Internet of Things Architecture

IoT-A

Project Deliverable D7.2 - Exact definition use case 1 and use case 2

Project acronym:	IOT-A
Project full title:	The Internet of Things Architecture
Grant agreement no.:	257521

Doc. Ref.:	D7.2	
Responsible Beneficiary :	FHG IML	
Editor(s):	Martin Fiedler (FHG IML)	
List of contributors:	Nicola Bui (CFR) Jourik De Loof (ALU BE) Edward Ho (HSG) Carsten Magerkurth (SAP) Benedikt Mättig (FHG IML) Gregorio Martin Romero (TID) Olivier Savry (CEA/LETI) Alessandro Serbanati (CATTID) Ebru Zeybek (CATTID)	
Reviewers:	Gregorio Martin Romero (TID) Stephan Haller (SAP) Markus Hinkelmann (NXP-DE) Edward Ho (HSG) Alessandro Serbanati (CATTID)	
Contractual Delivery Date:	29/02/2012	
Actual Delivery Date:		
Status:	Final	
Version and date	Changes	Reviewers / Editors
v0 – 09/01/2012	Table of contents	MF
v1 – 29/02/2012	Initial version before review	MF
v2 – 02/03/2012	2^{nd} Review Health Section	GMR
v3 – 03/03/2012	2^{nd} Review Retail Section	EH
v4 – 16/03/2012	Final Editing	MF

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2013)

	Dissemination Level	PP
PU	~~Public~~	
PP	Restricted to other programme participants (including the Commission Services)	
RE	~~Restricted to a group specified by the Consortium (including the Commission Services)~~	
CO	~~Confidential, only for members of the Consortium (including the Commission Services)~~	

Executive Summary

The objective of this deliverable, D7.2, is to define the exact use cases which would demonstrate the application of the Architecture Reference Model (ARM) developed in WP1 of the IoT-A project. The deliverable defines use cases in the two domains health and retail, which are linked story wise.

This deliverable follows the report D7.1, which provided an initial definition of the use cases. The use case collection in D7.1 was combined to an overall storyline in D7.2. The structure of D7.2 is as follows. In the first section, the document explains the use case development process, from its initial conception in D7.1 to the refinement of the use cases here in D7.2, and the subsequent integration of components and concepts from the technical work packages of the project. This deliverable shall be followed by the initial implementation and result in first demonstrators within D7.3 which will be used for stakeholder feedback.

While the focus of WP1 is centered on developing an Architecture Reference Model (ARM), which can be applied in developing Internet of Things systems, D7.2 addresses the **Application** side of the overall project. In the role of integrating the results of the other work packages, this deliverable applies first results of the ARM regarding modelling of use cases in the IoT domain. Furthermore D7.2 gives an outlook on what kind of devices (i.e. hardware of WP5) or components (i.e. WP2's service orchestration, WP3's communication stack or resolution framework of WP4) could be used for an implementation on a per use case view.

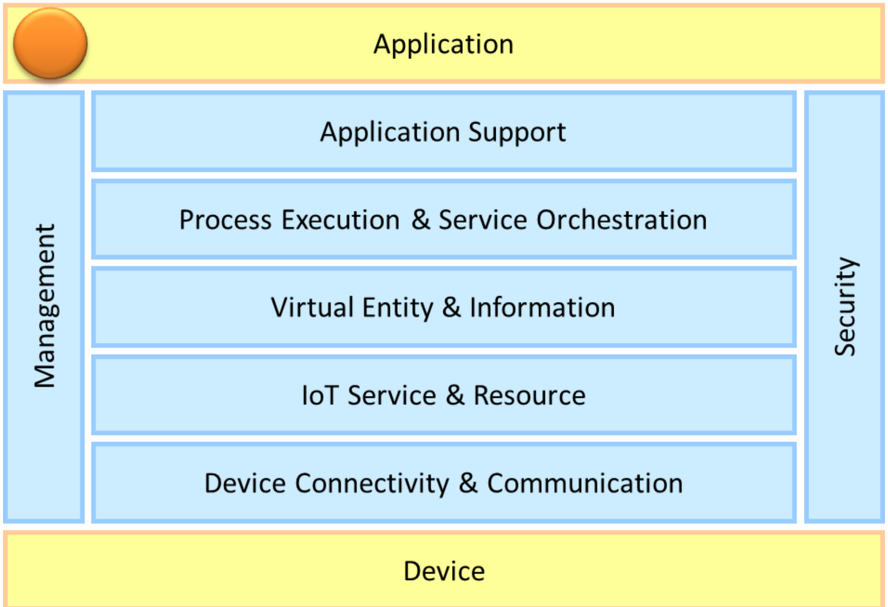

Figure 1: Functional Groups tackled in D7.2

Table of Content

List of abbreviations

Abbreviation	Description
ADA	Active Digital Artifact
API	Application Programming Interface
ARM	Architecture Reference Model
ATP	Availability to Promise
D	Deliverable
DA	Digital Artifact
EHR	Electronic Health Record
ERP	Enterprise Resource Planning
ESL	Electronic Shelf Labels
GUI	Graphical User Interface
HIS	Hospital Information System
ID	Identification
NFC	Near Field Communication
PDA	Passive Digital Artifact
PE	Physical Entity
PoC	Point of Care
POS	Point of Sale
RFID	Radio Frequency Identification
TFT	Thin-film transistor
UID	User Identifier
UML	Unified Modeling Language
URL	Uniform Resource Locator
VE	Virtual Entitiy
Wi-Fi	Wireless Fidelity
WP	Work Package

List of figures

List of tables

1. Introduction

The objective of this deliverable, D7.2, is to define the exact use cases which would demonstrate the application of the architecture reference model developed in Work Package one (WP1) of the Internet of Things Architecture (IoT-A) Project. WP1's focus is centered on developing an architecture reference model (ARM), which can be applied in developing Internet of Things systems. The concepts in the ARM were used in mapping the functional view of the system and in determining common components which could be used interoperably between subsystems in the use case. Further details of the architectural reference model mentioned in this deliverable can be found in the public deliverable D1.2 Initial architectural reference model for the IoT (Walewski, et al., 2011) and the internal deliverable IR1.4 Updated initial reference model (Nettsträter, et al., 2011).

