IoT for a Shock Warning System

John P. Brogan, *Edinburgh Napier University*, Dr. Christoph Thuemmler, *Edinburgh Napier University*, Dr. Samuel A. Fricker, *Blekinge Institute of Technology, Sweden*, Dr. Oli Mival, *Edinburgh Napier University*

***Abstract*— this technical paper details the interdependencies between Internet of Things (IoT) and latest Future Internet (FI) technology, demonstrated by using the example of a smart connected health (SCH) use case, for the monitoring and prevention of shock in hospitalized patients. Septic shock is a life threatening condition, which requires immediate attention by the doctor in charge. It is associated with a relatively high mortality of 30-50% whereby early detection of the condition is crucial for the outcome. FI-WARE Generic Enablers (GEs) are used for rapidly developing a shock warning system. The system is designed around sensors, providing data inputs, of patient heart rate and blood pressure. Input metrics are required for subsequently computing an appropriate output, defining a shock index measurement and triggered is an alert, such as a text message, to a doctor's mobile phone or a pager to raise awareness of the fact that a patient is deteriorating and about to slip into shock. The focus of this paper approaches a discussion surrounding research of GEs and their operation in wellness and ambient assisted living domains. Within health care contexts, a Medical Modular Architecture (MMA) approach might be needed to complement the GE concept because of associated legal and ethical requirements. Our work presents an IoT methodology for SCH and is useful for detecting patients at risk of slipping into shock earlier, thus increasing their chances of survival. In this context we will examine an Orion Context Broker (OCB) GE and its interoperability as an IoT interface with backend server components. Shown by our approach is a method for creating a shock warning system based on e-Health informatics. It is presented as a framework defined from a set of future internet (FI) middleware paradigms consisting of the 3-tier architecture, software to data, privacy and security.**

***Index Terms*—Internet of Things, Generic Enabler, Medical Modular Architecture, Future Internet, e-Health, Smart Connected Health, Context Broker, FI-WARE, FI-STAR, Shock index, Software reusability**

I. Introduction

FUTURE internet (FI) research is guided by the aim of conducting research relating technologies for developing the next generation of the internet. Future Internet Architectures (FIA) are being used in applications addressing the needs of systems necessary for future and legacy health care solutions [1]. An important goal of European partner research is for e-Health and m-Health to achieve social and technological alignment across European health care sectors. The work in this field is the on-going responsibility of the FI-STAR project community [2]. The Internet of things (IoT) is an integral part of the FI and a crucial aspect of developing smart connected health (SCH) applications [3]. This special issue paper approaches a discussion surrounding FI research and the application of IoT within wellness and ambient assisted living domains [4]. FI-WARE generic enablers (GEs) [5] are introduced and explained for the purpose of describing a SCH demonstration system.

II. Generic Enablers

The Future Internet (FI) is a topic of research funded by the EU, its function is to research the transition from the current internet approach to an extended and functionally enhanced internet in the Future and technologies for building future internet applications and services. Future Internet Architectures (FIA) provide a technology foundation for health care applications. The goal is to attain distributed health care provision and support self-management and greater patient and carer autonomy. (e-Health, m-Health). FI-STAR, Future Internet - Social and Technological Alignment Research aims to achieve STAR across European health care sectors. The FI-WARE public cloud platform provides a state of the art components engineering paradigm. Generic Enablers (GEs) are provided by FI-WARE and they are built to FI-WARE specifications for operation upon the core platform or as individual instances. GEs are reusable software building blocks designed for FI domains e.g. e-Health, e-Cities and Smart Grids [6].

FI-WARE GEs are building blocks of a platform and are described by FI-WARE partner supplied open specifications [7]. Specifications provide details to build compliant GE software. New systems are built from standard GE building blocks. GEs provide certain and unique capabilities that can be reused by a network plugin interface for them. Applications based on a GE framework require the FI-WARE core platform

This paper was submitted for review on 30th May 2014. The work is supported by the European Commission. Grant agreement No 318389.

John P. Brogan is a Research Fellow with the Institute for Informatics and Digital Innovation at Edinburgh Napier University, Edinburgh, UK (e-mail: j.brogan2@napier.ac.uk).

Dr. Christoph Thuemmler is Professor with the Institute for Informatics and Digital Innovation at Edinburgh Napier University, Edinburgh, UK (e-mail: c.thuemmler@napier.ac.uk).

Dr. Samuel. A. Fricker is assistant Professor with the Blekinge Institute of Technology, Department of Software Engineering, Software Engineering Research Laboratory, Sweden (e-mail: , Samuel.fricker@bth.se).

Dr. Oli Mival is Senior Research Fellow with the Institute for Informatics and Digital Innovation at Edinburgh Napier University, Edinburgh, UK (e-mail: o.Mival@napier.ac.uk).

i.e. a PaaS model facilitating IaaS and SaaS so as to provide XaaS capabilities which are relevant to FI application domains. GE applications achieve functionality defined to standards. They also provide APIs to enable interoperability.

A. Architectural Design Components of SCH Applications

GEs are applied within UML use case designs obtained from the 4+1 views of software specification which are the common practices for object-oriented systems design and development. However UML "use cases" are different to EU FP7/H2020 "use cases". A UML use case is a set of user interaction scenarios for a specific user goal. A FP7/H2020 use case is a technology evaluation scenario. With respect to GEs then mappings are created from the application UML design to FI-WARE chapters and relevant GEs. Building block macroscopes are applied to the design. The mapped components are used to replace areas of a system that instead can be implemented by standardized and reusable GEs.

B. Operation

Generic Enablers operate upon many different form factor devices e.g. PCs, servers, tablets, smart phones and medical hardware. GEs work on future and legacy processing or storage technologies e.g. CPUs, databases, data structures, DSPs and cache. GEs are implemented for different operating systems and are constructed from varying programming languages. GE implementations require to function using different drivers and protocols. Therefore they are multimodal instruments.

FI-WARE GEs are not portable because the GE software implementation is required to be recompiled specifically for various implementing processor architectures, however FI-WARE GEs use REST HTTP/S to enable communication between machine types. GEs interoperate over networks and interoperation is performed between GEs with external system entities, including also specific enablers (SEs). Each GE requires API servers and the server backend offers web services. However this is inefficient because several web server containers are required to host web services on the network. Ideally a single server and suite of web services, made available from the network, is preferred. Also, generally security and monitoring operations can be better tracked and implemented from a dedicated server implementation.

