages	43
Contributors	All partners
Version	4
Keywords	AAL architecture, local and cloud based architecture
Abstract	The aim of this deliverable is designing of an integrated modularised architecture for CAMI by analysing the pros and cons of existing AAL architectures and also depicting the interactions of CAMI modules via certain scenario analysis.

Table of Contents

Executive summary	4
List of Figures	4
1. Introduction	5
2. Scenarios	8
2.1. Physical Exercise Monitoring Episode	8
2.2 Medication Compliance and Reminding Scenario	9
3. Related work on AAL Architectural Solutions	9
3.1 Cloud based AAL architecture	10
3.2 Agent based AAL architectures	12
3.3 IoT Architecture for AAL	13
4. CAMI Architecture	15
5. CAMI Modules and Protocols	17
5.1 Sensor Unit	17
5.2 Data Collector Unit	18
5.3 CAMI Gateway	20
5.3.1 Event Stream Manager	22
5.3.2 Decision Support System	23
5.3.3 Reverse Connection Manager	25
5.3.4 CAMI Box MySQL DB	26
5.4 User Interface	26
5.4.1 Vocal Interface	27
5.4.2 Graphic User Interface	29
5.5. Telepresence	31
5.6. Communication with caregivers and health professionals	32
5.7 Cloud Services	33
5.7.1 Cloud MySQL DB and RDF Data Storage	35
6. Scenarios revisited	36
7. Conclusions	37
References	38

Executive summary

Aim of the deliverable

The objective of this deliverable is to analyze the existing architectures that support the development of an integrated architecture in order to develop an architecture framework for CAMI, outline the CAMI architecture and its modules and their interactions. The advantages and disadvantages of existing frameworks were carefully considered while developing the architecture framework for CAMI. The deliverable also details the innovative integrated architecture framework which we designed for CAMI with multi-functionality support and the independent modules. We have also detailed the interaction of CAMI modules using certain scenarios.

Brief description of the sections of the document

Section 1 deals with the overview of functionalities that are integrated to the CAMI architecture. In Section 2, we describe some scenario based analysis in order to have a visual interplay of these functionality modules. As part of Section 3, we outline the major architecture categories for AAL that can be potentially used as a base for developing CAMI architecture by identifying the pros and cons of each with respect to the requirements of CAMI architecture. In Section 4, we describe the CAMI architecture and further in Section 5, the architecture modules namely the Sensor Unit, CAMI gateway, user interfaces, telepresence, cloud services etc. are described. Section 6 deals with revisiting of existing scenarios described in Section 2 specific to CAMI with the actors involved and interactions between the various CAMI modules. Section 7 provides the concluding remarks followed by the references.

Major achievements

The major achieves of this deliverable are (i) a detailed literature survey of existing AAL frameworks (ii) development of an innovative architecture for CAMI (iii) Analysis of CAMI system modules and their interactions.

Summary of the conclusions obtained

The major outcome of this work is the design of an integrated modularised architecture for CAMI by analysing the pros and cons of existing AAL architectures and also depicting the interactions of CAMI modules via certain scenario analysis.

List of Figures

Figure 1 . Supervised physical exercises	6
Figure 2 Development of a generic framework for health monitoring through IOT [11]	11
Figure 3 A Distributed Ambient Intelligence Based Multi-Agent System for Alzheimer Health Care [12]	12
Figure 4 The IoT in an AAL scenario integrating KIT and closed loop health services [13]	14
Figure 5 CAMI Architecture Diagram	16
Figure 6 The EXYS9200-SNG	19
Figure 7 The EXYS9200 system software architecture	20
Figure 8 DSS architecture	25
Figure 9 CAMI User Interface input and output options.	28
Figure 10 CAMI Voice Command Manager Block Architecture	29
Figure 11 OpenTele client application main menu (interface in Danish).	31
Figure 12 Screenshot of weight overview interface in Linkwatch platform	32

1. Introduction

The increasing ageing population demands for solutions that help in the independent, healthy and risk free life of the elderly and prevent their social isolation. The improvements in Information and Communication Technologies (ICT) and Ambient Assisted Living (AAL), resulted in the development of technologies that support ubiquitous computing, ubiquitous communication and intelligent user interfaces. The smart home technologies, assisted robotics, health sensing and telemedicine are some of the few examples to site. Though these individualized technologies have grown so much, there are very less developments that have done in developing an integrated AAL solution which interconnects smart home facilities, health sensors, telemedicine, robotics and much more. In CAMI, we have developed an ecosystem architecture which have integrated all the major functionalities required in an Ambient Assisted Living System.

CAMI is offering a fully integrated AAL solution by providing services for health management, home management and wellbeing (including socialization, and reduced mobility support). CAMI builds an artificial intelligence ecosystem, which allows seamless integration of any number of ambient and wearable sensors with a mobile robotic platform endowed with multimodal interaction (touch, voice, person detection), including a telepresence robot with manipulator capabilities. The services offered by CAMI ecosystem address both healthy individuals as well as those with age-related impairments. CAMI solution will reconcile the increased demand for care in the current aging society with limited resources by supporting an efficient and sustainable care system. This will allow older adults to self-manage their daily life and prolong their involvement in the society while allowing their informal caregivers to continue working whilst caring for their loved ones.

CAMI will offer/we have to build a **Core System Component** with a set of APIs that can be used by several modules, offering different services to the user. The basic idea is that each module can and should be developed independently but must seamlessly integrate with the CORE System Component in order to have in the end a fully integrated and functional system. The system should be functional with all to none – except 5 – of the modules.

The main modules of the CAMI system that are able to cover all the functionalities are discussed below:

a. Health monitoring:

This functionality calls for (1) regular monitoring of health parameters and for (2) continuous monitoring of health parameters during exercise. Data recorded in (1) will be stored and integrated/correlated in the user profile. It will be available to caregivers and will be used for defining the level and intensity of physical exercises. Data recorded in (2) will be used for monitoring the health and activity level (e.g. heart rate) during physical exercises such that the user

remains in her/his comfort and low risk zone

b. Fall alarm:

It will be achieved through integration of data coming from various sources, e.g.: wearable sensors; mobile phone potentially carried by the user; information from a mobile platform. Detected falls will be sent to formal or informal caregivers on a mobile platform.

c. Report and communication to health professionals

Health parameters acquired with the first component can be provided to formal and informal caregivers. These vital parameters associated with a person will be visualized through a web interface. Emails or SMS will be sent in case of different types of alarms. These parameters or alerts will be provided in specific ways for different type of caregivers (formal or informal).

d. Supervised physical exercises

In case of low physical activity this component will advise the user to increase her/his level of physical exercise. The exercises will be performed interactively, as a game, specifically designed for elderly persons using the Kinect sensor. The intensity level of the recommended physical exercises will be generated based on vital parameters of the user measured by the health monitoring component. The physical exercises will be presented to the user using an avatar created based on the user's profile. Some examples of the exercises are given below:



	
<ul style="list-style-type: none"> • body straight, feet apart • fingers on the shoulders and rotate the arms forward four times and four times backwards 	<ul style="list-style-type: none"> • body straight, feet apart • upper left arm and the right foot, arm extensions pulling back, exchanged identical arms

Figure 1 . Supervised physical exercises

e. Personalized, intelligent and dynamic program management

Allows the introduction of personal data about the user: medication plan, daily, weekly and month program planning, exercise planner, record of data obtained from sensors, including medical data, interactions with formal and informal caregivers. Depending on various conditions and actual recorded data, the module compares what has been achieved to what has been

planned and is able to dynamically adjust the user program. Adjustments will be proposed by the system and validated by caregivers and/or doctor.

- f. Telepresence for communication (video, voice, graphics) with caregivers, family, friends and other users** - Mobile robotic platform. There are two basic directions of development:

1. telepresence like systems (no manipulation like capabilities)
2. mobile robot with manipulator

An important purpose is to ensure the interface between the human and the user. Because the interface is mobile, the interaction will happen where the user is, i.e. the user will not have to go himself to the interface. Thus, we will not need to install complicated hardware in all the rooms.

A second important functionality of the platform is that it is a mobile system equipped with several sensors and thus represents an important source of data in the system. For e.g. the platform can be equipped with a temperature sensor and since it will move throughout the environment there is less need to install sensors in the whole environment.

The mobile robot with manipulator has all the above functionalities plus the ability to manipulate objects. The most realistic manipulation scenario is through remote control by a human operator (e.g. family, caregiver).

In order to ensure a smooth and coherent we consider platforms based on ROS (Robot Operating System).

- g. An intelligent, informed, friendly collaborator, taking orders, giving advice or reminders and ready to help, and get help, when needed.**

This is the Intelligent Personal Assistant of the user. It is able to use the vocal interface (9). It is able to understand simple commands, but also to translate text-to-speech.

It is conceived as an interface to the program manager of the user and links/offers access to the other components of the system when needed.

The Assistant is able to take user input but also to identify context and react according to input, user preferences and context.

Among functionalities, we can mention: activity summary, event and program reminders, TV program reminders, appointments, time reminders (including medicine), weather update, etc. (we can add inspiration for functionalities from Siri and/or Google Now).

- h. Vocal interface to allow multi-modal interfaces**

The elderly remains excluded from technology, since they regard traditional computer interfaces as overly technical and difficult to use. However, the older users consider other forms of interaction easier to use - for example vocal commands. This component will help the users to access the CAMI functionalities in an easier way through vocal commands. Existing software for speech recognition and synthesis will be used.

- i. **Home and environment management:** This module will record and allow control of environment parameters in the user premises. Some of the parameters will be used in connection with the home appliances to increase the comfort of the user. Others will be used to control the environment, e.g. remote switching of lights.

The CAMI system architecture developed for catering into the various functionalities as described above and the individual modules are described in detail in Section 3.

2. Scenarios

In the introduction we presented the set of overall functionalities that the CAMI system will integrate and cater for.

To visualize the interplay of these functionality modules, we present a set of three short episodes envisaged to typically take place within CAMI.

We describe the actors (end users, primary and secondary caregivers) involved in each episode and hint towards the type of services that need to interact in order to realize the scenarios. In Section 6 we will revisit these episodes and perform a breakdown into a sequence of interactions between the microservices that have been introduced throughout this document. We will thus show that the chosen CAMI architecture enables the cooperation between very diverse services.

2.1. Physical Exercise Monitoring Episode

Jim is starting a physical exercise session based on a notification from CAMI, which reads the event from Jim's calendar. CAMI follows the form and the number of the exercises he performs through a Kinect camera. It is continuously monitoring Jim's pulse which varies during the exercises. Depending on Jim's pulse rate, CAMI will increase or decrease the exercise level, it will suggest a new exercise or it will suggest the ending of the current physical exercise session. The results are recorded and shared with the caregivers who are interested in the compliancy with the prescribed exercise plan.

This episode involves a single end-user (Jim) and the caregiver's (primary and/or secondary) who monitor his physical exercise history.

The implementation of the scenario involves use of personalized program management (for exercise reminders), supervised physical exercises, health data measurement and health data communication functionality modules identified in the introduction.

The complexity of the scenario is further raised by the fact that two types of data collection have to be handled at the same time: event-based data (e.g. Jim's pulse) and streamed data (e.g. the image stream recording Jim's correct execution of the planned physical exercise).

2.3. Fall Detection Scenario

Jim wakes at night and needs to go to the bathroom. Jim feels dizzy after getting out of bed. The light turns on automatically, but Jim loses his balance and falls down. The fall detector sensor Jim is wearing senses a fall. It sends an event to the CAMI Gateway, informing of a potential fall. The sensor is only partly aware that a fall has taken place, and reports having low confidence in the detected fall. Jim does not return to bed, and no further movement is detected in the home. The ambient fall detection sensors also report that they suspect a fall might have occurred. CAMI takes a decision to alert Jim's caregivers (family members), who receive an alert on their smartphone CAMI app.