The use cases described in D7.2 are concentrated in the areas of health care and retail. These two domains are the focus due to their importance and relevance, but as IoT-A aims to provide a generic Internet of Things architecture; it should also be applicable to other major domains such as manufacturing, entertainment, or defense. The two domains considered have in common that they affect many people - now and in the future. An interconnection exists in nutrition and health, where many people tend to look for healthy food to prevent diseases; others act in accordance with a health plan prepared by their doctors after a diagnosis. Technology can be used to support both cases and makes it easier to eat and stay healthy.

This deliverable follows D7.1 (Hagedorn, et al., 2011), which provided an initial definition of the use cases. The structure of D7.2 is as follows. In the first section, the document explains the use case development process, from its initial conception in D7.1 to the refinement of the use cases here in D7.2, and the subsequent integration of components and concepts from the technical work packages of the project.

The next sections detail each of the use cases in health and retail, and elaborate beyond the storyline found in D7.1 by providing class diagrams which use modeling components and concepts from WP1, and by listing the components from the other work packages which will be used in the implementation of the use cases. The class diagrams illustrate the domain model concepts found in IR1.4 and D1.2 and by modelling each use case with a class diagram, the link between the use case application and the ideas from the architecture reference model is established. In this manner, the link to the other aspects of the IoT-A project is provided.

This deliverable shall be followed by the initial implementation of the use cases in M22 (June 2012).

2. Use Cases development process

Deliverable D7.1 provided an initial definition of the use cases. These use cases were independently developed by the different partners; the use cases reflected the ideas the different partners had on how the Internet of Things would look like based on their experience and available technology. How the proposed use cases could be integrated into one demonstrator per domain is covered in D7.2.

The goal of D7.2 is to make a selection of the use cases from the individual contributions in D7.1 and come to a consolidated use case per domain with a realistic storyline that shows the strengths of the underlying technology under development by the different technical work packages. The demonstrators of these use cases should be appealing to a large audience and clearly illustrate the concepts of the reference architecture developed in WP1.

A first step in the development process was the decision to either continue with separate use cases per domain or to define one single use case per domain.
Going to one single use case per domain would allow for a more coherent demonstrator but would at the same time require a high level of integration from the beginning between the contributions of the individual partners.

Separate use cases would allow for more flexibility, but at the cost of a less consistent storyline. The decision was to work with one use case per domain, divided in different scenes. A scene is defined as an action in a single location and continuous in time, that can be implemented and demonstrated in a standalone way. This gives the partners the flexibility to independently develop the use case without sacrificing the coherency of the storyline.

A next step in the development process was to define and identify a scene in both the Health and Retail use cases that could be linked together. Joining the Health and Retail use cases emphasizes the strength of the architecture and illustrates that the architecture is still valid in a cross Application domain scenario.

The last step in the development process is to present the use cases to the stakeholders and ask for their feedback on how realistic the use cases and their objectives are.
Originally, it was foreseen to present D7.1 to the stakeholders and take their feedback into account during the definition of the use cases of D7.2. It was decided however to first define the consolidated use cases and in a second step ask for their feedback. The feedback process consists of presenting a slide set to the stakeholders where the two use cases and their respective scenes are described in detail, together with a questionnaire. The questionnaire will ask stakeholders to validate the realism and relevance of the use cases in their respective domains. The slide set should be ready at the same time as deliverable D7.2.

Deliverable D7.2 will be followed by deliverable D7.3 which will present the prototypes of both use cases and deliverable D7.4 which is the demonstration of the use cases. The questionnaire results from the aforementioned slide set shall be used as inputs for the feedback process and might result in changes in the description or implementation of the scenes of deliverable D7.4. It is important to note that the timeline of D7.3 does not allow the results of the technical work packages to be already included. Most technical work packages are still working on conceptual ideas and do not have a finalized view yet, let alone a working implementation.
Therefore, for D7.3, missing components from the technical work packages will be emulated by available hardware or by implementation in WP7 of missing software components.
As the technical work packages gradually release the missing components, integration is foreseen during the last year. For the final demonstration, a full integration is foreseen.

A schematic overview of the activities and content of the different deliverables, starting from D7.1 and leading to D7.4 can be found in Figure 2.