GEs operate in essence as machine to machine (M2M), component to component (C2C) or peer to peer (P2P) interoperating systems [8]. Therefore the particular issue arises, for health care domains, regarding security. The communication model of GEs is provided over public or private networks and so security, especially in public network contexts, is critical to ensure. Obviously this feature has a strong commercial objective and security should be implemented in a way to not compromise other quality attributes of a system operating with GEs, such as performance and dependability. Some security essentials are made available from FI-WARE, for example an identity management GE. However FI-WARE makes no provision for ensuring that all the GEs can operate within secured contexts.

C. How to Build a Generic Enabler

The method entails that each GE is required to be decomposed to operational component parts i.e. GE domain subcomponents, functions for operational part control, uniform I/O message passing over HTTP/S, XML DOM or JSON interpreters and dispatchers, APIs to identify GE supplied functions (as reusable network resources), structured data formats of XML, I²C or JSON and an API proxy backend server component. The result of their combinations, are GE implementations providing a suite of services to their uniquely supplied functionalities, on demand as network resources.

D. How to Use a Generic Enabler

GEs are downloadable applications from a repository of GEs provided from a FI-WARE Catalogue i.e. a GE app store. They are used as operational software components and are able to be wired into application design architectures. Enabling the GEs to interoperate with their chosen application domain and its software components requires a REST API and necessitates RESTful web services within the application design. Hence to use a GE then a web client/server is required within the design model. For the purpose that GEs and the remainder of the system can communicate to enable the effective operation of the software.

III. Medical Modular Architectures

If legal norms, ethics and technology standards have to be considered when planning, designing and implementing use cases based on IoT GEs applied within cyber-physical systems then Medical Modular Architectures [9] (MMAs) must be studied for implementations containing GEs. MMAs, instead describe a strategy of providing Specific Enablers (SEs) based on the specifications of GEs, relevant technical standards, legal and ethical requirements for subcomponent operations.

Future e-Health applications may be implemented and based upon MMAs. They describes a new type of GE approach of modular architectures in relation to legal frameworks and ethics of deployed application scenarios, within e-Health operation contexts. MMAs are deployed for ensuring patient data integrity according to legal requirements of an application domain and technical standards such as ISO 80001 and ISO 27000. The idea utilizes instead the Medical Modular Architectures. Whereby MMAs define how GEs are able to be integrated into a solution and importantly an MMA is observed to be useful for coping with Hippocratic or other sensitive data and its use in medical processing applications. These type of applications are built often from IoT sensors and are required also for the implementation of *cyber-physical* patient monitoring setups [10].

A. Examples of MMAs in SCH scenarios

MMAs are essentially deployed within secure and private locales for SCH use cases and implementations. The idea secures contexts of GE operation so that the GEs, deployed within systems, operate legally and ethically. MMAs support GE functionality by ensuring that communicated data is kept private and safe whenever data is exchanged between GEs and other SCH system components. For instance, based on a small selection of the overall number of use cases provided from the FI-STAR project, some situations where MMA’s might be used are described and shown are that the chosen use cases actually require medically approved GE instances.

For example, the virtualization of operating theatre environments and real time data integration for monitoring and reduction of errors is such a scenario wherein a GE requires to be appropriately implemented within an MMA architecture. Also, 2-D bar-coding for a real time reverse drug supply chain provides a further scenario whereby the integrity of GEs require to be uncompromised for prohibiting the possibility of illegal counterfeit drugs being inserted, at any point, into the supply chain. MMAs are incorporated into FIA developments for SCH applications, simply to ensure that legal and ethical norms and also standards are implemented for superseding the operation of less secure FI-WARE GEs and their deployments in health care scenarios. For more information please refer to the FI-STAR online literature.

IV. Shock Warning System

The primary goal of this paper is to describe our findings surrounding a proposal for e-Health software, implementing a patient monitoring application. We are reporting herein our research conducted on the development of an IoT shock warning system. The following section parts describe an ideal patient monitor system use case and its arising conceptual scenarios.

A. *Ideal System Specification*

In the ideal scenario, the system is to be provided as a full featured application and is intended to be created as software from a composition of components for handling continuous data signals, sent from sensors. Patient sensors are utilized for monitoring heart rate and blood pressure using a pulsoxymeter or ECG machine and a blood pressure measurement sensor device. The system’s component architecture consists initially of specific enabler and generic enabler building blocks.

A shock index [11], see Fig 1, is calculated during the system’s operation, for the purpose of automatically monitoring the condition of patients in a hospital ward. The shock index (SI) calculation is a quotient (result) obtained from dividing patient heart rate (HR) by patient blood pressure (BP). The shock index concept is fully accepted in the medical community and is well described in the literature [12]. When a SI value is determined relevant to a patient’s condition and is calculated so that it reaches a certain predefined threshold then an alarm is raised to a responsible caregiver.

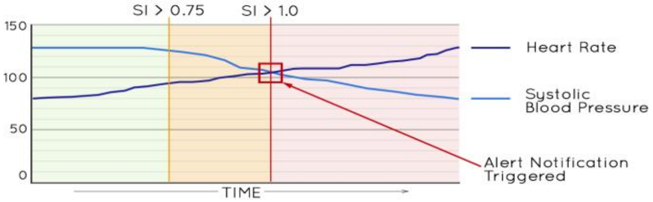


Fig. 1.a SI threshold detection from HR and BP advancing SI to critical state.

Raised alerts are sent to a relevant Doctor’s mobile device. Unique patient parameters, defining a critical shock index, are able to be supplied, by Doctors, to update the system’s state, in real time, for the purpose of ensuring that the system can realise dynamic thresholds for triggering alarms associated with patients where their wellbeing conditions are detected by the system to be in or out of a critical state of shock. A third party member of medical staff is required for assigning patients to sensors and available Doctors. A member of the medical staff can add and remove patients to and from the system. The system can be turned on and off automatically by using start up and shutdown procedures and these can be initiated by medical staff adding at least one or more patients or removing all patients from the system. The system is capable of detecting these changes after it is switched on. The Doctor conducts the system’s operational management procedures from a control unit (mobile device application). The Doctor’s control unit is utilized for notifying the doctor when there is a new patient to supply care to, supplying a new shock index parameter and to indicate to third party medical staff when a patient’s care has ceased. The Doctor updates the system with new shock index parameters whenever required and this capability is also necessary for a Doctor being able to remotely and arbitrarily set shock index thresh holds regarding a list of their patients receiving treatment.