[Optional: Jim's son logs onto the telepresence robot, and sees that Jim has actually fallen. Thus, Jim's son drives to Jim's home and helps him get up and make sure that he does not need acute care.]

This is an advanced interaction which makes intense use of CAMI's intelligent reasoning capabilities. Specifically, it involves progressive (i.e. in time) correlation of information from two sensor sources (fall detection wearable, ambient movement detection sensors) to strengthen the belief in an *actual* fall detection (reduce false alarms as much as possible). It further involves sending of alert messages to secondary caregivers.

2.2 Medication Compliance and Reminding Scenario

Jim forgets which medicine he must take and/or the time at which he must medicate himself. Using voice-based interaction, he asks CAMI to inform him of the prescribed medication plan. If Jim forgets to take his medication, CAMI reminds him he must do so. If he doesn't acknowledge the reminder within the hour, CAMI notifies a family member. If this situation repeats itself too often, CAMI will inform a caregiver directly.

This is a typical AAL interaction, where a user seeks information to conform to his prescribed medication plan. The added value from a simple schedule lookup is the ability to ask the system for the type and number of pills via voice commands, resulting in an easier interaction.

The intelligent and friendly collaborator module issues reminders in a context-aware manner. The CAMI system also seeks confirmation (via display or voice based interaction) of compliance to the medication reminder. Family members and caregivers are kept in the loop if the elderly user ignores critical reminders too often, using the communication with health professional's module.

The above scenarios cover types of interaction which have a complex interplay between diverse services (e.g. health data measuring, physical exercise planning and tracking, health data sharing, communication with caregivers, prompt reminders, intelligent personal assistance and decision making). In Sections 4 and 5 we detail the CAMI architecture and in Section 6 we revisit the introduced interaction episodes, performing a breakdown of each episode along the services of the CAMI system, showing how they interact to enable the scenario functionality.

One important requirement for the CAMI project is the *reuse* of existing consortium member knowhow and solutions.

In the next Section we perform an overview of different architecture types usually considered for AAL applications. We analyze their general pros and cons, but also their appropriateness to the exemplified scenarios and the perspective of integrating the listed CAMI functionality modules. The analysis will show why the CAMI system requires a custom and hybrid design, combining local and cloud-based deployment.

3. Related work on AAL Architectural Solutions

Ambient Assisted Living (AAL) applications are designed to ensure that elderly people can live independently and safe in their own home, enabling proper care and social interaction support when needed.

Observing the different AAL solutions in recent years, it becomes apparent that such applications must accommodate for functionalities from several interest domains: Ambient Intelligence (Aml), recommender systems, health monitoring, telemedicine, social interaction and privacy.

Thus, the backbone of an AAL system must essentially be grounded in an architecture that enables the integration of components from all these domains.

Considering the nature of AAL projects and consortium organizations it becomes very important to focus on reusability of the individual knowhow and technologies that are already being developed by project members.

Given this concern and the diversity of modules considered for the CAMI system, we place a lot of focus on the following list of non-functional attributes:

- Modularity and extensibility: the CAMI system must be composed in modular fashion, whereby a core infrastructure provides support for **pluggable functionality**, in the form of micro-services, which can be added in an incremental fashion to enhance overall capabilities of CAMI.
- Reliability: since the CAMI system involves interaction between several independent services, it becomes essential to monitor the runtime execution of each service, as well as that of the core infrastructure. Logging and error reporting mechanisms are required, as well as means to ensure continued functionality of the CAMI system if one of its services fails to operate properly.
- Security: CAMI involves the collection of personal health data, the ability to share such data, as well as the use of telepresence sessions which involve the streaming of data over the internet. This means that aspects of data transmission security are important and must be addressed by building CAMI around technologies and frameworks which offer strong security guarantees intrinsically.
- Ease of Implementation: targets the aspect of know-how and reusability of existing services. The core infrastructure of the CAMI architecture must offer the necessary support such that instances of a CAMI system can be quickly setup, configured and monitored. It also has to ensure that interaction between plugged in micro-services (which are developed independently) happens in an easy manner, without having to

redesign interfaces between each and every microservice.

Examining the AAL literature, one can identify several architecture types which address the construction of integrative AAL applications (i.e. those that focus on creating a holistic user experience, not just the development of a specific functionality, such as health data management or social interaction).

In what follows, we investigate three types of commonly used architecture types (cloud-based, multi-agent system based and Internet-of-Things centric), listing an example for each type and analysing their pros and cons with respect to envisioned CAMI functionality.

We end by motivating why the CAMI architecture needs a custom and hybrid design which borrows elements from each of the three presented architecture types.

3.1 Cloud based AAL architecture

Cloud based AAL architectures typically involve a collector unit which can retrieve data from the user and her environment and then directly send it to a cloud platform where all the actual intelligent processing takes place.

An example instance, taken from the ESS-H (Embedded Sensor Systems for Health) [11] solution is showcased in the diagram of Figure 2.

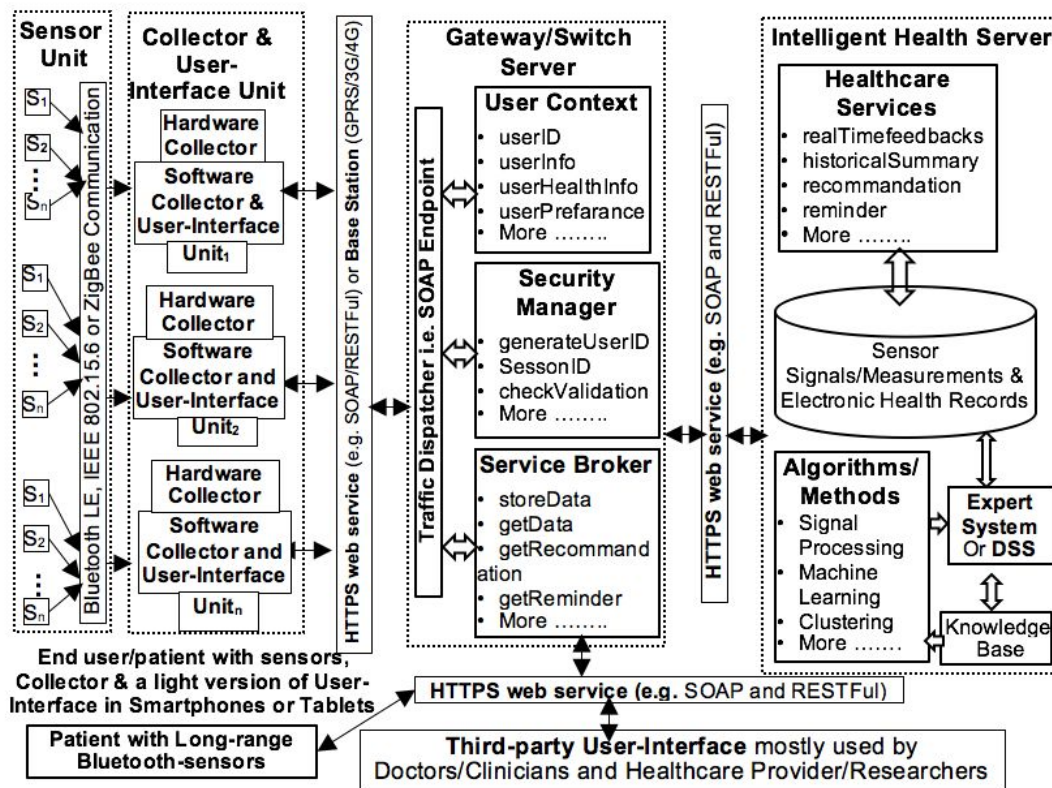


Figure 2 Development of a generic framework for health monitoring through IOT [11]

The major components in the architecture are the Sensor and Collector Units which perform data retrieval, the User-Interface Unit, as well as the Gateway and Switch Server and Intelligent Health Server which address communication, storage and intelligent processing.

The pros of the exemplified cloud based solution with respect to CAMI are:

- Architecture elements and functionalities well defined.
- The architecture is reliable and extensible.
- The system has a higher ease of implementation and maintenance because deployed services need not be monitored remotely and any update (improvement or fix) to system functionality becomes readily accessible to all users.
- Choice of communication protocols is flexible.
- Security and privacy aspects are addressed in the architecture
- Functionalities like health monitoring, communication with external third party users, recommendation and reminders are addressed.

However, a pure cloud-based solution has difficulty addressing two main CAMI functionality modules: physical exercise execution and the integration of robotic telepresence units.

As we detail further in Section 5.1, the physical exercise service involves analysis of image and depth data collected from video devices such as a Microsoft Kinect camera. Real-time streaming and analysis of such data in the cloud poses significant communication and performance challenges. Similarly, communication with a robotic telepresence unit (as detailed in Section 5.3) needs to happen in real time.

Consequently, the CAMI architecture has to provision for a local deployment of these services and a local processing of data collected from them.

3.2 Agent based AAL architectures

Multi-Agent Systems are a paradigm often seen as well suited for AAL and Ambient Intelligence (Aml) applications. Typical agent attributes such as autonomy, reactivity, proactivity and message-based communication are desirable with respect to AAL and Aml scenarios involving interaction between many independent services.

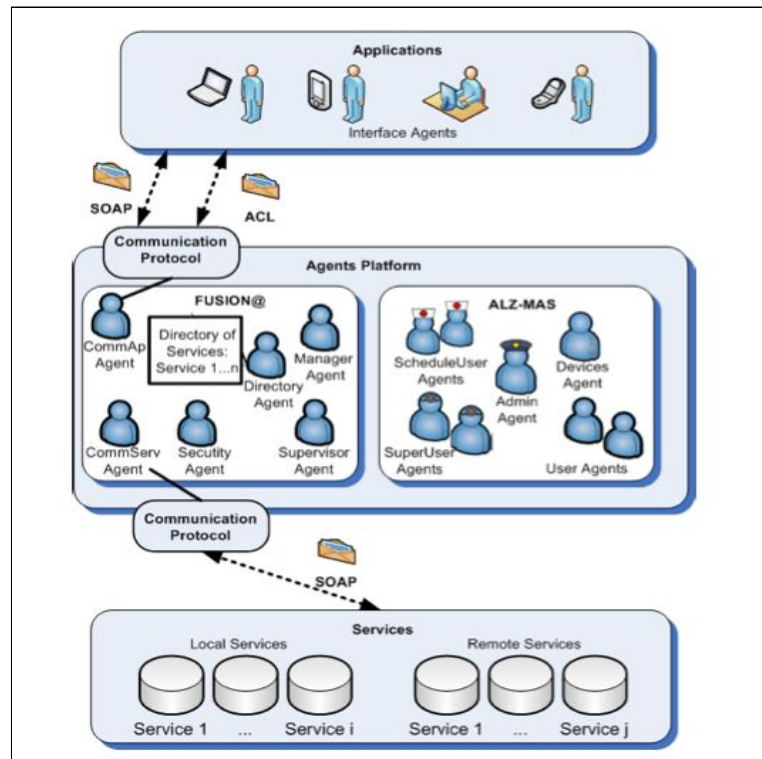


Figure 3 A Distributed Ambient Intelligence Based Multi-Agent System for Alzheimer Health Care [12]

We discuss a distributed multi agent system developed for Alzheimer health care [12] to showcase the pros and cons of such architectures. The architecture diagram is shown in Fig 3.

