An architecture for designing Future Internet (FI) applications in sensitive domains: Expressing the Software to data paradigm by utilizing hybrid cloud technology

Stelios Sotiriadis, Euripides G.M. Petrakis, Stefan Covaci, Paolo Zampognaro, Eleni Georga, Christoph Thuemmler

***Abstract*—The emergency of cloud computing and Generic Enablers (GEs) as the building blocks of Future Internet (FI) applications highlights new requirements in the area of cloud services. Though, due to the current restrictions of various certification standards related with privacy and safety of health related data, the utilization of cloud computing in such area has been in many instances unlawful. Here, we focus on demonstrating a "software to data" provisioning solution to propose a mapping of FI application use case requirements to software specifications (using GEs). The aim is to establish a provider to consumer cloud setting wherein no sensitive data will be exchanged but it will reside at the back-end site. We propose a prototype architecture that covers the cloud management layer and the operational features that manage data and Internet of Things devices. To show a real life scenario, we present the use case of the diabetes care and a FI application that includes various GEs.**

I. Introduction

In recent years we have been witnessing the fostering of cloud computing as a paradigm to offer virtualized resources to everyday users based on a pay on demand service provisioning model. The collection of services includes managing hardware, software, platforms as well as utilization of Internet of Things (IoT) devices for data collection from sensors. In many instances this has been proven to be a novel approach with regards to minimization of operational costs while it increases elasticity; yet not in the healthcare domain. Due to standards, regulations and recommendations such as national legislation, ISO standards such as ISO 80001 and the need to comply with security standards such as ISO 27000 there are severe restrictions to data transfer, storage, aggregation and analysis [4]. In contrast, cloud computing is typically presumed to be based on remote invocations and most fundamentally assumes that data management happens in distant datacenters; This has become a hurdle to the dissemination of cloud solutions in health care. In fact a large scale commercial solution for the health care industry based on public cloud technology provided by Google had to be abandoned in 2012 [10]

To overcome the existing governance issues, the Future Internet Social and Technological Alignment Research (FI-STAR) project is attempting to identify suitable software to data solutions based on Generic Enabler technology, to establish early trials in the health domain and prepare the role out of the technology into FI-PPP phase III. Specifically, based on [1] we aim to create a framework that allows GEs to be delivered to different physical locations. Provider cloud services will manage and upgrade the Generic Enablers on request from the consumer.. Generic Enablers (GEs) are considered as software modules that offer various functionalities along with protocols and interfaces for operation and communication. These include the cloud management for supervision of the underlying infrastructure, the utilization of various IoT devices for data collection and the provision of APIs (e.g. tools for data analytics) and communication interfaces (e.g. gateways, messaging etc.). It should be mentioned that GEs are provided by FI-WARE [8] and are stored in a public catalogue [5], thus developers could easily browse and select appropriate APIs to use.

In this work we focus on the analysis of current GEs provided by FI-WARE [2] in order to demonstrate a fundamental prototype architecture that overcomes the problem of medical data transferring in remote clouds. We established our solution on a twofold conception. Firstly the cloud provider (FI-WARE XI-FI nodes that offer the actual infrastructure and tools) offers required functionalities, and secondly, the private cloud consumer that instantiates GEs to develop FI applications (organized in back-end and front-end sites). To demonstrate such a setting we propose a) a use

Manuscript received July 31, 2013. The authors are members of the Future Internet – Social Technological Alignment Research (FI-STAR project, which is part of the Future Internet Private Public Partnership Programme (FI-PPP).

S. Sotiriadis is a research collaborator of the Technical University of Crete (TUC) and member of the Intelligence Lab, Chania, Crete, Greece (phone: 00306944271599; e-mail: s.sotiriadis@intelligence.tuc.gr).

E. G.M. Petrakis is a Professor of the Technical University of Crete (TUC) and director of the Intelligence Lab, Chania, Crete, Greece (e-mail: petrakis@intelligence.tuc.gr).