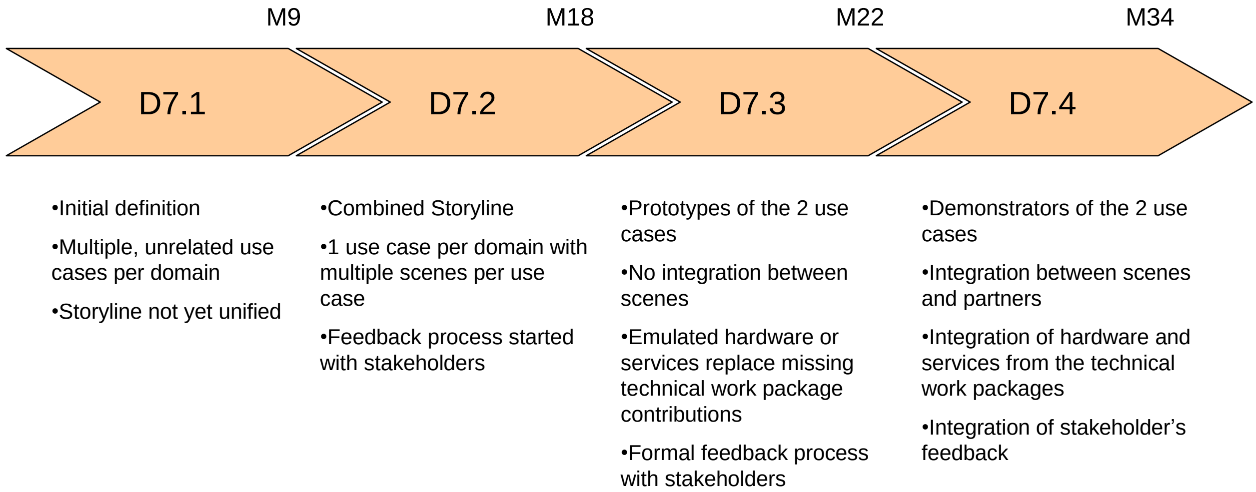

Figure 2: Overview of use cases development process

3. Use Case I: Health and Home

3.1 Defined Storyline

After having enjoyed a nice dinner with his daughter Salomée the night before, 55 year old Robert wakes up in the morning. Robert is proud of his daughter whom he has a good contact to. It's always nice meeting with her, as Salomée has a lot to talk about, which reminds him a bit of himself when he was younger and more healthy. Robert is suffering from high blood pressure and is a type II diabetes and since he has already suffered from a heart attack, he is considered a high risk patient. He is participating in a program organized by his health insurance company where his health condition is continuously and remotely monitored. Since his wife died a while ago, Salomée is registered at the health insurance company as a family member who supports Robert on tasks regarding housekeeping and simple medical care. This relation is stored in Robert's electronic health record (EHR).

This morning Robert is still thinking about the things Salomée talked about the night before so he forgets his IoT-Phone in his sleeping room. The IoT-Phone is Robert's new IoT-capable smartphone which Salomée told him a lot about. Normally Robert carries his IoT-Phone everywhere he goes. A backend system usually reminds him to take medical measurements on a daily routine which is usually three times a day. Now, Robert is not able to hear the alarm of the reminder. Since the alarm is not acknowledged, the system looks for nearby IoT devices like lights or buzzers in the vicinity of Robert and uses those devices to draw his attention. Robert sees the blinking of the lights in his living room and instantly remembers the meaning of this, as it has happened before. He goes to his sleeping room and picks up the IoT-Phone to acknowledge the alarm. The application guides him through the measurements he has to take.

Robert acknowledges the alarm, and is guided through the measurements by the application on his smart tablet. He has to take measure of his blood pressure, heart beat, blood glucose level, current weight and an indication on the activity he was doing. All measurements are stored in the system and an automatic analysis is performed, notifying his doctor in case some values are outside the normal range. The system calculates the amount of insulin he must inject. As Robert has to take more insulin as usual he takes his last NFC-tagged ampoule of insulin out of his medicine cupboard which immediately recognizes his action. The insulin stock level in Robert's medicine cupboard is tracked and as soon as it reaches a predefined refill-level an alarm is

raised. The IoT-Phone of Salomée is notified to buy insulin on behalf of Robert at a near pharmacy as she helps him.
After taking his insulin dose, his electronic health record is updated accordingly.

During his further day activity, Robert suddenly feels lightheaded and he presses a panic button he is wearing as part of a bracelet. The system detects that his mother Jane, which he lives with in the same flat, is nearby and notifies her. Jane looks for Robert and sees there is no need for further action since Robert already took a candy bar he always carries with him.

In the afternoon, Robert leaves his flat after lunch time. Robert is driving with his car to visit his daughter, when he is involved in a car accident. The other driver of the old-timer combustion engine powered vehicle must have overseen him under the bad weather conditions. Luckily, the acceleration sensor of Robert’s IoT-Phone instantly recognizes that something dangerous may have happened and queries his body sensors about Robert’s condition. The devices agree that Robert is in danger and, after a short time during which Robert can confirm he is safe, an emergency message sending the location data of where the dangerous condition arose as well as his personal ID to the emergency centre is released. Using location based lookup the nearest emergency centre is alerted and asked to send an ambulance to Robert immediately. The ambulance arrives at the car crash within 7 minutes and picks Robert and the other driver up to bring him to a hospital.

Arriving at the hospital, the check-in is quick, even though Robert is unconscious. Fortunately, the check-in procedures can be directly performed through interactions between Robert’s Citizenship ID card or Health ID Card and the hospital admission desk. The clerk first looks for Robert's Health ID Card, but is not able to find it. He finds only the Citizenship ID card which he may also use to check Robert in. With the help of an IoT-enabled mouse Robert may be identified with his citizen ID number by the local hospital software which may be used to grant the hospital access to some data of the citizenship database needed for check in and also allows the receptionist to look up all necessary medical insurance data as well as his entire medical file – that makes it easy to prepare all helpful information for the doctor beforehand without time consuming effort. And again it pays out to have signed for the program as Robert gets precedence over a young man who seems to have broken his arm but still spends time filling in all the papers. After the quick check-in Robert is taken care of by the further medical personnel who treat his wounds.