The major features of the architecture are as follows:

- Service Oriented Architecture based on distributed multi agent systems.
- The architecture is based on FUSION@(Flexible User and Services Oriented multi-agent Architecture) where MAS and SOA services are integrated.
- Agents are based on deliberative Belief-Desire-Intention (BDI) paradigm.
- Case Based Reasoning and Case Based Planning techniques.

For CAMI the pros of such an approach consist in:

- Independent functionality modules can be modeled as flexible, autonomous agents ensuring service continuity.
- The combination of message-based communication and direct calls via SOAP allow for flexibility in interconnection of agent wrapped functionalities (e.g. schedule management, device management), which is essential for integration and reuse of existing services
- Agent functionality follows the service oriented approach rendering the system more extensible.

However, from the CAMI perspective, several downsides of architectures such as the one in

Figure 3 can be identified:

- The local deployment of agents adds challenges in sharing information (such as in telepresence calls or health measurement sharing) with primary and secondary caregivers, since it would involve peer-to-peer connectivity.
- More security needs to be incorporated for transferring and storing the sensitive data.
- Pure local deployment makes system maintainability difficult since it means remote monitoring and updates to the system must be serviced individually.
- Difficulty in developing MAS is higher because there are no specialised programming tools to develop agents

Consequently, the CAMI architecture would have to provide a means for cloud-based deployment of a multi-agent architecture, which already renders the system closer to the architecture type presented previously.

3.3 IoT Architecture for AAL

The IoT is highly relevant in the scenario of healthcare and the architecture which we describe in this section based on IOT features the integration of Keep In Touch (KIT) which uses smart objects and technologies like Near Field Communication and Radio Frequency Identification for telemonitoring and Closed Loop Health Services [13].

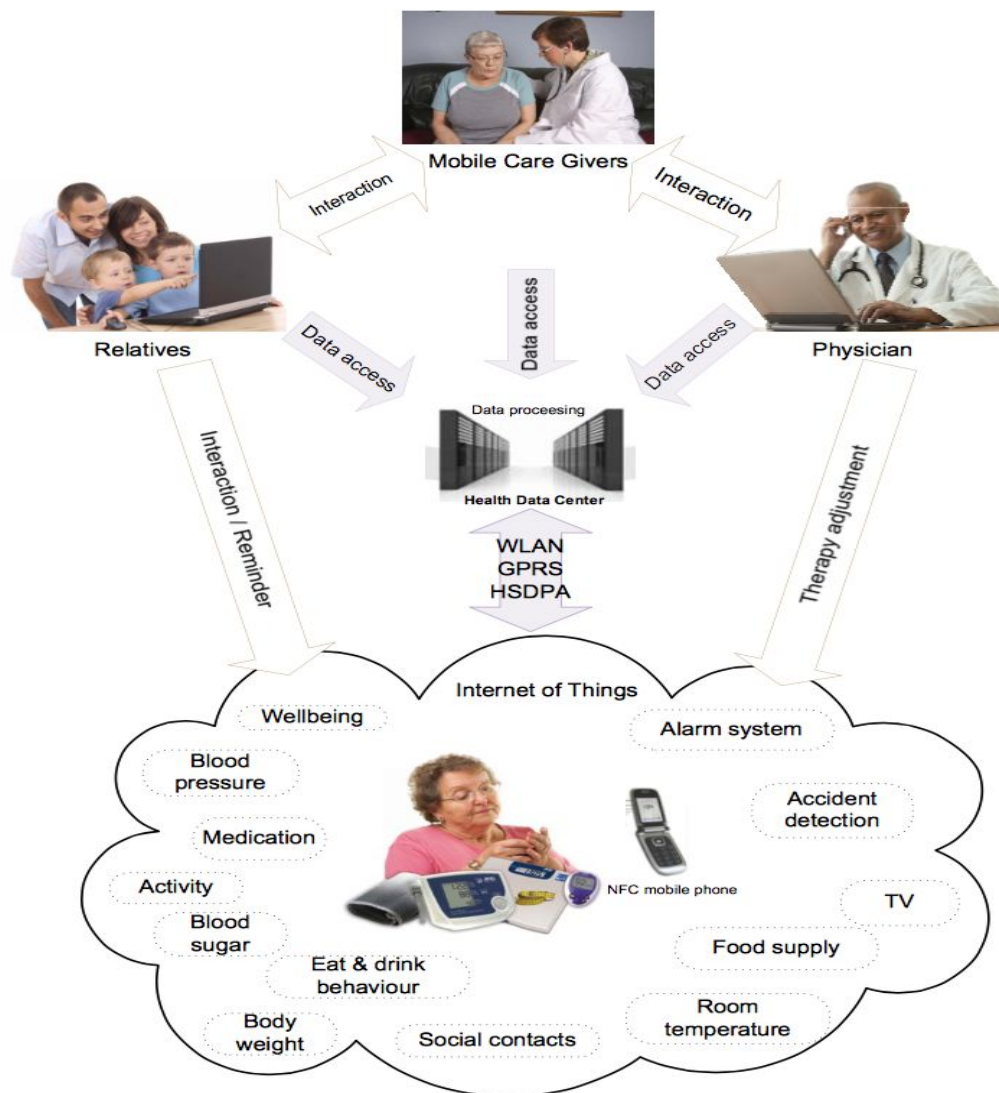


Figure 4 The IoT in an AAL scenario integrating KIT and closed loop health services [13]

As shown in Figure 4, the elderly people live in a smart home equipped with various smart objects. The mobile phone act as an interface for communication and sends data to service center using protocols such as WLAN, GPRS, HSDPA. This data can be accessed by physicians, caregivers and relatives.

The KIT system and architectures similar to it share the advantages of cloud-based architectures discussed previously, since the information flow and processing methods are similar.

However, from the CAMI perspective, we identify the following downsides:

- Mobile phone is the only interface for moving data to health center, which is like a central point of storage.

- The system fails to provision for inclusion of supervised physical exercises monitoring, robotic telepresence services or voice command interaction.

In this section, we showed that employing typical AAL architectures to support all the functionalities in CAMI (as discussed in Section 1) is not feasible.

Rather it is the case that we need to extend any of the existing architectures to suit CAMI functionalities. The need for both local and cloud-based processing, as identified through the scenarios in Section 2, lead us to a custom design, incorporating features from all three architecture types discussed here.

We begin the description of the CAMI architecture in the next Section.

4. CAMI Architecture

In this section, we discuss the proposed architecture for the core CAMI system, down to the level of individual functionality modules and the technologies that enable them.

Following the sections on envisioned scenarios and related work in AAL architectures, we concluded that the set of functionalities considered in CAMI warrant a custom architecture.

The motivation for this choice stems from the following assumptions that were made before formulating the design of the CAMI system:

- **Need of an architecture that leverages all the strengths of the partners in the project consortium** and their previous experiences.
- **Requirement of project feasibility from a commercial point of view.**
- **Appropriate provisions for handling health related data**, especially regarding means of sharing such data with primary (medical professionals) and secondary caregivers. In the EU and in the USA, there is a recent wave of laws regulating very strictly who can process and store such data and in what conditions (privacy and security related concerns).
- **Functional and business model related requirements (e.g. multiple system instances based on a pluggable microservice-based architecture) motivate a cloud-based infrastructure** for the CAMI system as a whole. This serves to store and perform complex processing of data, sharing it with the caregivers and make software updates much more convenient.

The CAMI system is based on a clean and robust skeleton, onto which several plugin modules (microservices) can be coupled.

The skeleton implements mechanisms for establishing a robust and monitorable data flow within the CAMI system. The skeleton consists of the standard, well known technologies that collect data from sensors and the human interface, route it using an event stream manager

and finally store it both locally, as well as in the cloud.

The rest of the components and services (e.g. activity recommendation service, intelligent reminders, fall alarms, voice commands, exercise analysis) act as plugin components (i.e. they are not strictly required for the CAMI system to work) that can be linked to the skeleton framework. This approach enables a clear separation of concerns.

The architecture diagram is depicted in Fig 5.

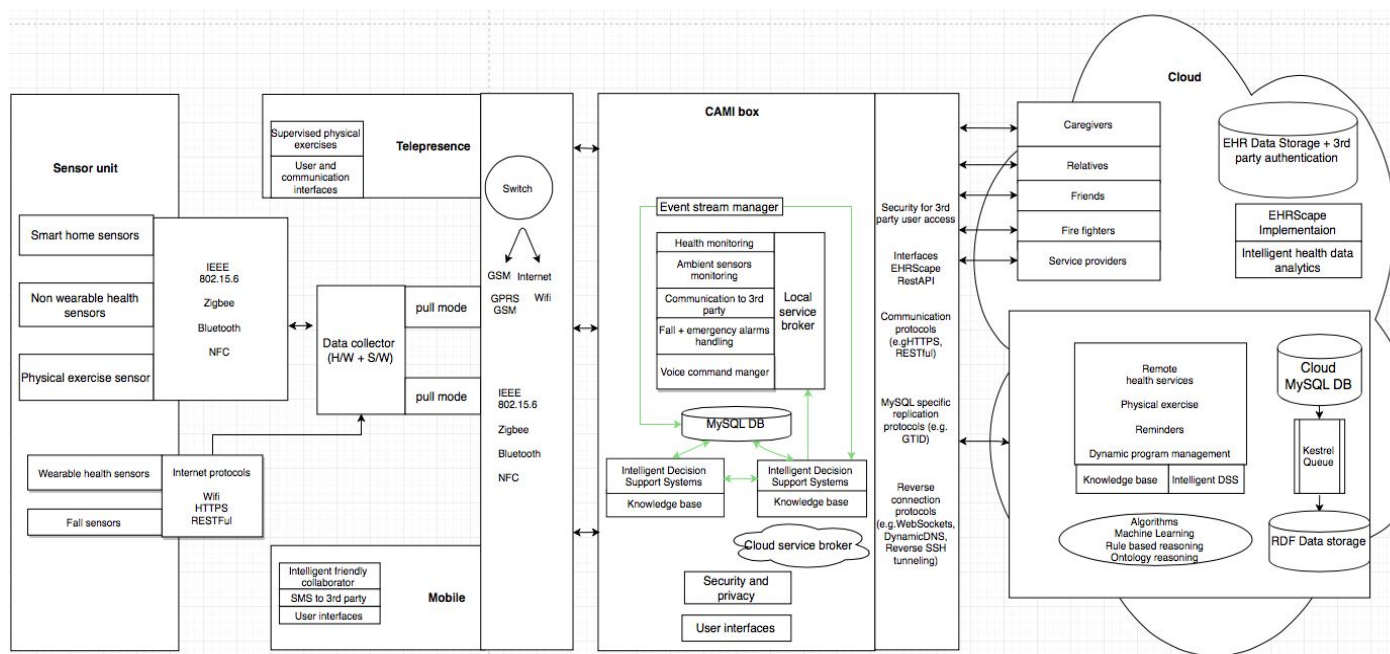


Figure 5 CAMI Architecture Diagram

The CAMI architecture considers six major blocks: Sensor Unit, Data Collector Unit, CAMI Gateway, Multimodal User Interface, Telepresence Unit and CAMI Cloud.

In the proposed CAMI architecture these blocks use **message-based communication** over a set of **publish/subscribe channels** to interact with each other. The main attributions of the CAMI skeleton thus become the following:

- Defining the input and output message envelopes for each CAMI microservice
- Implement the message routing infrastructure ensuring reliable transmission of messages on the publish/subscribe channels, as well as appropriate data retention policies

The above features allow for a good separation of concerns since individual microservices can be implemented independently, as long as they provide a specification of their output

message model and implement appropriate filters for the inputs they require from other CAMI microservices.