Some caregiving management control, pertaining to SI thresh hold, is also provisioned from a SE residing on a Doctor’s mobile device handset. Medical staff work with a terminal SE from which they can add and remove patients to or from IoT connected sensor banks and they also manage associations with the patient’s primary caregivers. Initially a baseline SI value is automatically provided for each patient assigned to sensors when patients are originally added as IoT entries to the system. Only pre-approved medical staff and Doctors may log in to the system, prior to use. Password based authentication is setup by the system administrator to enable Doctors and medical staff to log into the system. The interaction of Doctors and medical staff with the system is provided from graphical user interfaces using touch, gesture controls, graphical display (web page and mobile) and sound events to indicate system responsive feedback for all operations performed from hand held control units and also from medical staff terminals.

Patient data is stored internally within the system and is regarded as Hippocratic. Main inputs of the system, during operation, are *real time* sensor data streams and primary outputs are alarm alerts directed to Doctors or medical staff regarding a patient in a critical condition. Patient’s ID, name and physical condition, after patients have been assigned to a Doctor, are shown by a display on the Doctor’s control unit and also upon the medical staff terminal. Medical staff terminals do not retain any private patient information after a patient’s details have been entered into a terminal and subsequently deployed to the system.

V. Orion Context Broker GE

The Orion Context Broker (OCB) is an implementation of a

Publish/Subscribe Context Broker GE [13], providing the OMA NGSI-9 and NGSI-10 interfaces [14]. Using these interfaces, clients can do several operations, register context producer applications, e.g. a pulse sensor attached to a patient, update context information, e.g. send updates of HR, being notified when changes on context information take place e.g. the HR has changed, or with a given frequency e.g. get the HR each minute and query context information. The Orion Context Broker stores context information initiated as updates from connected applications i.e. queries are resolved based on that information.

NGSI-9 and NGSI-10 are networked REST API's designed for the interoperation of the OCB GE with backend server SEs and hence enables use of REST HTTP/S for communicating with IoT sensors, backend components and an alarm device. Operation is made possible due to the subscription feature that the OCB GE supplies. It is triggered using the NGSI REST API interfaces. A custom utility program (GE_REST_Utility) is required, at the very beginning, to set up and initialize an OCB GE from NGSI-9 and NGSI-10 API calls. The API enables their respective internal data models to be constructed and the GE to be made operational for subscription notification transmissions. Also required, to be purpose built, is a web service receiving HR and BP values sampled from IoT sensors, for the purpose of interoperating with the GE. All data is encoded to the JSON or XML formats during the runtime of the shock warning system. The web server hosts the alarm service which is activated from interactions between the GE and relevant web services connecting the system together.

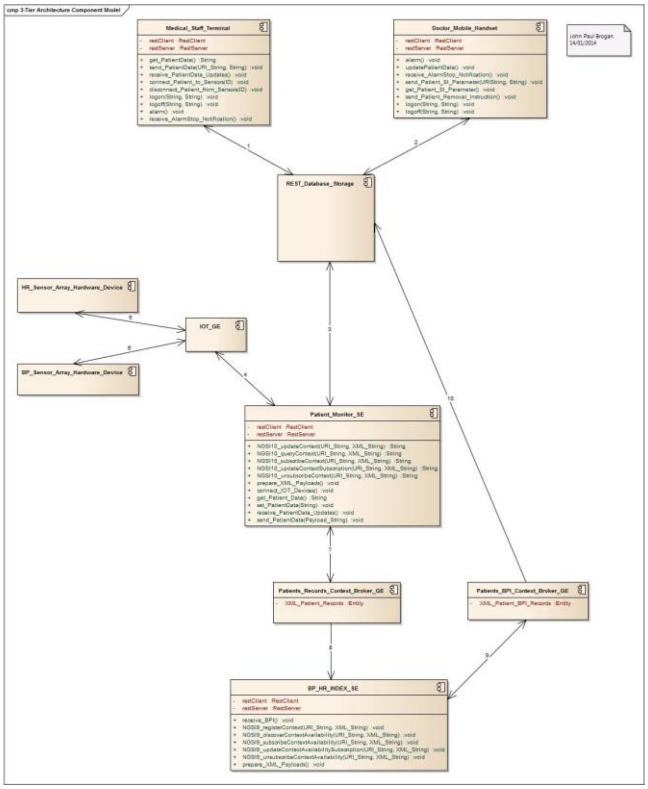

cmp 3-Tier Architecture Component Model
Medical_Staff_Terminal
- restClient :RestClient
- restServer :RestServer
+ get_PatientData() :String
+ send_PatientData(URI_String, String) :void
+ receive_PatientData_Updates() :void
+ connect_Patient_to_Sensors(ID) :void
+ disconnect_Patient_from_Sensors(ID) :void
+ logon(String, String) :void
+ logoff(String, String) :void
+ alarm() :void
+ receive_AlarmStop_Notification() :void
Doctor_Mobile_Handset
- restClient :RestClient
- restServer :RestServer
+ alarm() :void
+ updatePatientData() :void
+ receive_AlarmStop_Notification() :void
+ send_Patient_SI_Parameter(URIString, String) :void
+ get_Patient_SI_Parameter() :void
+ send_Patient_Removal_Instruction() :void
+ logon(String, String) :void
+ logoff(String, String) :void
John Paul Brogan
14/01/2014
1
2
REST_Database_Storage
HR_Sensor_Array_Hardware_Device
IOT_GE
BP_Sensor_Array_Hardware_Device
5
6
4
3
10
Patient_Monitor_SE
- restClient :RestClient
- restServer :RestServer
+ NGSI10_updateContext(URI_String, XML_String) :String
+ NGSI10_queryContext(URI_String, XML_String) :String
+ NGSI10_subscribeContext(URI_String, XML_String) :String
+ NGSI10_updateContextSubscription(URI_String, XML_String) :String
+ NGSI10_unsubscribeContext(URI_String, XML_String) :String
+ prepare_XML_Payloads() :void
+ connect_IOT_Devices() :void
+ get_Patient_Data() :String
+ set_PatientData(String) :void
+ receive_PatientData_Updates() :void
+ send_PatientData(Payload_String) :void
7
Patients_Records_Context_Broker_GE
- XML_Patient_Records :Entity
Patients_BPI_Context_Broker_GE
- XML_Patient_BPI_Records :Entity
8
9
BP_HR_INDEX_SE
- restClient :RestClient
- restServer :RestServer
+ receive_BPI() :void
+ NGSI9_registerContext(URI_String, XML_String) :void
+ NGSI9_discoverContextAvailability(URI_String, XML_String) :void
+ NGSI9_subscribeContextAvailability(URI_String, XML_String) :void
+ NGSI9_updateContextAvailabilitySubscription(URI_String, XML_String) :void
+ NGSI9_unsubscribeContextAvailability(URI_String, XML_String) :void
+ prepare_XML_Payloads() :void

Fig. 2. Shock Warning System 3-Tier Architecture UML Design. The components of the system depict the IoT OCB GE interoperating with the backend web server components of the system with REST HTTP/S.