The hospital Robert is staying is equipped with the Hospital Information System (HIS). This system continuously monitors the environmental conditions (temperature, humidity) in the rooms and avoids that wrong medicines are administered to the patients. Robert had to stay at the hospital overnight as the doctors had to check how he reacts on the medical treatment. The next day during the morning routine, the temperature readings of Robert and a fellow patient staying in the same room, are too high indicating a small fever. However, an analysis shows that the room temperature was too low due to a failure of the heating system. A facility manager is automatically called by the HIS to repair the defect.

Upon the further medical treatment and monitoring of Robert a nurse is coming to Robert two times a day. The nurse administers medication to the Robert, as he needs medication against the pain which is caused by his wounds. In the evening, the nurse scans Robert's identification tag followed by the box of medicine (ampoule)'s tag when suddenly an alarm is raised. The medicine she is about to administer is the correct one, but the dose of the ampoule is too high due to an error in the pharmacy of the hospital.

The moment he finishes the exercise, the house bell rings. It is Salomée, bringing his monthly ration of insulin. Robert is once again pleased to have these little smart things in and around him that help on his daily routine.

3.2 Demonstrated concepts

This section gives an overview on what kind of concepts of the ARM may be demonstrated by the defined storyline of the health use case. Therefore a mapping of the UC's single scenes,

which are detailed in section 3.3, to the functional components of the ARM's functional view was made.

Figure 3 shows the current development status[1] of the functional groups and functional components after IR1.4 (Nettsträter, et al., 2011) but before the delivery of D1.3. It was decided to use an intermediate version of the functional view, because a fundamental change had already been done and the version out of IR1.4 or even D1.2 would have been too different. The version shown in the figure will be closer to the final functional view of D1.3.

[1] The delivery date of this document

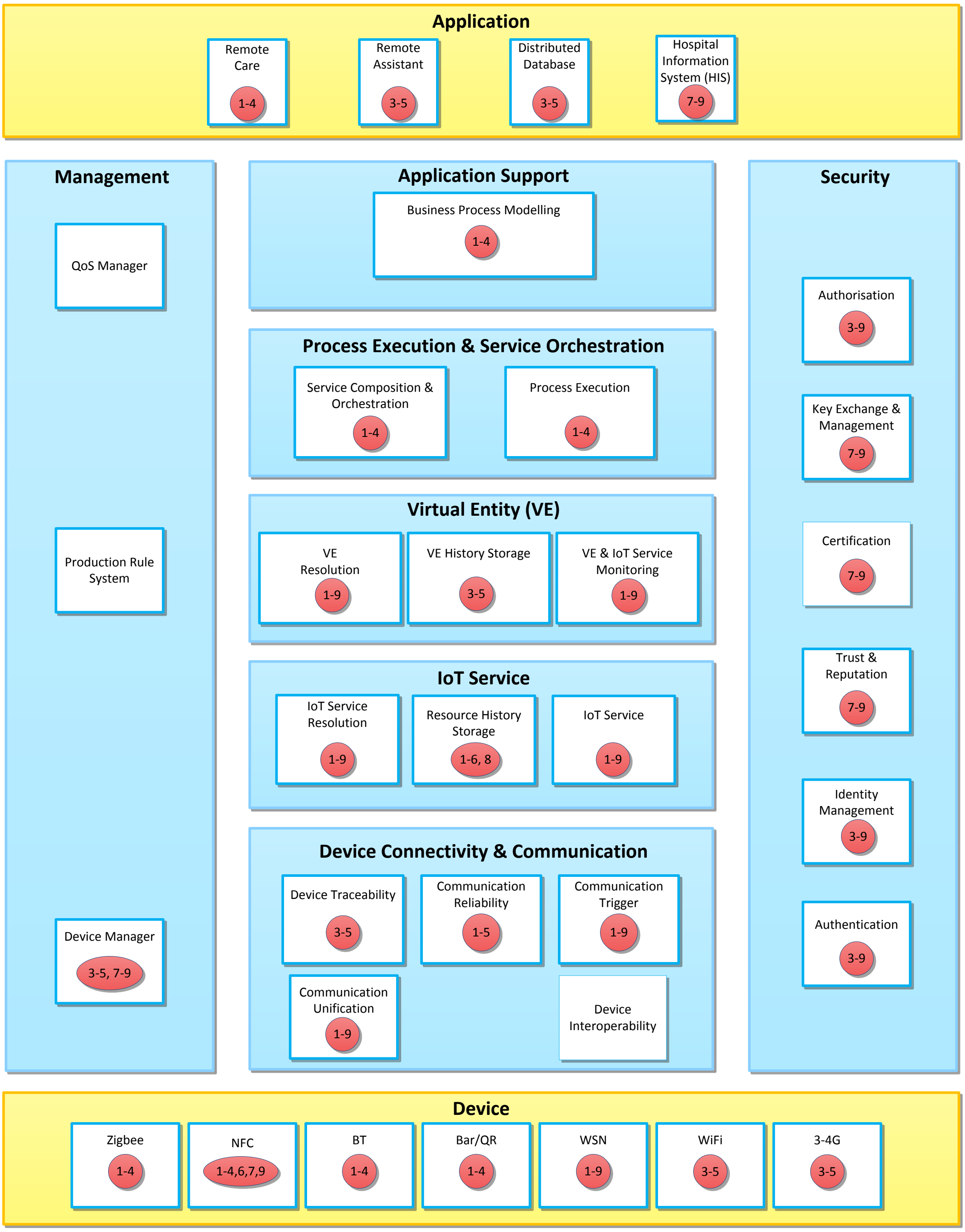

Figure 3: Functional component mapping of defined Health use case scenes

The figure shows the scenes 1 to 9 mapped to functional components of the ARM. A scene is shown as red circle, while a functional component is shown as white box. The numbers in the red circle state a specific scene number or a range of scenes which are mapped. As an example every health scene may demonstrate the concept (the usage) of the VE Resolution and IoT Service Resolution. Please note that a mapping does not necessarily imply that a mapped component is needed for a later implementation, but instead shows what ARM concepts could be demonstrated.