S. Covaci is senior solutions architect for future Internet services platforms at the computer sciences and electrical engineering faculty of Technical University of Berlin, Institute for Telecommunication Systems. (email: stefan.covaci@tu-berlin.de)

P. Zampognaro is a senior researcher at R&D Lab of Engineering Ingegneria Informatica S.p.A., Rome, Italy (e-mail:paolo.zampognaro@eng.it)

E. Georga is a researcher of the Department of Materials Science and Engineering, University of Ioannina, Ioannina, Greece (email: egeorga@cs.uoi.gr)

C. Tuemmler is a Physician and Professor of E-Health at the Institute for Informatics and Digital Innovation at Edinburgh Napier University. He is also a collaborator in the MUNICH platform project, based at Klinikum rechts der Isar, Technical University Munich, Germany. (email: c.thuemmler@napier.ac.uk)

978-1-4799-3163-7/13/$31.00 ©2013 IEEE

case analysis based on FI applications of the diabetes care, b) a porting approach based on graphical illustration of the use cases using Unified Modeling Language (UML) standards and c) a prototype architecture that correlates use case functionalities to the supported operations of current GEs. Section 2 presents the motivation of the study. The rest of the paper is organized as follows: Section 3 demonstrates the use case scenario and the presentation of the diabetes care applications and Section 4 focuses on the representation of the scenario to graphical illustrations using UML diagrams. Then in Section 5 we develop our architecture by mapping GEs to requirements. Finally we conclude in Section 6 with the future research directions.

II. MOTIVATION

The software to data cloud model has been described as an emerging approach to be explored as indicated in 2010 EC cloud report [1]. This is considered as part of the inter-cloud paradigm that comes to expand cloud capacity and to allow cloud providers to exchange services [5]. Since health care APIs have been characterized as one of the very resistant areas to be hosted on public clouds the reverse service approach highlights new requirements due to the shift to new, more suitable approaches such as Open Flo-Enabled Hybrid Cloud strategies [11].

In FI-STAR, we are motivated by the openings arising the possibilities of applying hybrid cloud strategies in the health domain. This approach will enable the adaptation of new standards and will offer new opportunities for cloud providers, web entrepreneurs and Small-Medium Enterprises (SMEs) to commercialize their products and services. On the other hand health care providers will be able to improve their effectiveness and efficiency by developing purpose build instantiations, which will be based on reusable modular architectures (Generic Enablers) . The reverse cloud approach will offer the required framework (Public Cloud) to allow Generic Enablers to be initiated at clients' sites and to be hosted in a Private Cloud [4]. Due to the socio-economic relevance of health care in economies globally the real world exploration of such scenarios become meaningful. Therefore FI-WARE has an important role as the provider of GEs and driver of the continuous development of further GEs to extend the GE Catalogue. [6]. By this way, we take advantage of cloud benefits while, at the same time, we minimize its main drawbacks related with security, safety and resilient management.

In detail, the FI-STAR software to data cloud model consists of a Provider Edge that offers certified GEs, the Consumer Edge that instantiates GEs from provider site and the application store for billing and accounting reasons [4]. This will guarantee that no personal data will be transferred to a public cloud. Based on this discussion, we present use case requirements analysis for developing specification for FI diabetes care applications. The analysis underlines the important features that will lead to the proposition of a software engineering methodology based on UML.

III. THE DIABETES CARE USE CASE SCENARIO

The scenario demonstrates an application for supporting patients in daily diabetes care by utilizing a mobile smart device, a sensor(s) and an API that allows connectivity with a private cloud. The back-end should offer a cloud management layer for dynamic, real-time data collection and analysis in order to store data and produce user notifications. The generic concept encompasses a private cloud and a set of GEs that integrate operations such as automatic measurements and monitoring of essential information (e.g. blood glucose, physical activities) from sensor(s) and other. The use case highlights a set of activities.

The user could download and install an application in a mobile smart device that manually or automatically collects data (from a GUI or a sensor). Then, the user performs authentication using dedicated credentials in order to be recognized by the system that represents the business logic implemented in the private cloud (consumer edge). In particular, the private cloud offers the infrastructure to host FI applications (GEs) within the premises of the consumer site (e.g. a hospital). The consumer cloud that communicates with a public cloud for getting software modules executes the resource management. The data storage and analysis occurs in the back-end that gives control over sensitive data and ensures security.

The user could perform data recording for a variety of cases such as a) monitoring glucose data in continually (e.g. every 1 or 5 minutes) fashion from a sensor(s), b) recording blood glucose at random times (based on manually readings), c) recording medication (e.g. type 1 regarding injections or type 2 that refers to pills), d) recording meals and e) recording physical activities. Each case could have multiple characteristics that represent the actual dataset (e.g. type 1 diabetes could have type of insulin –characters, units-numbers, injection site-characters etc.).