Our design, for a shock warning system, is based around the N-Tier or *Multi-tier* architecture design framework [15] and it is presented as a UML design in Fig 2. UI presentation and interaction are located at the top tier, a data access logic SE facilitating a REST authenticating database server exposes the UI but only to a database server SE. This SE is located on the next tier down, on the storage data tier and is responsible for mediating all interaction between tiers. A data access and logic SE facilitates also a REST database interface to and from this system's backend processing tier. The back end SEs and GEs comprise the shock warning system's processing tier and it is composed of an IoT OCB and two other OCBs, necessary for ensuring that data flows exhibit security and integrity.

The system is implemented as a 3-Tier architecture to ensure that a Hippocratic medical modular architecture is implemented for handling data operations within sensitive e-Health deployment scenarios. As a consequence, the OCB GEs operate instead as semi-secured and legitimate MMA components, utilized to fulfil the purpose of brokering context and state information of patient sensor data to the processing tier and finally to any determined SI decisions regarding potential outcome scenarios (alerting) to the presentation tier.

VI. IoT middleware Mapping

The IoT GE for the shock warning system is essential for enabling a context broker representation of patient sensor setups. Sensors must send data over REST HTTP/S to this IoT GE's internal representation. The IoT GE (OCB) is a specially customized FI-WARE configuration manager GE [16], necessary for configuring IoT devices and it is provided as middleware supplementing the operational implementation of a backend web server environment. It therefore performs the task of brokering sensor data to web services. However the sensing devices are also a requirement of the system, for delivering input data streams as a continuous time series to web services that consume this data and subsequently act on it.

The IoT network can be configured from the IoT GE (OCB), by initializing it with the representation of different types of sensors, necessary for recording a patient's heart rate and blood pressure. This is represented using the OCB's internal data model, relating to IoT for representing the entities connecting to it e.g. to describe any type of sensor device and their capabilities and also patient details. The remaining factor, to consider, is the requirement for implementing also web services ensuring that received sensor data is processed adequately and according to MMA specific requirements. Patient data processing is necessary for producing content e.g. an alert or a warning that an alert is about to occur for a patient. The crucial area of concern is performing IoT connections to the system's backend. Fortunately FI-WARE OCB is an IoT configuration management broker and can perform the role of interfacing IoT sensors with web service components. The difficult area to asses surrounds the sensors and their own particular middleware representations but there exists present research conducted in this field that explores current progress and trends relating to specific sensor implementations e.g. Arduino boards and Raspberry PI [17]. Often the investigated research mention also 6LOWPAN protocols for providing low powered and hence networked sensor implementations for the wireless embedded internet [18], [19]. Assuming that sensors are sophisticated enough for performing REST HTTP/S communications so as to invoke an IoT configuration management GE (OCB) via NGSI-9 or NGSI-10 API calls then interoperation of the backend for our shock warning system, with the required sensor devices, becomes a valid possibility. Alternatively then it may have to be conceded that the sensor middleware implementations are required to interoperate by using instead a dedicated REST HTTP/S web service at the sensor side. This is necessary for the purpose of converting lower level sensor produced signals to NGSI-9 or NGSI-10 API calls that hence act upon a GE. Also an OCB's internal data model is required to be initialized with an XML representation of the IoT model i.e. an IoT setup is in effect deployed for a given use case implementing them.

The next parts of this section present more details regarding the FI-WARE IoT configuration manager GE, generic sensor device implementations and a description of the methodology that is necessary for establishing their mapping. In particular a map is required to create, within an OCB GE, the representation of IoT, as middleware, for aiding the connection of patients to sensors and also their connection with the remainder of the system's backend processing tier.

A. FI-WARE IoT Configuration Management GE

The Configuration Manager GE is responsible for context availability registrations. The model relies on the concept of context entities, which are generic entities whose state is described by values of attributes and metadata. In the context of IoT, context entities and context entity attributes can be used to model sensors and the variables they measure. Arbitrary physical objects i.e. wards, people, things etc. and their attributes i.e. HR and BP can be represented by this GE. The GE implements context information registries, whereby context provider applications can be registered. Interoperating systems can query on context registration information or subscribe for detecting changes to it. The GE enables IoT discovery and subscription of context information, through the NGSI-9 and NGSI-10 interfaces. The GE publishes the availability of context information. It stores the context information in a local repository. Specifically the GE receives registrations from IoT Gateways and *thing-level* adapters and stores this information in a configuration repository, therefore the registry is updated for any combination of connected sensors.

Using the approach, a GE can maintain context information that is not available in the IoT gateways or devices. For example, a gateway may not know the concept of patient data but only maintains a list of sensors and their measurements. The information regarding which specific sensors provide information about same patients is also maintained by the GE.

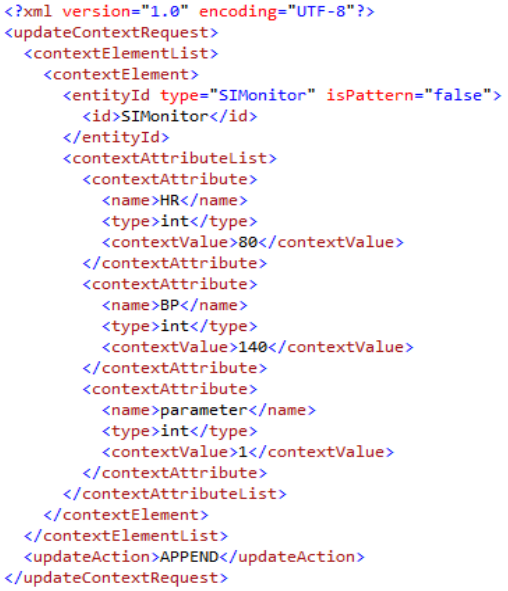
The Orion Context Broker is an implementation for the Configuration Manager GE and it provides the NGSI-9 and NGSI-10 network interfaces. The GE is intended to be used in combination with an IoT Broker GE (so the IoT Broker deals with NGSI-10 in a stateless fashion, relying on its repository as persistent storage for NGSI-9 registrations) although it can also be used as a standalone component with the NGSI-9 interface. Clients can do several operations such as register context producer applications, e.g. a HR sensor upon a patient, discover context information, e.g. which sensors are providing data for a given entity and being notified when changes on context information availability have occurred.