3.3 Description of Scenes

3.3.1 Users and Roles

Users	Roles	Descriptions
Nurse, Dietician, Doctor	Professionals	Healthcare professionals benefit from the IoT-A with faster access to information from different sources, and being able to deliver better services due to being better informed.
Salomée	Relative of Robert	Salomée (f, 29) represents the younger demographic who benefits from IoT-A enabled applications. This demographic is also tech savvy and highly proficient with new technologies.
Pharmacist	Retailer in the area of Health	The pharmacist is similar to a retailer at the Point of Sales, but dealing with health products.
Robert	Patient and relative of Salomée	Robert (m, 55) is suffering from type II diabetes, hypertension and faints from time to time by acute syndromes. He also suffers from high blood pressure and is a bit overweight. Apart from this he is generally in a healthy condition and mobile. Robert is living together with his mother Jane.
Jane	Mother of Robert	Jane (f, 82) must be reminded to take her medicines a couple of times a day. Jane is living together with her son Robert.

Table 1: Users and Roles in the health care use cases

3.3.2 Scene 1 – Remote Patient Notification (ALU-BE)

In the first scene of the Health use case, the Remote Patient Care application notifies the patient that some actions are required be taken by the patient
These actions can be related to administering medicines or to taking measurements on a regular interval.
Patients will carry personal devices such as smart phones or tablets which can become IoT-A enabled. Applications running on these devices can hence make use of all functions of the IoT-A compliant platform.
In this scene, the patient is notified by ringing an alarm on his IoT-Phone. This alarm is not acknowledged so the application will look for nearby resources such as light switches or buzzers in the vicinity of the last known location of the patient and use these devices to draw his attention.
The scene ends when the patient finally acknowledges the alarm.

Scene summary:

- Location: this scene is at Robert's house
- Covered topics:
    - Home Care
    - Remote health assistance
    - Remote health monitoring
    - Smart Objects
    - Service Discovery
- Key benefits:
  By using IoT-A enabled devices, everyday objects, not part of a particular application domain can be seamlessly integrated and used by an application.

3.3.2.1 Description from a User's Perspective

Robert is suffering from high blood pressure and from type II diabetes. Since he has already suffered a heart attack, he is considered as a high risk patient and he is participating in a program organized by his health insurance company where his health is continuously and remotely monitored.

Every morning, the alarm on Robert's IoT-Phone sounds indicating that it is time for his daily routine. Normally, Robert carries his IoT-Phone everywhere he goes, but today he has forgotten the IoT_phone in his sleeping room.

Since the alarm is not acknowledged, nearby resources such as lights or buzzers in the vicinity of the last know location of Robert are used to draw Robert's attention.

Robert is having breakfast, when suddenly the lights in his house start to blink slowly. He recognizes the message and starts looking for his IoT-Phone.

Robert finds his IoT-Phone in his sleeping room and acknowledges the alarm.