Based on these, the private cloud could analyze data and produce alerts in real time by evaluating data. Further the patient could be able to review information and define achievement goals. From the perspective of the personnel, there should be an authentication mechanism to allow access to a platform that includes the dedicated patients, their data and available APIs for analysis (legacy tools). Personnel could be able to perform data analytics (e.g. using specific GEs or APIs) for assessment and recommendation.

To define such a setting, we describe software quality characteristics based on the ISO 25010 [9]. Briefly, these include functional suitability for complete, correct and appropriate functionalities as well as performance efficiency that is very crucial. In detail, such characteristics should include the time management (e.g. sampling period of the used sensors), alerts (via notifications) and resource utilization (e.g. data storage, required bandwidth, CPU and RAM). Another important factor is compatibility with different sensors and software, and interoperability. Next section introduces the use case analysis and the mapping of GEs to use case requirements based on UML standards.

IV. Mapping Use Cases to GEs Using UML

This section details the representation of the use cases using UML standards. Briefly, UML allows the identification and graphical representation of functional requirements. We start by modeling a UML use case diagram of actors and various cases (e.g. install application, data recording etc.). This could give us a general view of the whole system. The back-end cloud is considered as the environment to host applications and tools for patients and personnel. The sensor is a specialized part of the patient that generates data stored in a private cloud.

A. Definition of diabetes care system using UML

This section demonstrates the transformation of the diabetes care use case scenarios to UML diagrams. We start by showing the possible actors (patients, sensors, cloud and personnel) and the relationships with use cases (e.g. a patient installs applications, performs authentication and generates data that are stored in the back-end cloud and are accessible by the clinicians – named as personnel). Fig. 1 shows the UML use case.

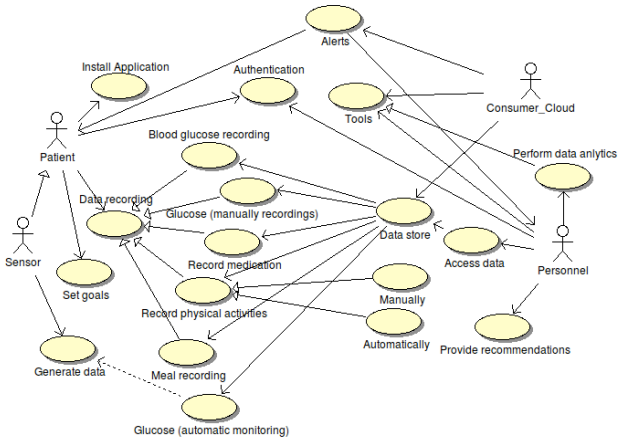

Fig. 1. The UML use case diagram of the diabetes care application.

Based on Fig. 1, we have produced three UML activity diagrams to show flow of communication that will assist in the design of a high level architecture. The aim is to demonstrate every aspect in terms of interactions and then analyze their features as software modules and identify offered GEs.

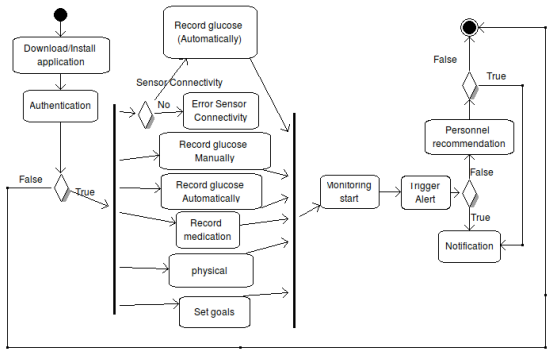

Fig. 2. The UML activity diagram of the patient activities/actions

Fig. 2 demonstrates the patient activity diagram and the workflow of actions. As described in Section 3, the user downloads the application, performs authentication and then accesses the offered services (e.g. record data manually or using a sensor). The monitoring started when essential data have been recorded (e.g. blood glucose levels). In case of a situation, an alert is triggered and patients and/or personnel are informed by getting a notification. The actual situation is triggered from an event or a set of events. Personnel also provide advices and produce notifications.