Another advantage of the message-based communication between CAMI microservices is that they enable an **event-based decision making process**.

Inputs from health and environment monitoring sensors can act as trigger events for algorithms that infer current user activity, health status or mood.

When inserted back into the system as events, the result of these inferences become new triggers for decision making algorithms (e.g. case-based or rule-based reasoning) which set up reminders, generate recommendations or perform rescheduling of planned user activities (e.g. physical exercises).

In the next section we analyse each of the major CAMI architecture blocks in more detail, highlighting the specific functionality of the microservices involved in each block.

5. CAMI Modules and Protocols

This section explains the different architecture components of CAMI, down to the level of the solutions used for implementing various functionality.

5.1 Sensor Unit

The CAMI eco system includes various health monitoring sensors, environmental sensors, physical exercise monitoring sensors and fall sensors. The term “Sensor” is used in broad to describe all components that will provide input to the CAMI system.

The sensors hereby described in detail by device class, usage scenario and possible interface components are the list of sensors that the CAMI system will support in the initial release.

Sensor classification by functionality (i.e. identify exactly the sensors that will be used based on the scenarios in Section 2)

- Health measurement sensors
 - Wearable
 - Fall sensor
 - Fixed
- Physical exercise sensors
 - Mobility problems are very common for elderly people and lead to reduced quality of life and limitations in everyday activities and social participation. However, there are some (especially age-related) factors that induce a progressive diminution of physical activity: the discomfort caused by some health problems, the lack of enthusiasm (apathy), social conformism. Moreover, these factors act as a negative feedback system (this means that the less physical activity someone performs, the bigger these factors become) and lead to a chronic sedentariness. Most existing exergames are made for

children or young adults, and they have many features that do not suit the elderly. Exergames for elderly should follow basic requirements for games, adapted for the abilities of the users. We'll use Microsoft Kinect sensor because it offers all three types of information that is needed in such an application: infrared video stream, depth stream, skeleton tracking. The color video stream can be ignored due privacy concerns, since there are voices that warn about the importance of hiding unnecessary details such as clothes, face, and environment.

- These physical exercises must be defined in order to guide, monitor, and encourage elderly people to perform them:
 - The discomfort sensation can be attenuated by choosing adapted exercises, in accord with physical capabilities, and by progressively increase the difficulty - we'll have three type of exercises: low, moderate and high. The exercises will be based on a set of movements of the whole body representing standard activities concerning physical training exercises (e.g. straight arms pumps: lateral and upper, lateral arm circle, lateral lunge, squats);
 - The apathy can be surpassed using adequate visual and audio stimuli: virtual animated models, rewards, encouragements, musical sequences, etc.;
 - The level of the exercise will be adapted permanently in speed and also the level of the exercise may be decreased based on the user's medical parameters.

- Home monitoring sensors

Sensor classification based on necessity within the project:

- Mandatory sensors
 - Support for at least one specific device, needed by a given scenario are implemented. Alternate devices providing the same / similar functionality in the scenarios, are also considered mandatory if the end-user actions for usage follow a different pattern.
- Optional sensors
 - Device supported if covered by existing gateway service needed for other mandatory device support. Integration to the full CAMI ecosystem depend on implementation done for other mandatory devices.
- Optional commercialization sensors (i.e. those that will be considered only during the subsequent commercial phase of the project)
 - Support for these Sensors / Devices are important for the commercialization of CAMI and implementation in the CAMI research project will be done under consideration of the given time constraints.
 - The general commercial aspects of specific sensor / device support are a

topic of WP4 and will cover all sensors / devices – Mandatory and optional for the CAMI research project.

5.2 Data Collector Unit

Interfacing a wide range of specific sensors/devices to the CAMI ecosystem are delegated to “gateway services”, either implemented as physical devices or micro services running on the CAMI Gateway (see Section 5.4). The gateway services handle all low-level hardware, communication and protocol specific details of interfacing a specific device. It is also the responsibility of the gateway service to enumerate devices and provide device specific setup and configuration interfaces when needed. Moreover it acts as an intermediate layer aiming at clearly separating the sensor/devices context from the rest of the CAMI ecosystem, thus increasing its modularity and very loose-coupling character.

Within the project consortium, a gateway solution that falls well within the scope of the CAMI ecosystem has already been developed and will be reused within the project.

The EXYS9200-SNG (Sensor Network Gateway) is the evolution of the NITICS project’s gateway and data collector. It is a member of the EXYS9200 family of networking devices. Figure 8 shows the EXYS9200-SNG gateway:



Figure 6 The EXYS9200-SNG

The device is based on a micro-PC embedded platform, offering fast processing capabilities (it is equipped with a dual-core processor) and local data storage, USB connection, Ethernet and Wi-Fi (802.11) connection, as well as Bluetooth and Z-Wave adapters. The EXYS9200-SNG is also equipped with a GSM module ready to be used by plugging an external GSM/USB adapter (shown in Fig. 6, and delivered with the system as an optional). In the frame of the CAMI project, the EXYS9200-SNG gateway can act as an interference-shielded, highly performing interface gateway between the raw data coming from the devices/sensors and the CAMI box. It is currently supporting the Bluetooth protocol (IEEE 11073), that is well established and widespread in the eHealth community, the Z-Wave protocol which is likewise widespread at the Home Automation (domotic) market level, and finally the 802.11 (Wi-Fi) protocol. The architecture is modular, so that is relatively easy to add software (and hardware) functionalities. For instance it is foreseen in the near future to add the ZigBee support, for both the hardware (adapter / module) and software ([IEEE 802.15.4](#) protocols stack). The EXYS9200-SNG has also local processing capabilities and is equipped with a performing local database server (MongoDb, a very fast no-sql database).

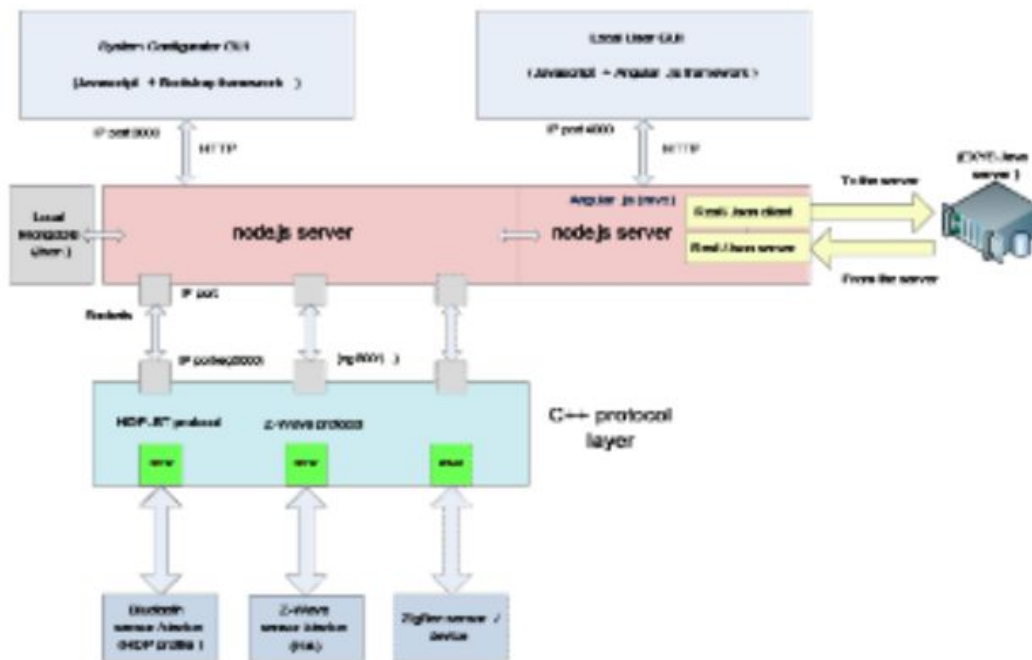


Figure 7 The EXYS9200 system software architecture

Figure 7 presents a description of the software architecture and functionalities.

The bottom C++ layer controls the processing and communication logic with the physical devices and sensors. For every device type a specific driver and communication protocol is implemented. Currently a Bluetooth v.3 driver with the HDP-BT protocol is implemented for the Health-BT devices; similarly, the Z-Wave driver and protocol is implemented for the Home Automation (HA) devices. The mechanism for the protocol to gather data from the devices is a polling mechanism. The C++ layer in turn exchange data with the above node.js layer via dedicated IP ports, one for each device protocol. The blocks signed in dashed lines indicate the possibility to extend the system by adding new hardware/software functionalities (e.g. ZigBee, see section 5.1). The gateway is actually high modular.

The node.js server layer is an intermediate layer having three main tasks: first, it passes device information coming from the C++ layer, to the above web GUI applications; second, it acts as a RESTfull client/server communicating Json messages to/from the NITICS Core Server; finally, it persists information to a local MongoDB database. In fact, this layer is composed of TWO node.js servers, one acting as a RESTfull client/server, and serving the System Configurator, the second acting only as intermediary in passing messages from/to the User GUI. This avoid redundant implementation.

The two applications, also developed at EXYS premise using innovative JavaScript + Html5 + Angular.js technologies, and laying on top of the software stack are web applications accessible from a common web browser (Google Chrome, Mozilla Firefox, Microsoft IE, ...). The System Configurator GUI allows the operator to setup and manage the gateway and the devices. The Local User GUI allows the user to configure device parameters, to check device information and to send device messages (data/commands) to(/from) a central server (currently the EXYS Java server).

The Data Collector Unit forwards retrieved information to the CAMI Gateway, which implements the core CAMI infrastructure that enables communication between all the plugged in CAMI micro-services. Communication with the CAMI Gateway will be performed via a RESTful webservice interface or a direct web-call to the sensor channels of the Event Stream Manager detailed in Section 5.4.1.

5.3 CAMI Gateway

The **CAMI Gateway** is foremost a collection of software modules that implement the core infrastructure of the CAMI system. Its purpose is to enable the easy interconnection of the micro-services that provide the main functionality modules of CAMI (the ones described in the introduction: e.g. sensor data collection, intelligent short-term event processing, forwarding of shareable data to CAMI cloud services).

Together with the Data Collector Unit described previously, we envision the CAMI Gateway deployed physically as a mini-PC, installable in an end-user home.

Consideration of a “box” unit is motivated by the fact that end-users need not have a laptop or PC on which to install the CAMI system. Also, a box unit is simpler to setup from the target user perspective, they already have an expectancy around that: someone comes in to install

and configure the box, and then it runs as long as it's plugged. This builds on the psychological model created by Internet Service Providers (ISPs), that everyone is familiar with.

We envision that a minimum CAMI installation will feature the “box” and a tablet (acquired with the box at a discounted price or pre-existing) to act as a display for the graphic user interface display and as a video communication device (for telepresence).

From the software deployment perspective, we intend to modularize the software running on the CAMI Gateway in self-contained micro-services running isolated from each other, and communicating through message queues. To this end, we chose a light-weight virtualization solution ([Docker](#)) that has the following advantages:

- has very strong industrial adoption (the company behind Docker has raised over \$180M in venture capital to fuel its growth from internationally-acclaimed investors)
- the difference of performance between running a service as a Docker container and running it natively is negligible
- Docker containers are an easy way to replicate locally any functionality. It's extremely handy to move a container from a server to another. From the development process perspective, we envision creating a set of Docker containers locally, prior to deploying them on the CAMI box

In accordance with the scenarios in Section 2 and the main functionality modules presented in the introduction, a typical CAMI Gateway deployment hosts the following microservices:

- Event Stream Manager: part of the core infrastructure solution enabling message-based interconnection between all the other micro-services of running on the CAMI Gateway
- Local Data Storage: a MySQL based short term storage solution for data collected from sensors and situations inferred about the user by the collection of decision support algorithms
- Decision Support System: a collection of symbolic and data-driven reasoning algorithms that continuously monitor the short-term state of the user (current health status and mood, current and planned daily activities, required reminders etc).
- Voice Command Manager: service implementing voice-based interaction with the CAMI system.
- Communication with 3rd party Health Platforms: service allowing the sharing of selected health measurements and physical exercise sessions with primary and secondary caregivers
- Connection with the CAMI cloud: replication of locally collected data to the CAMI cloud platform (see Section 5.7) for longer term and higher level processing, as well as communication with the CAMI cloud to retrieve the result of performed analyses or suggested actions (e.g. context-aware rescheduling of planned activities, physical exercise recommendations).