3.3.2.2 Relation to the Domain Model

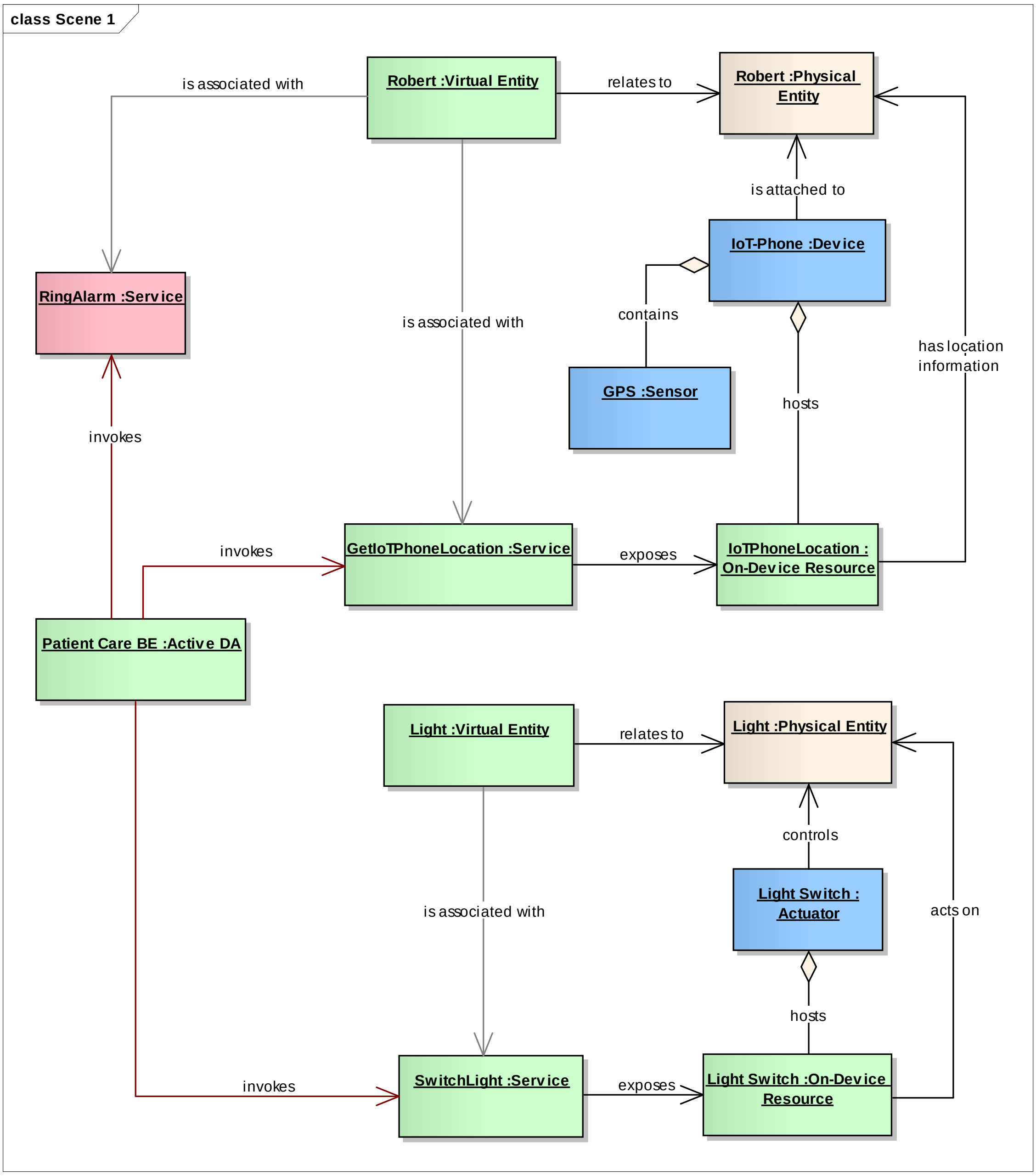

Figure 4: Class instance diagram of Scene 1: remote patient notification

The projection of the system components on the domain model is depicted in the class instance diagram of Figure 4.
Robert is the patient and is hence modeled as a *Physical Entity*.
Corresponding to the *Physical Entity* we have the *Virtual Entitiy* of Robert, which is the representation of Robert in the digital domain.
Robert's *Virtual Entity* is associated with an alarm service. This service is running on the IoT-Phone but is considered a legacy service and therefore not further detailed. The alarm service is however integrated in the resolution framework and an association between this service and the *Virtual Entity* of Robert is made.

Robert's IoT-Phone is a *Device* which contains a GPS *Sensor* for location determination. The information from the GPS sensor is modeled as an *On-Device Resource*, exposed by the IoT-A *Service* GetSmartPhoneLocation.
Robert's *Virtual Entity* is associated with the GetSmartPhoneLocation *Service* so that if an application wants to find out the location of the *Physical Entity* Robert, it can do a simple lookup on the related *Virtual Entity* and find it.
The application in charge of driving the first scene is run in the backend and is the *Active Digital Artefact* Patient Care BE.
In the first scene a discovery mechanism is also illustrated to find a nearby light.
The light is a *Physical Entity* with a corresponding *Virtual Entity*. The light switch controlling the light is an *Actuator* and hosts the Light Switch *On-Device Resource*. The SwitchLight service then gives access to this *On-Device Resource*.
The *Virtual Entity* of the light is associated with the *Service* to switch the light on or off.
The Patient Care BE *Digital Artefact* will invoke the resolution framework and do a discovery for a light service which is located in a certain perimeter from Robert's last location. This service can then be invoked by the application to switch the lights on or off.

3.3.2.3 Implementation details

The following table contains software, hardware and components which are planned to be used for the implementation of this scene. Details on the individual items may be seen in chapter 6.

Applications	Devices	Functional Components
SW.ALUBE.1 (Patient Care BE App)	HW.ALUBE.1 (IoT-Phone)	C.ALUBE.1 (Virtual-entity Resolution)
SW.ALUBE.2 (Patient Care Mobile App)	HW.ALUBE.2 (Switch)	C.ALUBE.2 (IoT Service Resolution)
	HW.ALUBE.3 (Light)	

Table 2: Applications, devices and functional components planned for scene 1 implementation

3.3.3 Scene 2 – Remote Patient Measurements (ALU-BE)

A majority of elderly adults suffer from chronic or acute illnesses. Remote measurements can help to minimize hospital stays or ensure premium patient care beyond the hospital room reducing the cost of overall healthcare. Additionally, remote measurements can alert caretakers in case of injury or harm to the patient and assure a prompt medical intervention. For elderly patients with beginning dementia, special attention should be given to reminding the patient that an action such as taking a glucose measurement or taking insulin is needed.

In the second scene, the patient will be assisted by the application in taking regular measurements such as blood glucose level or blood pressure.
After measurements are submitted to the Electronic Health Record (EHR) of the patient, automatic data analysis will take place and if needed, a caretaker will be notified.

Scene summary:

- Location: this scene is at Robert's house
- Covered topics:
  - Home Care
  - Remote health assistance
  - Remote health monitoring
  - Smart Objects
- Key benefits:
  The main key benefit shown in this scene is the use of devices of different technologies that all are integrated and seamlessly interact.

3.3.3.1 Description from a User's Perspective

Robert picks up his IoT-Phone and sees that it is time to take his daily measurements.

The application guides Robert through the measurements. First weight, then blood pressure and finally blood glucose level is measured. In case it would not be clear how to take the measurement, he could ask for detailed instructions and guidance.

After each measurement, Robert has to confirm the data. The confirmation of the data is done using biometric identification. Since some of the medical devices are shared between different patients, it is of utmost importance that there is no possibility for confusion between the measurements of different persons.

Robert places his finger on the fingerprint reader which submits the data to his EHR and triggers the automatic data analysis. If the measurements are within normal range, the data is archived with no further action taken.

In case the automatic analysis shows an anomaly, the caretaker is notified so he can login to the system and analyze the measurements to see if any action is needed or any adjustment to the medication should be proposed.