Fig. 3 demonstrates the activity diagram of the personnel. For example each clinician is an authorized user that accesses the data of the private cloud. The hypothesis is that data needs to be stored in the back-end site for safety and privacy. First, they perform authentication (system assigned credentials). By retrieving and evaluating data they can examine notifications, produce recommendations and perform data analytics using private cloud dedicated APIs. Also, they access data mining and data integration tools (e.g. GEs or legacy system APIs).

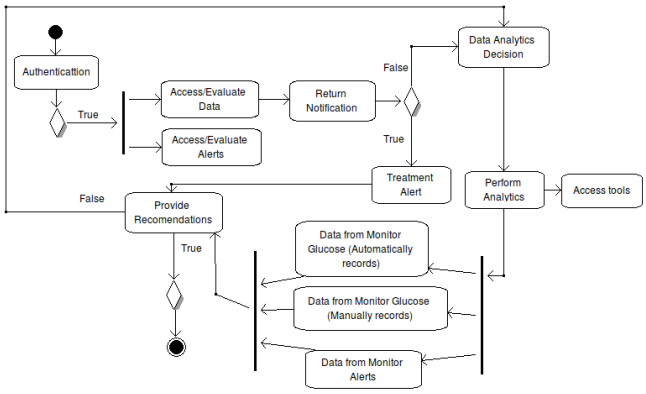

Fig. 3. The UML activity diagram of the personnel activities and actions

Next we focus on the analysis of the high level architecture by porting UML diagrams to software elements.

B. Porting UML diagrams to high level architecture software elements of the diabetes care system

We define a high level architecture that encompasses a set of software configurations required to reason about the system. We present our solution in a modular structure of various elements. These are the various software modules, their associations and the properties and operations of both properties and interactions. The aim is to present a strategy to port use case requirements to the diabetes healthcare system shown in Fig 4. The details are presented below.

- The "record data automatically" module includes a) an interface to detect the sensor and ensure communication, b) an interface to input the data stream from the sensor (these could be in from Bluetooth, Wi-Fi, and/or 3G networks), c) an interface to configure device with essential parameters (e.g. diabetes type and a measurement) and d) an interface to subscribe data based on required parameters (to identify contextual information).