In what follows we provide additional detail about the above mentioned services.

5.3.1 Event Stream Manager

The Event Stream Manager is a central part of the CAMI skeleton outlined in Section 3. It is the service that enables the CAMI system to connect all its various microservices with one another in a custom and scalable manner.

The Event Stream Manager collects all the messages/events created by CAMI microservices and forwards them to specific publish/subscribe channels from where other CAMI modules that are interested in consuming the messages can retrieve them.

For each channel the Event Stream Manager defines specific message envelopes and handles routing policies relating to time-to-live and persistence, delivery acknowledgements, publisher confirms etc.

Within the CAMI architecture we envision the following message/event channels which correspond largely to the six major CAMI architecture blocks:

- Health Measurement Channel
 - Handles messages from wearable or static health measurement sensor considered in the CAMI system (see Section 5.1). This is the channel to which the Data Collector Unit (see Section 5.2) will publish its messages. This includes fall detection events, which will be given special preference within the channel.
- Physical Exercise Channel
 - Handles messages both to and from the Physical Exercise Service which enable the CAMI system to configure existing sensors (e.g. a Kinect camera) at the start of an exercise session (command messages), as well as to monitor the execution of the exercise by the user (e.g. using a Kinect camera and other wearable health measurement sensors).
- Environment Monitoring Channel
 - Handles messages from the environment monitoring sensors described in Sections 5.1 and 5.2.
- Health Professional Communication Channel
 - This channel is used to publish health related messages that need to be shared with health professionals. Health data will have been priorly retrieved from the Health Measurement and/or Physical Exercise Channels by the Decision Support Service (see Section 5.5.1) and brought to a message format that is internally consistent across the whole CAMI system. From this channel, 3rd party cloud services such as OpenTele, Microsoft Healthvault or ehrScape will be able to listen for messages and forward them to online platforms that enable viewing and analysis by health professionals.
- User Interface Input Channel
 - Handles messages representing commands and triggers issued by the user in

his interaction with the multi-modal CAMI user interface. The channel will provide a means to structure user commands that are issued through various means: the CAMI GUI on display devices (e.g. tablet, smartphone), voice activation or gesture observation.

- User Interface Output Channel
 - Handles messages that represent information, notifications or actions that need to be communicated to the user through the multi-modal user interface. This means that services such as Display Management, Voice Command and Telepresence will be subscribed to messages on this channel.
- Decision Support Channel
 - Is a central part of the Event Stream Manager and, by extension, the CAMI Gateway. The channel handles messages that are consumed and produced by the Decision Support Service. The latter service consists of several interworking techniques (see Section 5.5.1) which allow the CAMI system to make sense of the user's needs, activity, health and mood status. The channel will essentially act as a temporary storage for messages and events that enable the incremental inference of high-level knowledge and actions (e.g. activity or mood recognition, reminders, rescheduling) from low-level data (health and environment data, issued user commands).

We are planning to roll out an implementation of an event stream using [RabbitMQ](#).

RabbitMQ is a messaging broker, giving applications a common platform to send and receive messages and perform safe storage until they are received. The service allows for very flexible routing, allows for easy monitoring and tracing and supports a large number of API clients in many programming languages. RabbitMQ is an open-source platform used by many popular services to date (e.g. Reddit, Instagram, Soundcloud, Salesforce).

One significant feature is the ability to define custom plugins for message routing logic, which will prove particularly useful in the context of the CAMI system. It also features configurations for durability (keep messages across restarts and even some types of crashes) and reliability (optional acknowledge that a message has been correctly consumed on the receiver). These two make it an ideal choice for healthcare apps, where we want messages to be received properly and not lost in the improbable case of an unfortunate event.

5.3.2 Decision Support System

CAMI system objective is to improve quality of life for elderly people through combination of advanced reasoning algorithms and voice recognition to ease self-management and decision making in different episode of disease. With this regard, CAMI will act as an intelligent, informed, friendly collaborator that can offer recommendations, reminders through vocal interfaces to support people in their daily activities, learn about risk factors and detect anomalous situations (either health concerns or activities atypical in nature).

Thus decision support systems (DSS) are crucial in the design of CAMI. Such systems can be simple software systems or a complex system with an associated Knowledge Base.

As shown previously, the DSS service itself will be composed of different communicating microservices that can be implemented independently by consortium partners leveraging their expertise in a given domain (e.g. recognition of activities of daily living, health measurement anomaly detection).

The existence of the Event Stream Manager ensures that one DSS microservice can leverage the output of another as its own input. This leads to intelligent decision making that is *flexible*, *updateable* and that can be done in an *incremental* fashion.

The architecture of the DSS service, shown in Figure 8, presents the envisaged flow of information to and from the DSS. As described by the CAMI architecture diagram, the decision making is done with the local and cloud based Decision Support systems with redundancy support. In order to ensure that all critical decision making is processed in real-time and without any dependency with internet, all the critical decision making lies with the local DSS. This also ensures that non-critical decisions do not interfere with critical decision making.

The DSS includes a knowledge base (KB) and inference engines that support decision making. The knowledge base contains all information regarding the events and contexts, activities of elderly, health measurement, user preferences and also suggestions from health care professionals. The latter can update the content of suggestions and customize parameters that help detect critical situations using the CAMI cloud (see Section 5.7). Section 5.3.3 explains how these changes are retrieved locally, in the DSS running on the CAMI Gateway.

The useful sensor data like health parameters can also be stored in local MySQL DB (Section 5.3.4). The sensor information can be used to generate an environment model and other models which help in context-awareness as well as for detecting any anomalies in usual events. The location and time information is also needed to understand the criticality of the event.

One noteworthy aspect of the DSS architecture in Figure 8 is the combination of two types of decision making methodologies are being combined, a *data-driven one* and a *symbolic one*. Data driven approaches can be successfully used to determine activities of daily living by analysing data from various sensors to detect the event.

Then, if the event is critical, decisions are taken by inference engines based on *rule-based reasoning and case based reasoning* (a symbolic decision making method) which has the big advantage of leveraging existing domain knowledge, as well as enhancing *comprehensibility of a decision system*.

The decision from the DSS can be stored in the KB, reinserted in the Event Stream Manager (to be exploited by other decision making microservices in an incremental fashion) and also sent to any other interfaces like mobile, telepresence and vocal interface.

It is worth mentioning that the DSS on the CAMI Gateway is designed to work with short term information (i.e. collected throughout a given day).

All the events collected by the Event Stream Manager are also forwarded to the CAMI Cloud. An implementation based on the Microsoft Azure Event Hub [5] and the Linkwatch Platform¹ [14] ensures the ability to make both deterministic (i.e. rule-based), as well as data-driven analyses of end-user events over a longer period of time.

Circumstances of particular health situation, particular choice of medication, a care plan or an inferred observation as a case that may lead to another decision will be mapped in set of decision rules that will reach conclusions such as alert, reminder and recommendations.

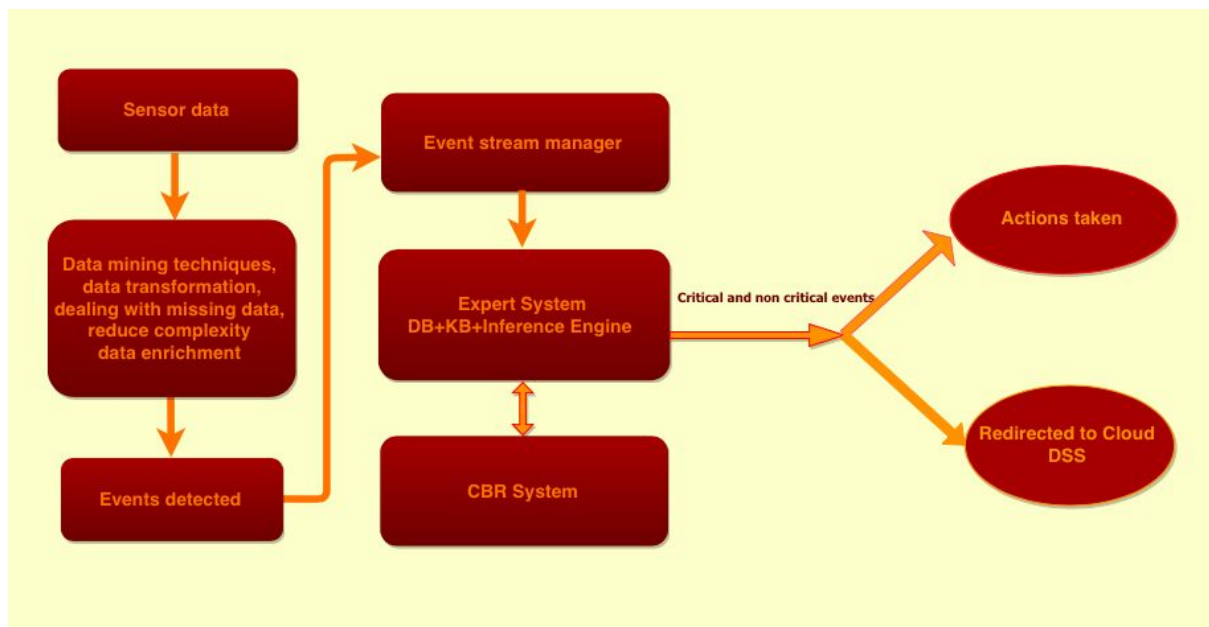


Figure 8 DSS architecture

5.3.3 Reverse Connection Manager

The role of the reverse connection manager is to establish a 2-way connection between the cloud system and the local system.

Why do we need a reverse connection from the cloud to the local box? Mainly in order to push the results of complex computation offloaded to the cloud. This follows the model of established companies like Adobe and Autodesk that are moving their most precious computation in the cloud, charging a monthly fee for it. In business parlance, this is called Software as a Service (SaaS). In the particular case of CAMI, we can consider that the most valuable data analysis will be offered as a SaaS. For the end-user, this will reflect in having the following advantages:

- CAMI as an organization does not need to deploy significant computation power in

¹ <https://linkwatchrestservicetest.azurewebsites.net/Help>

the home of the user. This follows the philosophy of the throw-away Chromebook, which is currently having immense success in education

- CAMI as an organization keeps its most valued algorithms in the cloud. This allows it to update and monitor them easily. The alternative would be to have dedicated personnel for on the premise visits in order to update the software. Also, monitoring-wise, the costs for monitoring a cloud solution are much lower, as there are well established strategies for dealing with failure in the cloud.

In this organization of responsibilities, the information captured from the sensors will only be stored locally for a limited amount of time (ephemeral) and will be replicated to the cloud for longer term/complicated analysis.

The reverse connection manager is needed in order to stream the results of the CAMI cloud computations back to the local application. To this end, there are 2 main solutions that can be used:

- A reverse SSH tunnel from the local CAMI box to the CAMI cloud
- A websocket opened from the local CAMI box to the CAMI cloud

In our opinion, the second solution would be much better suited, and is currently also used by Slack (www.slack.com), a very popular messaging app for connecting its Desktop client to its cloud infrastructure.