B. IoT Sensors

Contemporary IoT technology is generally based upon the fields of hardware, wireless public clouds and middleware [20]. Sensors are often referred to as being *seamlessly* integrated to form wireless networks of addressable resources. These resources are primarily accessed and controlled remotely using applications connected to the internet. IoT gateways are the bespoke interfaces connecting vendor sensor hardware implementations to networked applications for varied purposes, including for example fields within the FIA domains of smart cities, smart agriculture and smart connected health. IoT systems are macroscopic entities and there exists different routing procedures. Routing tasks are facilitated by heterogeneous platform instances and these are composed of different hardware components, required for the production of data and maintaining operational efficiency of the sensing components necessary of IoT applications. The components have interaction protocols and these can be affected by commands issued from internet connected homogenous control applications [21]. With regard to the myriad of presently available IoT sensor device implementations then a uniform API standard, for all sensors to utilize, is a requirement. REST HTTP/S permits M2M, C2C or P2P communications using a standardized and interoperable network application programming interface. IoT gateways should be constructed to this standard otherwise connecting sensors to an IoT cloud system is a complex challenge based on the need to transform from one representation of sensor data to another representation. REST HTTP request method payloads and XML or JSON data models are the ideal transportation solutions for IoT system applications. IoT systems are inherently multimodal [22]. The complexity of multiple types of sensor connected systems implies that there will be an increased chance of error conditions arising when combining the I/O modes of the custom interfaces for each vendor specific sensor device and their IoT gateway interfaces. The alternative proposal already hinted at is a solution which instead requests for the construction of REST HTTP/S compatible sensors and IoT gateways for the purpose of obtaining their seamless integration with holistic IoT system architecture back ends.

C. OCB GE IoT internal Mapping

A simple example is now used for demonstrating the method of mapping IOT sensors for a SCH shock warning system. Also shown is the method of how this can be achieved and represented with XML data models, uncovered from FI-STAR research. The data model setup for an OCB is demonstrated and describes the relevant sensor data types, for capturing HR and BP context information. Context subscriptions are used to enable the OCB to notify when changes in context, for data type values, occur. Context availability of data type values are also possible to be detected by the OCB and this feature enables publishing of data, using notification from the OCB.



```
<?xml version="1.0" encoding="UTF-8"?>
<updateContextRequest>
  <contextElementList>
    <contextElement>
      <entityId type="SIMonitor" isPattern="false">
        <id>SIMonitor</id>
      </entityId>
      <contextAttributeList>
        <contextAttribute>
          <name>HR</name>
          <type>int</type>
          <contextValue>80</contextValue>
        </contextAttribute>
        <contextAttribute>
          <name>BP</name>
          <type>int</type>
          <contextValue>140</contextValue>
        </contextAttribute>
        <contextAttribute>
          <name>parameter</name>
          <type>int</type>
          <contextValue>1</contextValue>
        </contextAttribute>
      </contextAttributeList>
    </contextElement>
  </contextElementList>
  <updateAction>APPEND</updateAction>
</updateContextRequest>
```

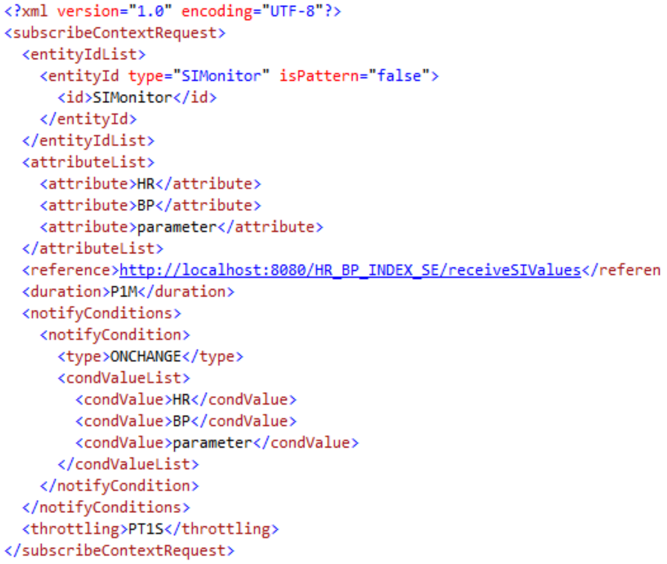
Fig. 3 NGSI-10 OCB Update Context Request XML data model.

Context data model creation, updates and subscriptions are initiated using the OCB NGSI-10 API and it is a REST HTTP compatible API. Context availability data model registration, updates and subscriptions are initiated using the OCB NGSI-9 API. Hence the HTTP POST request method enables XML payloads of data models to be transmitted to the OCB. Basic GET methods may also be used. However the purpose of demonstrating the setup and operation procedures for the OCB is to show how the GEs can be easily implemented as XML representations for modelling IoT i.e. to complement the backend processing tier's implementation of our shock warning system. The OCB will start in an empty state, so first it must obtain a representation of the existence of the required entities. Created will be HR, BP and parameter entities, each with arbitrary attributes representing HR, BP and parameter values. The creation of the entities is achieved using the OCB's updateContextRequest operation, to set the values within the OCB data model. Shown by Fig 3 is an XML data model, describing the IoT sensors for representing the data source devices of this cyber-physical system model. The data model is used to initialize the OCB for the reason of recognizing the IoT sensor devices and to provide transmission data formats in XML. The formats capture the input streaming data values. The data model provides sensor data types and values to the OCB and also to the other middleware parts, describing the context setup for the system's operation. With regard to the situation of implementing our shock warning system then the initial and primary context monitoring is for detecting changes in the streaming of HR, BP and parameter values over representational state transfer methods.

localhost:1026/NGSI10/updateContext, is an NGSI-10 API HTTP URI and this is called with the POST request method. As shown by Fig 3, the updateContext request method payload contains a list of contextElement elements and therefore a contextElement is associated to an entity. The identification is provided in the entityId element. Hence for the case of our example then present is the definition for *SIMonitor* and the data model also contains a listing for any contextAttribute elements. Each contextAttribute provides the value for a given attribute, identified by name and type of the entity. The payload includes also an updateAction element. Used too is APPEND, which implies that added will be new information to the data model.