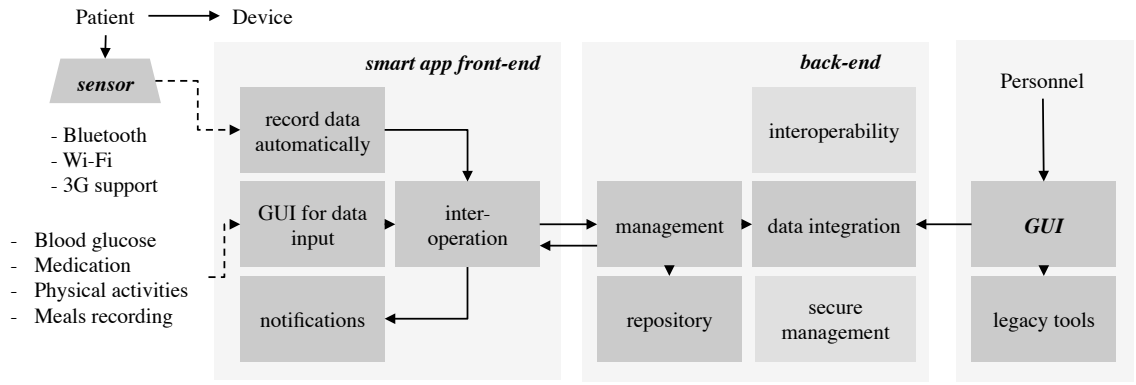

Fig. 4. The high level architecture of the diabetes care use case scenario

- The "GUI for data input" module comprises interfaces for manual data input. This will have a) an interface for data collection from the GUI (e.g. recording of meals or physical activities), b) an interface for sending configuration request for specific parameters to the system (e.g. to store context) and c) a subscription interface to receive data for selected parameters.
- The "interoperation" module includes interfaces for query management. These are a) interfaces for retrieving context data in a request/response or in a subscription mode from either applications of patients (e.g. from the sensor control API), and b) an interface for querying data and/or registering data to the system.
- The "notification" module includes an interface for collecting data according to the output modes of the interoperation module. This will produce static (e.g. general recommendations for meals and activities) or dynamic warning (e.g. notifications on real-time based on events or situations generated from a set of events) available to users through a GUI.
- The "management" is the entry point of the system for accessing information from the interoperation module. This will include a) an interface for querying properties of things (patient sensor), b) an interface for forwarding updates with regards to properties (to other modules), c) an interface to input data about IoT resources (patient sensor as a resource), d) an interface to input sensor data and their properties, e) an interface to input relationships of how sensor and modules are linked and how properties (e.g. blood glucose levels) are delivered.
- The "repository" is a module for registering the context information of provider applications. It includes a) an interface for input a query for the location of where context is available and b) an interface for discovery of contextual information.
- The "data integration" is the module that offers the properties of the various actors of the system. This includes a) an interface for patients properties input, b) an interface for information input for the repository (contextualized properties such as the blood glucose level for a specific patient) and c) an interface for device properties (to monitor properties from sensors).
- The "interoperability" defines a module that offers mediation in communication. In particular, we made the assumption that there is an interoperability mechanism (offered by an API) to allow association of different communication protocols and different data models. This will include a) an interface to provide a virtual proxy for input data from administration GUIs (e.g. the user interface for input data manually) either from patient or personnel and b) an interface to provide ontological description of services in order to assist on the translation of services. It enables transmission of the meaning of services to assist on the discovery and data management between different modules.
- The "secure management" refers to an interface that allows secure communication among the back-end, the smart application and the personnel GUI. The interface is related with point-to-point authentication strategy between the patient application and the private cloud in addition to the back-end private cloud firewall. It will have an input for login authorization from the patient and will output an access token.
- The "GUI" module of personnel includes interfaces for secure access and control of sensitive data. This will have a) an interface to request and respond data from the personnel GUI and b) an interface for accessing the file storage (e.g. reports or notifications).
- The "legacy tools" module represents APIs for data analysis that are offered from the cloud back-end or are installed in the hospital private cloud. It describes the current software installed in the use case site (e.g. a blood glucose special software). This extends the interoperability module to allow communication with other modules. The core functionality provides an interface to convert messages to different formats. Another aspect is the meta-data representation of the meaning of services (using semantics) to allow

efficient porting of APIs to private clouds.

To conclude, the high level architecture describes the software modules and their interactions. Next we demonstrate the mapping of GEs in such scenarios and we prototype a system for the FI-STAR use case applications.

C. *Mapping high level architecture modules to GEs*

In this section we correlate the architecture modules and their functionalities to the offered GEs available in the FI-WARE catalogue [3]. Next, we present a brief presentation of GEs [2] and associations with diabetes care modules.

- The "Protocol Adapter GE" serves in between of the registered devices (e.g. the patient sensor) and the gateway that offers the communication layer to the host environment. It supports data collection, sensor detection and connectivity. This GE is linked with the Gateway Device Management GE or the Data Handling GE.
- The "Gateway Device Management GE" allows communication between backend and devices (e.g. the protocol adapter). It is linked with the Data Handling GE, Security GE, Protocol Adapter GE, Devices with machine-to-machine (M2M) protocols, and the Backend Device Management GE.
- The "Data Handling GE" is an attribute based access control system for safe data storage. It offers a repository and a policy based on Privacy Policy Language (PPL). This GE is linked with the Gateway Device Management GE.
- The "Things Management GE" is a backend component that acts as the central point of contact to receive information about Things and their Properties [2]. This GE is linked to gateway GEs (e.g. the Data Handling GE). It involves two GEs as follows.
  a. The "Internet of Things Broker GE" based on the OSGI framework [6] to provide an interface (Next Generation Service Interface (NGSI)-9/10 [2]) to communicate with other GEs. This GE communicates with the Context Broker GE and the Configuration Management GE (using the NGSI9/10 protocols). In brief, the interface OMA NGSI-10 allows exchanging information about entities and their attribute while the interface OMA NGSI-9 offers the availability of information about entities and their attributes. So, instead of swapping attribute values, the exchanged information refers to which provider could offer such attribute. It is linked with the Configuration Management GE.
  b. The "Configuration Management GE" is responsible for context availability registration and discovery. It uses IoT agents that make use of NSGI-9 interface. It is linked with the Publish/Subscribe GE, IoT Broker GE and the Gateway Device Management GE.
- The "Publish/Subscribe Context Broker GE" offers publication of context information by entities to the context consumers (e.g. NGSI-9 clients). This module allows push and pull communication with the configuration management GE using the NGSI-9 interface to exchange context aware information.
- The "Mediator GE" is a RESTFul API [7] to handle mediation services. It is build on a client/server architecture where clients make requests and servers produce appropriate responses. This GE offers interoperability among protocols and data models.
- The "Identity Management GE" offers authentication mechanism as part of the generic security GE. The GE could be linked to the Gateway Device Management GE to allow an authentication framework.
- The "Cloud Edge (Proxy) GE" provides agents located in the back-end site and ensures the link between cloud provider and the end-user. It offers various roles such as the management of a catalogue of applications that are compatible with a set of Cloud Proxies (named as Service Aggregator). This GE accesses it by using a Service Platform Management Interface (SPMI) [2] to manage cloud platform, instances, images and users.
- The "Cloud Datacenter Resource Management (DCRM) GE" offers cloud hosting capabilities as well as management of the infrastructure resources using OpenStack cloud software. This GE is linked to the Edgelet Management GE that offers a distributed environment to run edgelet container software [2].
- The "Edgelet Management GE" offers hosting of lightweight application components (based on JavaScript) for achieving high data rates and low latency in distributed settings (e.g. cloud proxy sites). This GE is linked to the Cloud Edge (Proxy) GE.
- The Advanced Communication Middleware GE enables flexible and secure communication between distributed applications and to/between FI-WARE GEs.