3.3.3.2 Relation to the Domain Model

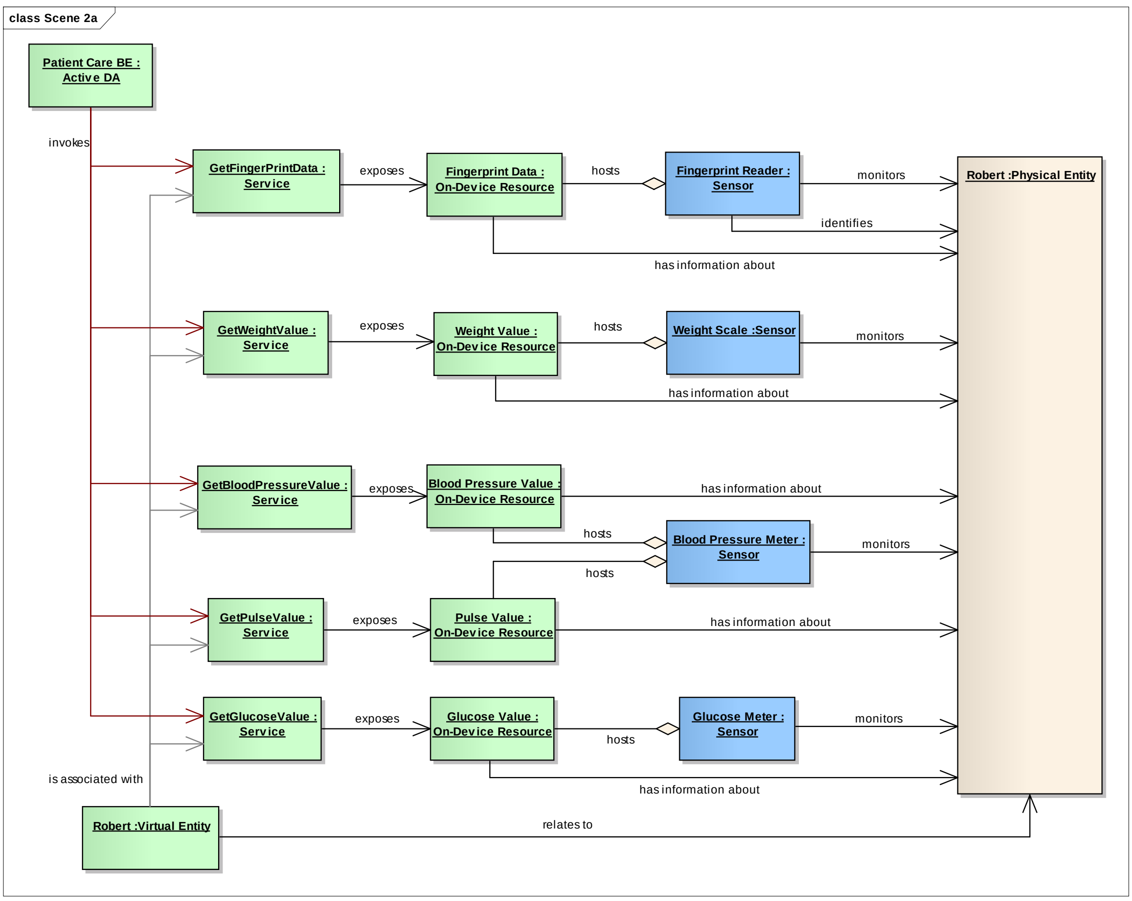

Figure 5: Class instance diagram of Scene 2: taking measurements

The modelling of the different sensors used in scene 2 for taking the weight, blood pressure and blood glucose level is quite straightforward. The patient is Robert, modeled as a *Physical Entity*, with a corresponding *Virtual Entity* in the digital domain.
Each device such as the weight scale or the blood pressure meter is modeled as a *Sensor*, hosting an *On-Device Resource*. This *On-Device Resource* is then exposed by a *Service*.
For easy lookup, the Virtual Entity of Robert is associated with the *Service* corresponding to each device.
For confirmation of the data, a fingerprint reader is used. This reader is modeled in the diagram as a *Sensor*, with an *On-Device Resource* and exposed by a *Service*.
Again, the *Virtual Entity* of Robert is associated with this *Service*.
Finally, the backend application Patient Care BE, which is an *Active Digital Artefact* (and thus a *User*) will look up the associations to find the different services, and then invoke them.