5.3.4 CAMI Box MySQL DB

For storing data both locally and in the cloud, we've chosen MySQL, an enterprise-grade storage solution that is open source and widely deployed across the globe.

Since in the introduction to Section 5.3, we've described the division of responsibilities between the CAMI Gateway and the CAMI cloud, it begs the question: how does the data from the local MySQL arrive in the remote MySQL?

To this end, we've chosen MySQL replication via GTID (Global Transaction ID), a standard mechanism for transporting changes from one database server to another.

From a privacy perspective, a very important aspect is that every CAMI client will have their own separate, isolated database, so that in no case, due to any programming error, one user can access the data of another user.

Even though GTID replication is not the only solution to do this (we can also do it via RabbitMQ, for example), we believe that this is the most robust way to do it. There are also a number of tools in the open source ecosystem available to monitor the replication process (the so-called *replication lag*) and alert the persons in charge with the CAMI cloud infrastructure about a potential problem.

If the data in the cloud isn't fresh enough, wrong conclusions can be pushed back via the Reverse Connection Manager to the CAMI box, rendering the system useless for the end

user.

5.4 User Interface

Traditional human-machine interfaces were always a barrier between non-technical people and any new device generally and between elderly people and any new device more particularly. CAMI's interface should be easily accessible by users and it should interact with them in ways that are most natural to them. Therefore, a multimodal interface should be implemented into the CAMI system. Compared to other types of human-machine interaction, multimodal interaction offers users a more natural way of interacting with the machine.

Since people prefer to use easy and natural interactions, CAMI's multimodal interface integrates vocal and touch-based interactions. Furthermore, CAMI's interface is responsive, and is intended to work on different devices and to adapt to any display size or orientation of a device.

Figure 9 shows the input and output options for the CAMI multi-modal User Interface. As discussed, input options include keyboard, touch-based devices (tablet, smartphone) and voice commands. The output options include a graphical user interface, text-to-speech (i.e. voice replies).

A special part of the CAMI user interface is the interaction with the robotic telepresence unit, seen as a powerful add-on to the conventional UI options. The robotic unit can be used for both input and output interactions and is furthermore capable of actuation (e.g. movement). We dedicate Section 5.5 for a more detailed overview of envisioned interfacing with the robotic unit within CAMI.

The remaining subsections discuss the graphic and voice based interaction methods.

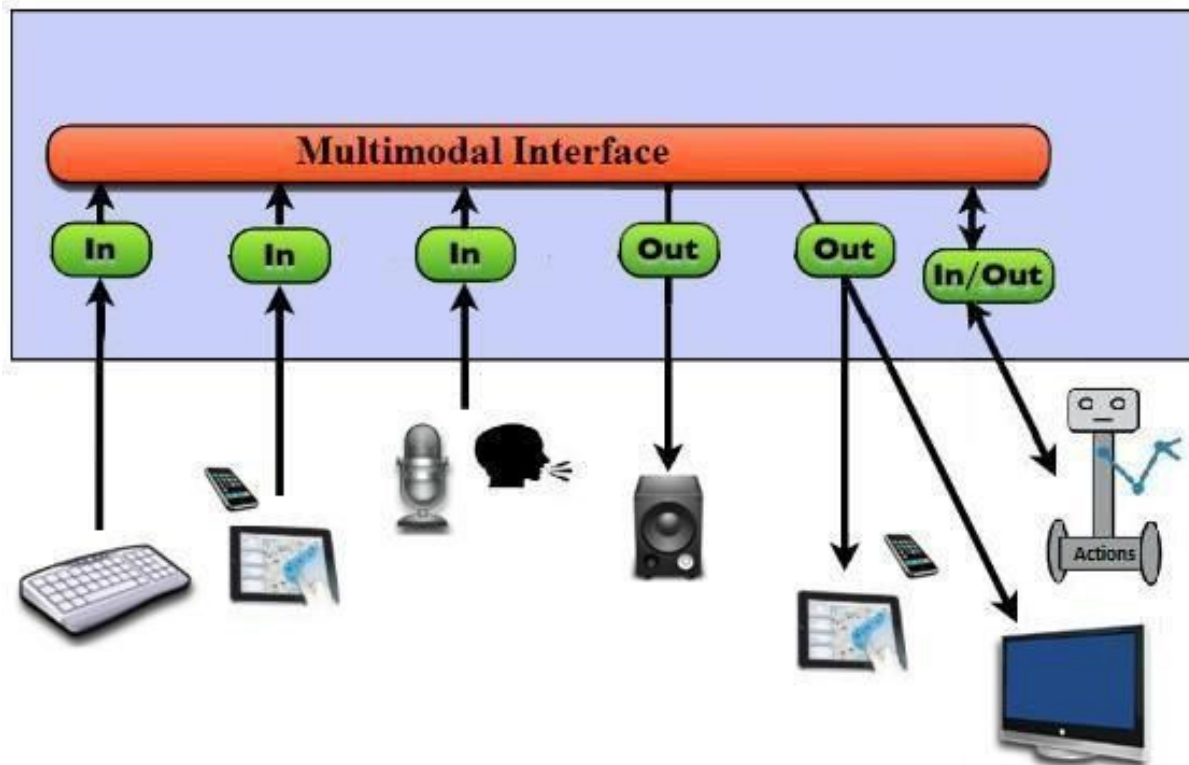


Figure 9 CAMI User Interface input and output options.

5.4.1 Vocal Interface

Oral communication makes the process of conveying thoughts easier and faster, and we consider it an essential part of CAMI's interface, specifically for senior citizens.

In the multinational survey conducted as part of the CAMI project, the interviewed seniors have expressed interest in voice-based interactions, but have also indicated desire to be able to revert to classical interfaces (keyboard, touch screen), presumably citing previous unsatisfactory voice based interactions.

For this reason, the intention in CAMI is not the development of a general purpose speech recognition and dialogue engine. Rather, the focus falls on establishing simple command/query patterns that are very likely to be used in the context of AAL and Ambient Intelligence applications. Some typical examples include the following:

"CAMI: **List activities** for **today**."

"CAMI: **Show medication** for **this evening**."

"CAMI: **When** is my **next exercise** **scheduled**?"

"CAMI: **turn off** the **lights**."

As can be seen in the example above, the color coding corresponds to three different entity types: blue covers a type of **action**, green covers the *object of the action/intent* and orange covers a time reference.

The idea behind this type of command parsing is that the mentioned entities can be easily converted to a well defined message format that can be inserted in the User Interface Input Channel of the Event Stream Manager.

The definition of well defined message forms for both input commands, as well as their output means that the CAMI system can manage an interweaving of touch and voice based interactions (i.e. an initial touch based menu navigation can be followed by a voice command and result in a speech-based answer), thus effectively enabling multi-modal user interfacing.

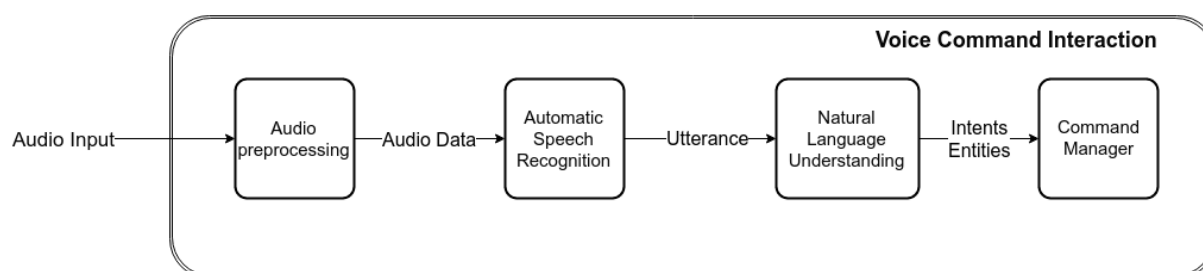


Figure 10 CAMI Voice Command Manager Block Architecture

Figure 10 displays the Voice Command Manager Architecture and contains four main blocks: Audio Preprocessing, Automatic Speech Recognition (ASR), Natural Language Understanding (NLU) and Command Manager.

Audio Preprocessing is done to ensure the best result for the ASR submodule. This includes aspects such as setting the proper bit rate, bit depth or number of channels for the audio input.

Automatic Speech Recognition enables the detection and use of speech as a method of interaction. The outcome of the ASR submodule is the *text* form of the spoken utterance.

The essential requirements of the ASR submodule are the creation of appropriate phonetic and language models that will cover the set of commands expressed by the user.

In deliverable D2.1 several frameworks and services had been identified as possible candidates for implementing voice command functionality.

The ASR submodule can be developed using any number of the analyzed technologies. Several APIs such as Google Speech API [6], Amazon Alexa Voice Service [7], Wit.ai [8] or Microsoft Bing Speech Recognition [9]. The implementation effort will be an iterative process and during the project an effective comparison of the efficiency of the above mentioned services will result as a secondary outcome.

Furthermore, as it is less likely to find support for languages such as romanian and polish in the aforementioned APIs, main focus falls on validating the voice command pipeline using the english language. An in-house ASR module, developed using the CMU Sphinx, will be considered towards the end of project lifetime or during the project commercialization phase.

Natural Language Understanding receives the plain text utterance obtained from the ASR to determine the *intent* of the user and extract the relevant entities, which are wrapped under a command form and forwarded to the Command Manager.

We have already used the Wit.ai [8] API with appropriate success on a sample of queries/commands such as the examples listed previously. For instance, in the utterance *CAMI: Show medication for this evening.*, Wit.ai will return a JSON output specifying that the recognized intent is Medication and it will identify two entities: action = “show” and time = “this evening”.

The Wit.ai service furthermore allows defining roles for each recognized entity. For example for an object entity of type blood pressure, two numeric values (systolic and diastolic) can be defined and used in the recognition process.

Wit.ai currently works well on utterances in english. Like for the ASR, development of the NLU submodule will progress in an iterative fashion and support for languages such as romanian, polish or danish will be investigated towards project end.

For the romanian language, results from the ANVSIB project [11], under development at UPB, can be applied for both ASR and NLU.

The output of the NLU submodule is the recognized command or query, transformed to the internal message format used by the User Interface Input Channel. The Event Stream Manager ensures that all commands are received in order such that the Command Manager (which is subscribed to the channel) can process them appropriately.

5.4.2 Graphic User Interface

The Graphic User Interface in CAMI serves as the traditional means to convey access and control over the collected information and running services.

In CAMI, the envisioned GUI has three form factors: a web interface which is adapted for PC/laptop screens, as well as tablet screens and a native smartphone interface.

Touch screen must make users feel comfortable that they cannot “do anything wrong”. It should give them a feeling of instinctively understanding how to use the interface since reaching out for what you want is an instinctive gesture.

The web interface foremost addresses the need for control and ease of access of end-users (the elderly) to CAMI provided services such as telepresence, home monitoring and physical exercise execution.

Furthermore, the web-based GUI provides views for summarization of medical status and history of recent activities, in a form that is easy to understand for senior citizens.

The smartphone application has two intended versions: one for the caretakers (elderly users) and one for the informal caregivers (e.g. family, friends).

The focus of the smartphone application lies on addressing the type of interactions between CAMI and its users which have a higher frequency of occurrence. Specifically, this refers to notifications, recommendations (e.g. to rest if increased pulse is observed for too long a period), reminders (e.g. for medication intake) and alerts (e.g. in case of a fall event).

The smartphone application functions differently according to the target user.