V. THE GE PROTOTYPE ARCHITECTURE FOR HOSTING FI-STAR APPLICATIONS

The FI-STAR conceptual model consist of the FI-WARE provider site (platform) that offers various GEs, the private cloud consumer site(s) that is the back-end of the system and integrates the use case trial software modules (and their associated GEs) and the interoperation setting for communication with the front-end that represents the human user (GUI and sensors). In general, a cloud provider uses the DCRM GE and the Edgelet Management GE to link provider and back-end sites. This refers to the initial configuration and deployment as next the back-end is detached. Fig. 5 demonstrates the design of the proposed architecture.

Particularly, data entered (GUI or from sensor) are submitted to the system (through data handling or protocol adapter) and are managed by the gateway. Then, communication is forwarded to the backend sites (things management). The publish/subscribe context broker allows push/pull or message subscribing for alerts triggering.

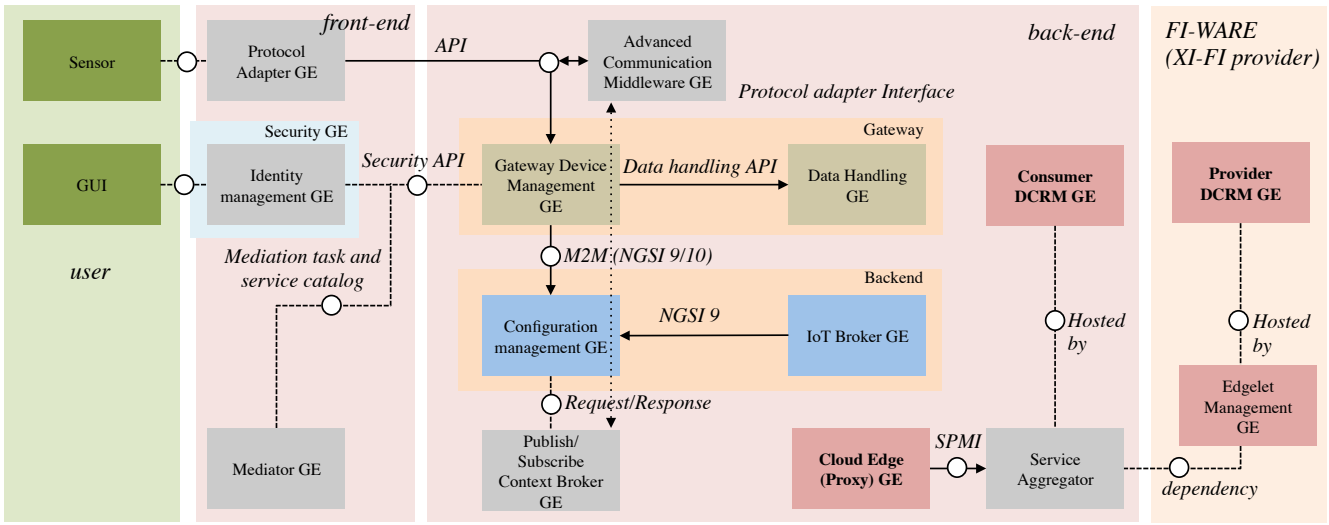

Fig. 5. The GEs prototype architecture and communication model among GEs for FI application of the diabetes use care scenario.