Orion Context Broker has another powerful feature, the ability to subscribe to context information. Therefore when changes to data contexts happen then the application will get an asynchronous notification. Therefore it is unnecessary to continuously repeat queryContext requests i.e. polling, the OCB will let the application know the information when it arrives. The broker supports two subscription types. ONCHANGE subscriptions are used if notifications arrive when an attribute's context changes. ONTIMEINTERVAL subscriptions are useful for when notifications are triggered, after a given time interval has expired. For the purpose of our example only ONCHANGE subscriptions are considered. They are achieved using the OCB's subscribeContextRequest operation, set with the ONCHANGE notify condition type.

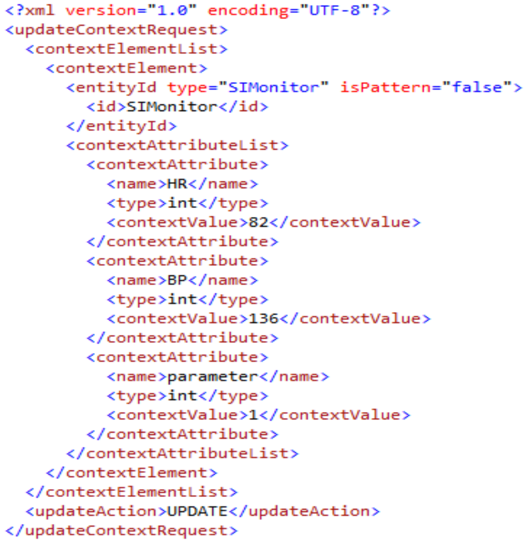
localhost:1026/NGSI10/sunscribeContext, is the NGSI-10 API HTTP URI and this is called with the POST request method. As shown in Fig 4, the subscribeContext request method payload contains, the notifyCondition element and uses the type ONCHANGE. A condValueList is defined and contains also an actual list of condValue elements, each one with an attribute name. They define the triggering values i.e. attributes, where upon creation/change due to entity creation or update, trigger a notification. The rule is that if at least one of the attributes in the list changes then a notification is sent.



```
<?xml version="1.0" encoding="UTF-8"?>
<subscribeContextRequest>
  <entityIdList>
    <entityId type="SIMonitor" isPattern="false">
      <id>SIMonitor</id>
    </entityId>
  </entityIdList>
  <attributeList>
    <attribute>HR</attribute>
    <attribute>BP</attribute>
    <attribute>parameter</attribute>
  </attributeList>
  <reference>http://localhost:8080/HR_BP_INDEX_SE/receiveSIValues</referen
  <duration>P1M</duration>
  <notifyConditions>
    <notifyCondition>
      <type>ONCHANGE</type>
      <condValueList>
        <condValue>HR</condValue>
        <condValue>BP</condValue>
        <condValue>parameter</condValue>
      </condValueList>
    </notifyCondition>
  </notifyConditions>
  <throttling>PT1S</throttling>
</subscribeContextRequest>
```

Fig. 4 NGSI-10 OCB Subscribe Context Request XML data model.

The intention is to produce a notification of the values of HR and BP, each time the values change. A throttling element is used to specify a minimum inter-notification arrival time e.g. setting throttling to 1 second, as in the XML in Fig 4, makes sure that a notification will not be sent if a previous notification is sent less than 1 second ago. The reference element of the payload, defines the destination to which OCB generated notifications are to be sent to, in this case a web service component of a shock warning system which is consuming sensor heart rate and blood pressure values as they are being read by the sensors.



```
<?xml version="1.0" encoding="UTF-8"?>
<updateContextRequest>
  <contextElementList>
    <contextElement>
      <entityId type="SIMonitor" isPattern="false">
        <id>SIMonitor</id>
      </entityId>
      <contextAttributeList>
        <contextAttribute>
          <name>HR</name>
          <type>int</type>
          <contextValue>82</contextValue>
        </contextAttribute>
        <contextAttribute>
          <name>BP</name>
          <type>int</type>
          <contextValue>136</contextValue>
        </contextAttribute>
        <contextAttribute>
          <name>parameter</name>
          <type>int</type>
          <contextValue>1</contextValue>
        </contextAttribute>
      </contextAttributeList>
    </contextElement>
  </contextElementList>
  <updateAction>UPDATE</updateAction>
</updateContextRequest>
```

Fig. 5 NGSI-10 OCB Update Context Request XML data model.

Updates to the values of the data model's entity attributes are accomplished by using the updateContext operation with another setting for an UPDATE action type. Therefore, APPEND creates new context elements, while UPDATE updates already existing context elements. Next another data model mapping of IoT sensors is used by our system for operating as a context producer application i.e. a source of context information. Assuming that the sensors detect the HR and BP of a patient as 82 and 136 respectively, then the sensors IoT GE from the design in Fig 2 issues the update request payload shown by Fig 5 to a security validation service SE. The IoT GE has been prior initialized from payloads described in Fig 3 and Fig 4. Now, as and when the sensor readings change then updates are published automatically to the backend processing tier of the system.