Elderly users are expected to have less frequent interactions with their phone than younger ones and be less inclined to *actively* check the information contained in the CAMI application. This is why the main functioning mode of the end-user oriented CAMI smartphone application will be *notification-based*.

That is, the application must ensure timely, sound-based notifications for reminders and recommendations issued by CAMI. Notifications from different services (e.g. medication reminders, physical exercise reminders, suggestions based on health status) must be easily distinguishable by end users, by using an icon or color-coded based representation, as well as accompanying sounds.

The graphic interface for end users must further provide buttons that allow for easy *acknowledge/cancel/postpone* means.

In order to display notifications in a timely manner, the smartphone application takes into account the existence of the CAMI Event Stream Manager, from which it will receive its notifications, as they get processed by the DSS.

The application for informal caregivers, on the other hand, will provide a richer interface, which gives the ability to view a log of the elderly person's activity (e.g. acknowledged medication reminders, executed physical exercises, recent health measurements), computed over different periods of time (e.g. daily, weekly, monthly trends), as well as alerts which require immediate contact of the caretaker (e.g. notification of forgotten meds, fall alarm).

This is based on the assumption that caregivers such as family constitute a younger group of users which are used to actively check applications running on their smartphone.

Existing partner graphic interfaces

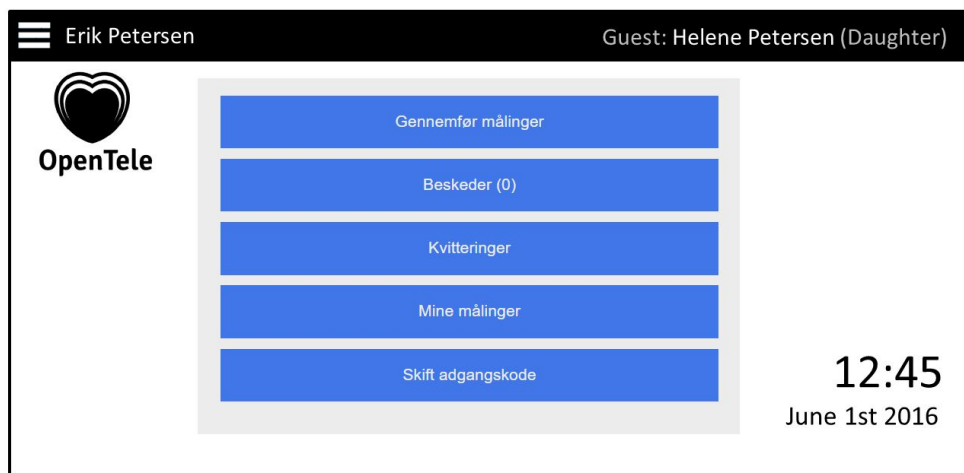


Figure 11 OpenTele client application main menu (interface in Danish).

Within the consortium partners, web-based applications for telemedicine or health monitoring have been developed. The noticeable platforms are OpenTele and LinkWatch (see Section 5.6 and Figures 11 and 12).

The CAMI system overall integrates a larger set of functionality than the above mentioned platforms, but it may very well be the case that potential CAMI users may already be accustomed with the visual interface provided by these applications.

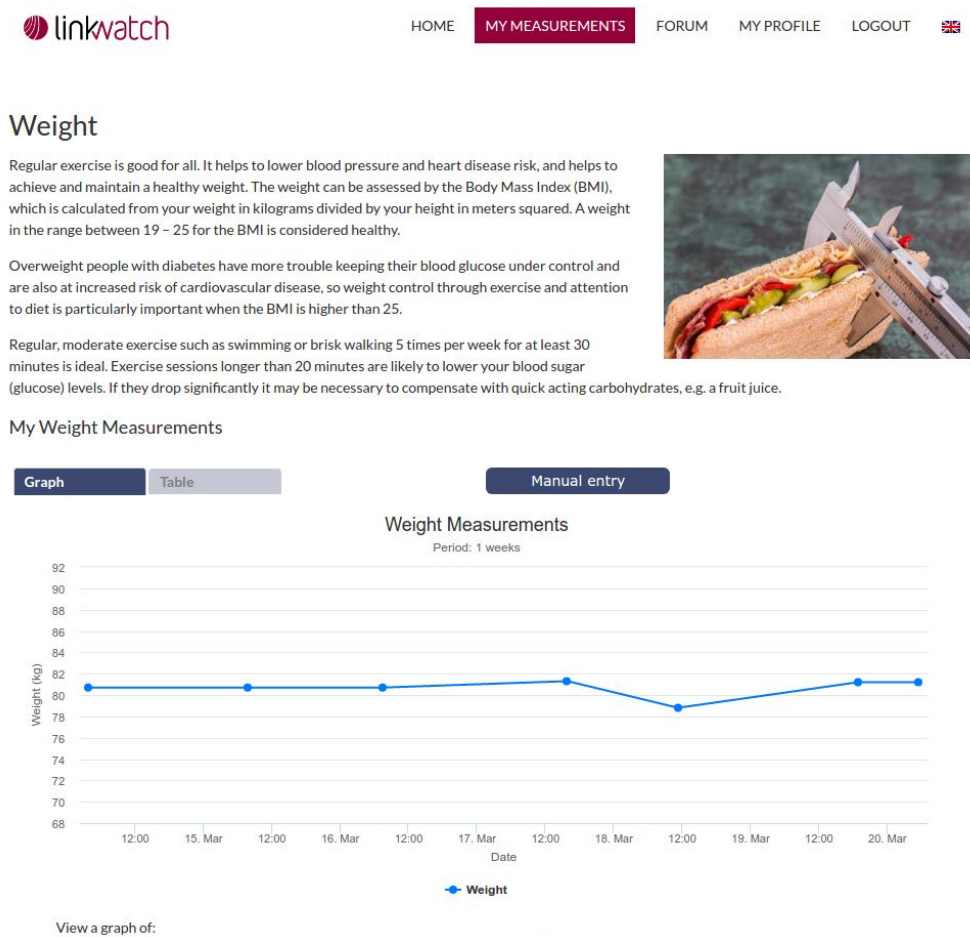


Figure 12 Screenshot of weight overview interface in Linkwatch platform

Thus, the CAMI System will ensure that measurements and events collected within the system are replicated in these platforms, to be viewable there as well by users who are already very familiar with the particular interfaces. The data integration is performed using the RESTful APIs provided by both OpenTele and LinkWatch (see more in Section 5.7).

5.5. Telepresence

The system can either be a tablet-like system or can be a mobile robot with manipulation capabilities. In the initial CAMI architecture version, we are restricting the robotic telepresence unit to a tablet, which function just like another interface to the CAMI box, but we will use a ROS compatible systems to ensure further extensibility of the framework by adding service robots. We will include the communication interfaces, user interfaces and supervised physical exercise modules in the tele presence system. Kubi and Tiago are robotic telepresence systems are the potential candidates that could be used in CAMI. However we need to check on the ROS compatibility of these solutions as well as on the fact that how these solutions could be used in our CAMI as *totally* different means of ensuring telepresence support.

5.6. Communication with caregivers and health professionals

The area of computer intelligence created new possibilities for retrieving, storing, viewing and analyzing the information within the medical record, by changing the physical nature of records to an electronic format. To enable their communication between different information systems standards should be implemented and platforms should function across different types of devices.

During the last three years a massive investment has gone to create standards into this domain. During our research, we had identified the following standards:

Health Level 7 (HL7) provide a framework (and related standards) for the integration, exchange, sharing, and retrieval of electronic health information. The standards define how information is packaged and communicated from one party to another, setting the language, structure and data types required for seamless integration between systems. Those standards are recognized as the most commonly used in the world.

OpenEHR is a virtual community working on techniques of turning health data from the physical form into electronic form and ensuring universal compatibility among all forms of electronic data. The openEHR approach is multi-level, single source modelling within a service-oriented software architecture, in which models built by domain experts are in their own layer. Technically it is a platform approach, rather than a set of standards or monolithic specification or product. It offers the most comprehensive semantic framework available in ehealth, combining formal clinical modelling, terminology, and a services infrastructure; It deals directly with the very difficult challenges of e-health; It ensures that the customer retains control and ownership of the data; It is easy to understand and well documented; It is also easy to start developing with it (available downloads, demonstrator server sites and SDKs); It can be integrated with HL7 FHIR.

The communication of health records with caregivers and health professionals is an important function of any Ambient Assisted Living system. In CAMI project, it has been considering to allow users to share their own personal health record with health professionals and trusted people.

In Europe, several countries adopted their own laws and customs, and according to them they shape telemedical regulations. Therefore, CAMI eco system will have his own solution to share data between his different components and it will bear third-party plugins to share the user's health record with caregivers and health professionals based on existing standards. The third-party plugin will vary in function of user's country to ensures its fitting with the local laws.

While the previous approach is commercially crucial for the CAMI eco system, it will impose some constraints in using the full functionality of CAMI eco system. The CAMI research project will implement examples of this type of 3rd party interfacing.

Internally on the CAMI box, health measurement data will be stored in a common format alongside with the same data stored in formats dictated by a given 3rd Party plugin requirements.

Communication to informal / non professional caregivers (relatives, friends etc.), will use a cloud based approach (Web services / Sites) with the possibility of SMS and/or email based alerting. As an example of this the CAMI project has considered to allow user to share their own personal health record to the relevant stakeholders.

During our research, we have identified the following solutions that can be used as a third-party plugin for the communication with caregivers and health professionals part:

OpenTele is an open source telemedicine ecosystem developed in Denmark by Silverbullet for handling data and measurements from personal health devices. It has become National Reference standard and was CE certified earlier 2016. The platform allows people to gather, store, use and share the collected health data. It allows the communication between a health professional and his patients. Furthermore, the server will forward measurements to the KIH Database for integration with other clinical systems. It is based on standards such as OIOXML, Continua Health Alliance and HL7.

Microsoft HealthVault is a cloud-based platform designed to help people to gather, store, use and share health information with trusted people and health professionals in a private and secure environment. HealthVault enables a connected ecosystem that currently includes more than 300 applications and more than 80 connected health and fitness devices (vary by country). The HealthVault data type system supports industry standards (such as CCR and DICOM). HealthVault supports a variety of SDKs for common platforms however HealthVault provides an XML-based web service API.

HälsaFörMig (or HealthForMe) is a secure and interactive health platform that allows users (Swedish-users) to view, collect and store data in their personal health record. This solution is approved by the Swedish government.

LinkWatch is an end-to-end solution for remote patient monitoring, it gives the possibility to patients to communicate with their caregivers, friends and also their kin. Linkwatch API- can provide interface on different type of languages such as German, Danish, English and Swedish.

5.7 Cloud Services

The CAMI System targets the integration of several sets of functionalities into a coherent framework, as well as the ability to integrate itself with existing 3rd party platforms.

The integration and remote communication need spans over many of the services proposed in CAMI: collection of data from sensors, sharing of information between caretaker and caregiver applications, as well as with other health monitoring and telemedicine platforms (see also Section 5.5), long-term analysis of health and activity trends.

Consequently, the CAMI System requires the existence of a cloud module (see Figure 7), which addresses the above mentioned communication and integration requirements.

Sensor data collection

Several types of health monitoring sensors considered in the project (e.g. weight scales from Withings, smartwatches, wearable bands) have a policy of sending their measurement data directly to the cloud (e.g. Withings cloud, Google Fit or iHealth, Jawbone cloud). Access to that data can then only be made via API calls to the respective cloud endpoints.

This, of course, entails the setup of user accounts and API access credentials for each individual type of device.

The management of such accounts and credentials in CAMI is best done using an online platform, since it enables much easier overview and maintenance of created accounts capabilities for the support staff of the CAMI System.