The general hypothesis is that the cloud proxy has been configured and compatible services (registered in the service aggregator) are available in the back-end site. Hence, we can now map software modules of section B to GEs specifications of section C as next. Protocol Adapter GE to "record data automatically" module, Data Handling GE to "GUI for data input" and "personnel GUI" module, Publish/Subscribe Context Broker GE to "interoperation and notifications" module, the Gateway and Backend GEs to the "management and repository" module, Configuration Management and IoT Broker (and interfaces) to the "data integration" module, Secure GE (Identity Management GE) to "secure management", Mediator GE to "interoperability" module, Service Aggregator (Cloud Edge GE) to legacy tools module. Next, we extend our and we focus on a generic architecture of the FI-STAR applications. The "Advanced Communication Middleware GE" is for event management.

VI. Conclusion

This work presented a prototype architecture for a diabetes healthcare application for patient monitoring. Through the analysis of initial requirements (in the form of text) to the classification of use cases and characterization of actors and their operations (in the form of UML) we have concluded to prototype architecture for hosting FI applications using the FI-STAR reverse cloud approach. We suggest that relevant use cases that require a health driven cloud reverse approach could utilize this as a basis for developing an architecture to host FI applications in the premises of a private cloud. However, for specific use cases the approach is to follow different software development processes, still using UML to characterize their architecture.

Acknowledgment

The authors are members of the Future Internet – Social Technological Alignment Research (FI-STAR) project, which is part of the Future Internet Private Public Partnership (FI-PPP) run by the European Commission. FI-STAR is a FI-PPP phase 2 project which commenced on 1st April 2013 and will conduct at least seven early clinical and non-clinical digital-health use-case trials in seven or more European countries.

References

[1] L. Schubert, K. Jeffery, B. Neidecker-Lutz (2010) "The Future of Cloud Computing –Opportunities for European cloud computing beyond 2010", European Commission [Online]. Available: http://cordis.europa.eu/fp7/ict/ssai/docs/cloud-report-final.pdf, Accessed: 8 June 2013

[2] FI-WARE wiki Available: https://forge.fi-ware.eu/plugins/mediawiki/wiki/fiware/index.php/Welcome_to_the_FI-WARE_Wiki, Accessed: 8 June 2013

[3] FI-WARE catalogue, Available: http://catalogue.fi-ware.eu, Accessed: 8 June 2013

[4] C. Thuemmler, J. Mueller, S. Covaci, T. Magedanz, S. D. Panfilis, T. Jell and A. Gavras, "Applying the Software-to-Data Paradigm in Next Generation E-Health Hybrid Clouds", *In Proc. Proceedings of the 10th International Conference on Information Technology (ITNG2013)*, IEEE Computer Society, ISBN 978-0-7695-4967-5

[5] S. Sotiriadis, N. Bessis, F. Xhafa, N. Antonopoulos, "From Meta-computing to Interoperable Infrastructures: A Review of Meta-schedulers for HPC, Grid and Cloud," in Proc. Advanced Information Networking and Applications (AINA), 2012 IEEE 26th International Conference on Advanced Information Networking and Applications (AINA-2013), 26-29 March 2012, pp. 874-883

[6] OSGI framework, Available: http://www.osgi.org/Main/HomePage, Accessed: 8 June 2013

[7] B. Mulloy "Web API design", Available: http://info.apigee.com/Portals/62317/docs/web%20api.pdf Accessed: 8 June 2013

[8] FI-WARE, Available: http://www.fi-ware.eu/, Accessed: 8 June 2013

[9] ISO-25010, Available: http://sa.inceptum.eu/sites/sa.inceptum.eu/files/Content/ISO_25010.pdf, Accessed: 29 July 2013

[10] Google Health, Available: http://www.google.com/intl/en_us/health/about/, Accessed: 29 July 2013

[11] Open Network Foundation (2012), "OpenFlow-Enabled Hybrid Cloud Services Connect Enterprise and Service Provider Data Centers", ONF Solution Brief 2012, Available: https://www.opennetworking.org/solution-brief-openflow-enabled-hybrid-cloud-services-connect-enterprise-and-service-provider-data-centers, Accessed: 29 July 2013