VII. Conclusion

In the absence of public servers or HTTPS compatible GEs then our system, to calculate and evaluate a shock index, is yet still not secure enough. This is a worry because the implication is that there may need to be applied a GE or SE that meets the ethical and legal security requirements for the prevention of any attempt at hacking the system. It's an interesting issue, there was a lot of discussion for example when Barnaby Jack demoed his pacemaker hack [23]. It's not that a hack here would "kill" the patient, but rather could prevent an alarm sounding, as such it should not be considered a critical failure because this is a monitoring system rather than a regulating system (like a pacemaker). It is an important point though so there is a requirement to investigate further the security surrounding REST technology implementations on a private cloud platform. WiFi enables 128bit WEP encryption within a single network. Secure Sockets Layer (SSL) and a public key infrastructure (PKI) is only possible between machines on separate networks connected over a public/private transport layer. So WEP or the stronger WPA enterprise wireless security protocols will have some basic security, in-built, for a private cloud offering. SSL is for public/private cloud offerings between two different networks connected together via a tunnel. Therefore communications between clients and a database are already encrypted at the point when the WiFi connection is set up and subsequently entered are pass keys to join the WiFi network. Anything else that is deemed to be more secure than this would have to be built in to each communicating node from the point of the network design i.e. to separately conduct the cryptography operations per transmission performed by each node in the system's design. In effect implementing SSL and PKI for an already WiFi-encrypted private cloud. FI-WARE currently does not directly support this type of technology in the form of reusable and simple to implement GEs. However security and privacy is ensured and deployed IoT middleware SE components are able to be constructed into legitimate but nonetheless bespoke MMAs because these can be constructed and applied within the guidance of a PKI. Hence certificate tokens are used within the use case design and implementation operating conditions. Under this context of operation the shock warning system is following the *software to data* paradigm [24], whereby the SCH setup is brought to the patient over a private cloud network. The reason for researching a transport layer security (TLS) solution in a private and ad hoc ambient, assisted living network setup is that originally envisaged was SSL certificates, provisioned over a public cloud FI-WARE core platform. This could have been enough to give adequate security features to meet requirements. SSL is useful for verifying users securely and having the ability to remove users easily by revoking their certificates. However the obvious security flaw is that Hippocratic patient data is being sent over a public network within a *data to software* paradigm. For legal and ethical MMA related requirements then this solution turns out to be insufficient. Therefor a private cloud offering is preferred and hence it is designed and implemented as an alternative proposition. Assuming the Orion Context Broker GE can be used in a peer to peer model then a method for building the concrete implementation consists of the use of a single laptop to host three separate ORION GEs, one for accepting raw data from the IOT sensors input interface, the second for handling patient's HRs and BPs directly for computation and then the third for estimating if the calculated SI critical events are determined as critical enough for the system to signal that the patient is in a critical state. Use of a workstation (Webserver) to host several SEs is required. The SEs follow a client or client-server (web services) hybrid model. A primary SE is required as a server component for maintaining control, start up, shut down, removal of accumulated records and coping with connections of secured data flows from scaled internet of things sensors. Also required are client driven SEs that initialize the GEs at startup and demonstrate end to end test and interaction/interoperability with ORION Context Broker GEs, acting now as secured MMAs. The system is comprised also of a workstation and mobile SEs for the requirements of medical staff and Doctors. From the design in Fig 2 the IOT_GE (MMA1) contains the initial screening data model for patient sensor setups. The data regarding patients is held internally and also securely (encrypted) because the GEs have been implemented according to the specifications of MMA SEs e.g. operating so as to function with PKI certificate tokens. The 3-tier architecture design also shows that REST_Database_SE (SE3) mediates the presentation tier with the remaining tiers of the system for security and logging requirements. The design describes Patient_Monitor_SE (SE4) sending messages of HR and BP values to the Patient_Records_Context_Broker_GE (MMA2), GE2 is constructed to hold the operational patient monitor XML data model for automatically referencing HR_BP_INDEX_SE (SE5) when SE4 sends validated HR and BP readings, per connected patient, to it. So, MMA2 forwards onto SE4 HR and BP values. SE5 computes the SI for valid readings that it receives and sends it in turn to Patient_BPI_Context_Broker GE (MMA3), as required for each patient being processed. The data defines a critical or noncritical entity type, for raising an alarm for a patient, if it is necessary. MMA3 is constructed to store the patient monitor system’s data model and it is setup to automatically notify Medical_Staff_TerminalSE (SE1), Doctor_MobileHandset_SE (SE2) or via mediation again with SE3. Generated message flows are directed towards the top presentation tier but only when a critical entity type for a SI is calculated by SE5, relevant to each individual patient. SE1 or SE2 then raises an alarm for a patient when the SI is received as critical for that patient. So in effect all SEs require client-server components for modelling the necessary web services interaction required to enable complete control and regulation throughout the entire system. Hence every SE requires a client-server implementation. The servers are difficult components to construct because necessary also is that each SE is constructed with a REST compatible web container therefore each SE will require further at least one specific implementation of a web/application server e.g. Glassfish supplied by Oracle with the Java EE distribution. Each server component, implemented by every SE, will also have to expose their own set of services, as well as the services to use from all GEs. Furthermore because several web/application servers are required then implementing the system simply with just a single laptop and the workstation is not possible. Required instead are more machines to host web/application servers, unless of course it is possible to have more than one instance of a web server container operating on one machine. This is generally unwise so a requirement instead is to use more machines for the implementation. A solution in general is to organize a laptop and workstation/server to host several Virtual Machines (VMs) in-turn hosting several web servers to scale the SCH setup to operate effectively for the needs of the institutions requiring its deployment.

Acknowledgment

The authors are members of the Future Internet – Social Technological Alignment Research (FI-STAR) project [25], partly funded by the European Commission and part of its Future Internet Private Public Partnership (FI-PPP). FI-STAR is a FI-PPP phase 2 project, which commenced on 1st April 2013 and will conduct early clinical and non-clinical digital-health use-case trials in seven European countries. This paper is based on research and analysis conducted by FI-STAR.

References

[1] European Commission. (2012, June). Communication of the European Commission to the European Parliament, the Council, the European Economic and Social Committee and the Community of Regions, e-Health Action Plan 2012-2020 – Innovative healthcare for the 21st Century. *.Digital Agenda for Europe.* [Online]. Available: http://ec.europa.eu/digital-agenda/en/innovative-healthcare-21st-century

[2] FI-STAR project partners. (2014, April). Project blog describing FI-STAR activities and events. [Online]. Available: http://fistarblog.com/

[3] Jaeho Lee. (2011, September). Smart health: Concepts and status of ubiquitous health with smartphone. Presented at International Conference on ICT Convergence (ICTC). [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6082623

[4] Jara, A.J.; Zamora, M.A.; Skarmeta, A.F. (2012, June). Knowledge Acquisition and Management Architecture for Mobile and Personal Health Environments Based on the Internet of Things. Presented at IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom) [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6296204

[5] X. Smith, Brendan; Phillips, Robert; Madigan, Veronica; West, Malcolm, 2012, 1023: Decreased Mortality, Morbidity and Emergency Transport in Septic Shock; A New Protocol Based on Advanced Noninvasive Haemodynamics and Early Antibiotics, Critical Care Medicine, December 2012

[6] J. P. Brogan; Christoph Thuemmler, “Specification of Generic Enablers as Software,” to be published at 11th International Conference on Information Technology: New Generations, Las Vegas, Conference, Technology, IEEE Computer Society.