Sharing of information

Section 5.5 explains that within CAMI there are separate smartphone applications for elderly users (end-users) and for their informal caregivers. Furthermore, 3rd party platforms, such as OpenTele or LinkWatch, are considered as additional web-based interfaces where data collected by CAMI can be visualized.

In order to keep end-user and caregiver applications, as well as remote platforms in sync with the data collected for an end-user, the communication between the application instances needs to occur at cloud level, via a web-based protocol.

In this case, the CAMI cloud acts as a known reference for all client applications.

The synchronization mechanisms between the local CAMI Gateway and the CAMI cloud have been explained throughout Section 5.3.

Intelligent Services

Section 5.3 introduced the Decision Support Service as an important component of data analysis within the CAMI Gateway. However, since the gateway is conceived as a local device, long term storage of events and collected data is infeasible.

Therefore, long term storage and analysis of user data is performed using the CAMI cloud.

The CAMI cloud MySQL DB receives data from replication of transactions happening in the local CAMI Box DB. The cloud database keeps a theoretically unlimited history of user events in order to provide support for long term user behavior analysis.

The Decision Support Service supports enhancements via its cloud counterpart, since the cloud storage considers a semantic modeling of the collected user data, using RDF storage engines.

Consequently, reasoning methods (e.g. production rules, ontology reasoning) are further supported by performing a conversion of selected data from the MySQL cloud database to an RDF format (e.g. corresponding to a specific custom ontology vocabulary used for activity analysis). Inferences and queries over RDF data become possible using a quadstore engine such as Stardog².

Since they run in the CAMI cloud, the results of such inferences for each user are then accessible for interpretation to informal and formal caregivers, which can approve of the suggested recommendations for smart program adjustment, or add in their own suggestions based on the long term analysis.

Lastly, frameworks such as LinkWatch, which are to be connected to the CAMI System, have a set of cloud based analysis resources (running on the Microsoft Azure platform) which can thus be readily used.

5.7.1 Cloud MySQL DB and RDF Data Storage

- The CAMI cloud MySQL DB receives data from replication of transactions happening in the local CAMI Box DB. The cloud database keeps a theoretically unlimited history of user events in order to provide support for long term user behavior analysis.
- Reasoning methods (e.g. production rules, ontology reasoning) are further supported by performing a conversion of selected data from the MySQL cloud database to an RDF format (e.g. corresponding to a specific custom ontology vocabulary used for activity analysis).
- Inferences and queries over RDF data become possible using a quadstore engine such as [Stardog](http://stardog.com/).

CAMI box can also be acknowledged as CAMI- middleware with type of IoT service catalogues such as:

-Network Manager Service Catalogue, that will be synchronized in-between all the networks of component in a high level of security. However, this solution needs to be employed in

² <http://stardog.com/>

advanced level of cloud technologies and architecture to achieve a cloud hybrid platform based services and open sources.

-The IoT Resource Catalogue that contains more metadata and annotations of the services available by using of LinkWatch gateways. This catalogue provides the means to deliver the information about IoT Entities in the platform and to actuate on them. LinkWatch gateway is emerged from LinkSmart which support variety of e-health standards for exchanging and storage of data (e.g. personal health values) in the cloud SQL Services as well as SQL database. This gateway enables communication with Raspberry Pi 3 as well.

LinkWatch API's new version is well- integrated to Azure -Microsoft hybrid cloud based platform to support integration of different frameworks with variety of modules, databases, language programming (e.g. PHP, Java, Node.js, .NET, Python, JavaScript) and different devices as well as backend for iOS, Android and Windows. It will enable easily integration with Docker and also run Linux containers.

This functionality gives possibility for routing requests such as reporting vital signs values or even air pollution rate while visualizing graph analysis in the interface for end users. This catalogue discovers and keeps track of available IoT Resources in the network. It provides a REST -based interface to select and retrieve data about IoT Resources. IoT Resources are software objects that provide IoT services for applications and end users, e.g retrieving and analyzing data from the physical world, invoking actions and so on. Currently three types of IoT Resources can be implemented by LinkWatch such as: IoT Device, IoT sensors and IoT Thing. IoT Resource provides Services and Action by suing of two other components as IoT Service and IoT Observation.[1]

In order to fulfill requirements of CAMI box and its functional operation, we consider using of LinkWatch cloud based solution as an open source with reliable features for virtual machine, supporting RF technology and Data policy and security as well. This open source solution will be supportive for both time and cost in a highly manner of CAMI deployment.]

6. Scenarios revisited

In this Section we revisit the episodes introduced in Section 2 and perform a breakdown along the set of services available in CAMI, that were described previously. The breakdown helps visualize the nature and sequence of required microservice interactions, highlighting the complexity that the CAMI system helps address.

Episode 2.1 Breakdown - Physical Exercise Monitoring

Actors involved in the episode: end user (Jim) and caregivers.

We envision it requiring a set of independent interacting services, such as: A User Interface Service (UI), Decision Support Service (DS), Physical Exercise Service (PE), Health Data Collector Service (HC), Data Visualization Service (DV), Health Data Sharing Service (HDS), Calendar Service (CAL).

- At start CAL notifies PE of the scheduled exercises.

- PE checks if Jim still need to exercises despite his activity. It decides that Jim has not had enough exercise today and he should do the scheduled exercises.
- PE asks the DS if Jim can exercise (may be prevented by high blood pressure etc).
- DS notifies PE that Jim can do exercises now.
- PE notifies the UI to show an exercise notification and ask for Jim for his confirmation (maybe Jim is not in the mood now). Jim agrees to start exercising.
- UI notifies PE that Jim agrees to begin the exercise.
- PE tracks the exercises and updates the HC to store the data as exercises are being performed.
- PE determines the end of the exercise session and calls HC to store exercise session report. It generates notifications for caregivers following the exercise program and forwards them to the caregiver's EHR system of choice using HDS.
- The Caregivers access their EHR platform to analyze the exercise session report.

Episode 2.2 Breakdown - Medication Compliance and Reminding

Actors involved in the episode: end user (Jim), primary caregiver (doctor) and the notified family member.

The envisioned services that enable this episode are: User Interface Service (UI), Calendar Service (CAL), Communication Service (COM), Medical Compliance Service (MC), Data Store Service (DS).

- Jim asks UI about his prescribed medication plan.
- UI calls MC to ask about Jim's medication plan.
- MC calls CAL to check the schedule and get the reply.
- MC calls UI to notify Jim about his prescribed medication plan.
- Some minutes after a scheduled time for a Medicine, MC determines that Jim hasn't used the pill dispenser (or has not confirmed that he has taken his medicine).
- MC calls UI to notify Jim that he has forgotten about the medicine (if pill dispenser is not used, UI will asks Jim to confirm that he has taken the pills).
- After a while MC determines that Jim still hasn't taken the pills.
- MC asks COM to notify a family member that Jim missed to take the pills.
- MC stores into DS that Jim missed to take the pills.
- COM contact a family member.
- MC checks DS if Jim often forgets to take the pills. MC found that Jim had forgotten to take his pills many times.
- MC asks COM to notify Jim's doctor.
- MC stores into DS that Jim's doctor was notified.

Episode 2.3 Breakdown –Fall detection Scenario

Actors involved in the episode: end user (Jim), primary caregiver (doctor) and the notified family member.

The envisioned services that enable this episode are: Data Collector Service (DC) Communication Service (COM), Decision Support Service (DSS), Data Store Service (DS),

Medical Compliance Service (MC).

- 1) The fall sensors (wearable and non-wearable) use the DC service to report the occurrence of a fall event.
- 2) The DC used DS services to record the event in the event stream manager and the DB.
- 3) The DSS is invoked to take a decision about the occurrence of fall, its criticality and the action.
- 4) The DSS upon recognizing a critical fall uses the MC service to activate the COM service to notify Jim's doctor and family member.
- 5) The MC service stores into DS that Jim's doctor and family member was notified.

7. Conclusions

CAMI is an ecosystem architecture, integrated with almost all the functionalities required in an AAL system. CAMI's objective is to create an artificial intelligence cloud based platform which demands implementing an expert system to handle many rule based algorithms and decision trees. This will consider in relation to have a central point for many type of decision making modules. Thus general requirements from section 5.5.1 may be valid in this document for future development of knowledge based decision system in CAMI platform.

In order to fulfil the purpose of CAMI's objective we consider to allow end users to have free choice to use any platform and any devices. This requires to create a flexible architecture that could support variety of data format and communication protocols with different interfaces based on user's needs to improve their quality of life based on healthy lifestyle.

It will have recommended highly depending on user acceptance based on health conditions and affordance as well as cultural attitude towards using of technologies. Thus, CAMI Platform will be flexible to adapt different languages, modules and friendly interfaces to increase usability of CAMI platform as a meaningful ecosystem for users.

Therefore, cooperation of variety components takes in account based on a comparison to evaluate both strength and obstacle selected components towards full functionality of CAMI platform with respect to these aspects that pointed here above.

The major highlight of CAMI is that we have an architecture that integrates the major functionalities required in an AAL system. The CAMI architecture is highly maintainable compared to distributed agent based framework. All the functionalities of CAMI subsystems are well defined and choice of the communication protocols are also flexible. The major advantages of CAMI also include the introduction of robotics to the AAL framework, which paves the way for huge developments in the area of assisted living. Also, many AAL systems that existed in the past could not be used reliably in areas where network connectivity was

not stable. In CAMI, we have introduced a switch that can interconnect between internet and GSM and hence this major problem is also solved. All the critical functionalities in CAMI is taken care locally without the necessity of internet connection. We also have redundant DSSs

References

1. Murthy SK. Automatic Construction of Decision Trees from Data: A Multi-Disciplinary Survey. *Data Min Knowl Discov.* 1998;2(4):345-89.
2. Fürnkranz J. Separate-and-conquer rule learning. *Artificial Intelligence Review.* 1999;13(1):3-54.
3. Quinlan JR. *C4.5: programs for machine learning*: Morgan Kaufmann Publishers Inc.; 1993. 302 p.
4. Stroetmann KA. Health system efficiency and eHealth interoperability-how much interoperability do we need? *Advances in Intelligent Systems and Computing* 2014. p. 395-406.
5. Microsoft Azure Event Hub:
<https://azure.microsoft.com/en-us/documentation/articles/event-hubs-overview/>
6. Google Voice API: <https://cloud.google.com/speech/>
7. Amazon Alexa Voice Service
<https://developer.amazon.com/public/solutions/alexa/alexa-voice-service>
8. Wit.ai <https://wit.ai/>
9. Microsoft Bing Speech Recognition API:
<https://www.microsoft.com/cognitive-services/en-us/speech-api>
10. CMU Sphinx: <http://cmusphinx.sourceforge.net/>
11. ANVSIB: <http://speed.pub.ro/anvsib>
12. Ahmed, M.U., Björkman, M. and Lindén, M., 2015. A generic system-level framework for self-serve health monitoring system through internet of things (iot). *Studies in health technology and informatics*, 211, pp.305-307.
12. Tapia, D.I., Rodriguez, S. and Corchado, J.M., 2009. A distributed ambient intelligence based multi-agent system for Alzheimer health care. In *Pervasive Computing* (pp. 181-199). Springer London.
13. Dohr, A., Modre-Opsrian, R., Drobnics, M., Hayn, D. and Schreier, G., 2010, April. The

internet of things for ambient assisted living. In *Information Technology: New Generations (ITNG), 2010 Seventh International Conference on* (pp. 804-809). Ieee.

14. LinkWatch: <https://linkwatchrestservicetest.azurewebsites.net/Help>