[7] FI-WARE. (2014, April). FI-WARE Catalogue GEs. [Online]. Available: http://catalogue.fi-ware.org/

[8] Stojmenovic, I. (2014, March). Machine-to-Machine Communications with In-network Data Aggregation, Processing and Actuation for Large Scale Cyber-Physical Systems. Internet of Things Journal, IEEE. [Online]. *pp (99), 1*. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6766661

[9] Thuemmler, C.; Mival, O.; Benyon, D.; Buchanan, W.; Paulin, A. (2013, October). Norms and standards in modular medical architectures. Presented at IEEE 15th International Conference on e-Health Networking, Applications & Services (Healthcom) [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6720705

[10] Stankovic, J.A.. (2014, March). Research Directions for the Internet of Things. *Internet of Things Journal, IEEE.* [Online]. *pp (99),* 1. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6774858

[11] *Rady MY1, Nightingale P, Little RA, Edwards JD.* (1992, June). Shock index: a re-evaluation in acute circulatory failure. Resuscitation. 23(3):227-34.. Available: http://www.ncbi.nlm.nih.gov/pubmed/1321482#

[12] *West J Emerg Med.* Mar 2013; 14(2): 168–174.
doi: 10.5811/westjem.2012.8.11546
PMCID: PMC3628475
Shock Index and Early Recognition of Sepsis in the Emergency Department: Pilot Study

[13] FI-WARE. (2014, April). Publish/Subscribe Context Broker – Orion Context Broker. [Online]. Available: http://catalogue.fi-ware.org/enablers/publishsubscribe-context-broker-orion-context-broker

[14] Bauer, M. ; Kovacs, E. ; Schülke, A. ; Ito, N. ; Criminisi, C. ; Goix, L.-W. ; Valla, M. (2013, October). The Context API in the OMA Next Generation Service Interface. 2010 14th International Conference on Intelligence in Next Generation Networks (ICIN). [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5640931

[15] Shan, T.C. ; Hua, W. (2006, September). Solution Architecture for N-Tier Applications. IEEE International Conference on Services Computing, 2006. SCC '06. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4026951

[16] FI-WARE. (2014, April). Configuration Manager – Orion Context. [Online]. Available: http://catalogue.fi-ware.org/enablers/configuration-manager-orion-context-broker

[17] Ferreira, A. ; Bernardos, A.M. ; Bergesio, L. ; Casar, J.R. (2013, December). ARTHE: Experiencing Projected Augmented Reality with THings of the Everyday, Presented at IEEE 10th International Conference on and 10th International Conference on Autonomic and Trusted Computing (UIC/ATC), Ubiquitous Intelligence and Computing. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6726249

[18] Zanella, A. ; Bui, N. ; Castellani, A. ; Vangelista, L. ; Zorzi, M. (2014, February). Internet of Things for Smart Cities. Internet of Things Journal, IEEE. [Online]. *pp (99), 1*. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6740844

[19] Wei Shen ; Youzhi Xu ; Donghua Xie ; Zhang, T. ; Johansson, A. (2011, September). Smart Border Routers for eHealthCare Wireless Sensor Networks, Presenyted at IEEE 15th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM) [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6040606

[20] Qian Zhu; Ruicong Wang; Qi Chen; Yan Liu; Weijun Qin. (2010, December). IOT Gateway: BridgingWireless Sensor Networks into Internet of Things. Presented at 8th International Conference on Embedded and Ubiquitous Computing (EUC), 2010 IEEE/IFIP. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5703542

[21] Galache, J.A.; Gutierrez, V.; Aguero, R.; Munoz, L., (2007, October). Towards the Integration of Heterogeneous Wireless Sensor Platforms: A generic API Approach, Presented at International Conference on Sensor Technologies and Applications, 2007. SensorComm 2007. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4394956

[22] Bruno Dumas , Denis Lalanne , Sharon Oviatt, "*Multimodal Interfaces: A Survey of Principles, Models and Frameworks*", Human Machine Interaction, Pages 3-26, Springer-Verlag Berlin, Heidelberg, 2009.

[23] Bansai, R. (2008, December). Jolt from the Blue? Antennas and Propagation Magazine, IEEE. [Online]. Volume:50 , Issue: 6, Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4768947

[24] Thuemmler, C.; Mueller, J.; Covaci, S.; Magedanz, T.; de Panfilis, S.; Jell, T.; Schneider, A.; Gavras, A. (2013, April). Applying the Software-to-Data Paradigm in Next Generation E-Health Hybrid Clouds, Presented at Tenth International Conference on Information Technology: New Generations (ITNG) [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6614349

[25] FI-STAR project partners. (2014, May). Project website describing FI-STAR activities and events. [Online]. Available: https://www.fi-star.eu/home.html

John P. Brogan received the MEng degree in Software Engineering from Heriot-Watt University, Edinburgh, in 2006.

From 2006 to 2009, he was a Research Associate with the School of Mathematics and Computer Science at Heriot Watt University. Since 2009, he has been a Software Developer with the University of Edinburgh's School of Biological Sciences and he is presently a Research Fellow with the Institute for Informatics & Digital Innovation at Edinburgh Napier University.

His research interests include Information Systems, e-Health applications and Software Engineering.

Dr. Christoph Thuemmler studied Medicine, Political Science and Educational Science and is a fully trained specialist in General Internal Medicine with 20 years of clinical experience. He has worked in Germany, the US and the UK before focusing more and more on smart devices and service oriented architectures in health care. He completed his PhD on Cerebral Haemodynamics with distinction at Heidelberg University, Germany.

Over the last 10 years he has been involved in extensive knowledge transfer activities, especially in promoting the integration of life sciences, clinical practice and computing. His research interests lie in the area of the Future Internet including the Internet of Things, work on social technological alignment in health care and the governance of e-Health in a wider context.

Dr. Samuel A. Fricker is assistant professor in the Software Engineering Research Laboratory (SERL) at Blekinge Institute of Technology (BTH), interested in software product management and requirements engineering. He has more than ten years of experience as senior consultant, is global process responsible, lecturer, and senior researcher with companies at any scale, from startups to Fortune500.

Dr. Oli Mival is an internationally recognised expert in the areas of Human Computer Interaction (HCI), User Experience (UX) and Interaction Design (ID). His research interests include also e-Health applications. As the designer and originator of the ICE (Interactive Collaborator Environment). Oli has been invited to consult for a wide range of global industrial and government clients including the NHS, Astra Zeneca, Enquest Oil, CGI, Blue Rubicon, and Redlands Police Department in California.