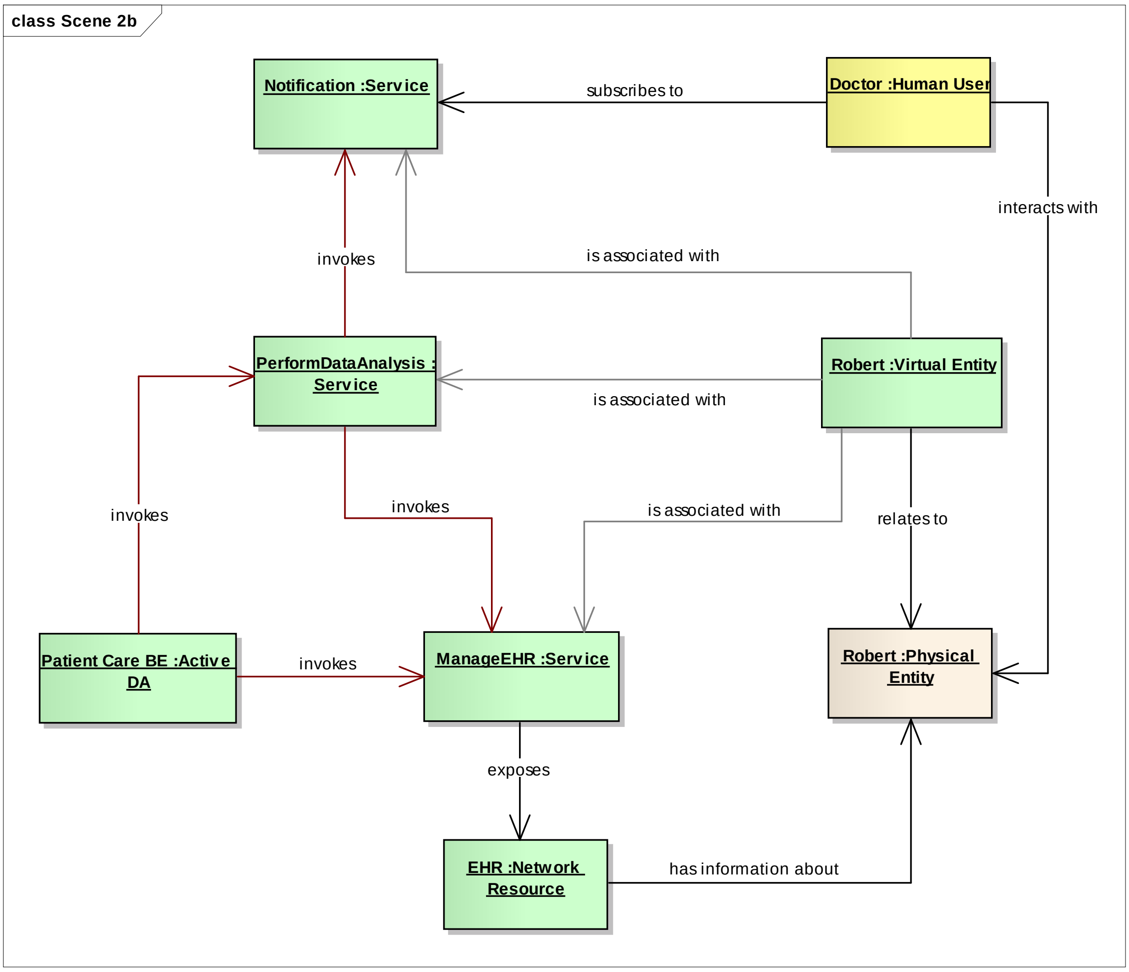

Figure 6: Class instance diagram of Scene 2: data analysis

Once all measurements are taken, the data must be archived and an analysis performed.
The data is stored in the Electronic Health Record, seen in the figure as a *Network Resource*.
This *Network Resource* is exposed by the EHR *Service*.
Robert's *Virtual Entity* is associated with this service, so that the backend application Patient Care BE can perform a lookup and find the service to update the *Service* providing access to the EHR of Robert[2].

[2] Note that another modeling possibility could be to introduce the EHR as a *Virtual Entity* associated to the *Physical Entity* Robert. We have chosen though for simplicity's sake to have only one *Virtual Entity* associated to Robert. In some cases though, this is not possible. For example, in reality, the EHR could be managed by the National Health Service, while the hospital could use another (i.e. local) Virtual Entity as a digital representation of Robert.

Once the data is archived, the Patient Care BE will invoke the *Service* Data Analysis, which will fetch the data in the EHR of Robert, check if the measurements did not cross bounderies and in case they did, invoke the Notification *Service*. This *Service* is again associated with the *Virtual Entity* of Robert.
The Doctor, a *Human User* has subscribed to the Notification *Service* and receives notifications sent by the Data Analysis *Service*.
After receiving the notification, the Doctor will typically access the EHR to consult the patient's data and to decide if some actions are required. This step in the use case is not represented in the class instance diagram above.

3.3.3.3 Implementation details

The following table contains software, hardware and components which are planned to be used for the implementation of this scene. Details on the individual items may be seen in chapter 6.

Applications	Devices	Functional Components
SW.ALUBE.1 (Patient Care BE App)	HW.ALUBE.1 (IoT-Phone)	C.ALUBE.1 (Virtual-entity Resolution)
SW.ALUBE.2 (Patient Care Mobile App)	HW.ALUBE.4 (A&D Blood pressure meter)	C.ALUBE.2 (IoT Service Resolution)
SW.ALUBE3 (Data Analysis App)	HW.ALUBE.5 (A&D Weight scale)	
SW.ALUBE.4 (Notification App)	HW.ALUBE.6 (Entra glucose meter)	
SW.ALUBE.5 (EHR App)		

Table 3: Applications, devices and functional components planned for scene 2 implementation

3.3.4 Scene 3 – Remote Patient Care: insulin alarm (CFR)

As in the previous scene, this scene aims at demonstrating how IoT technologies can be used to assist people in their medical routine, and to minimize the need and the amount of hospital stays. In addition, this scene enhances the patient's self-awareness about their health condition and how the system can help them by giving advices and providing remote assistance. This is particularly true for elderly patients which may start suffering from early case of dementia.

In the third scene, Jane is reminded about her periodic blood glucose level measurement by an automatic alarm bell played by her IoT-Phone.
Jane performs the measurement and the data is instantly sent to her via her IoT-Phone. Finally, she can update her Electronic Health Record (EHR) providing information about the activity she was performing at the time of the measurement. The backend system stores her data in her historical database and analyzes them: if an abnormal condition is reported, both Jane and her doctor are notified in order to provide a prompt solution for the condition.

Scene summary:

- Location: this scene is at Robert's house
- Covered topics:
  - Home Care
  - Remote health assistance
  - Remote health monitoring
  - Smart Objects
- Key benefits:
  This scene highlights the advantages provided by smart device interoperability (measurement tool + smart phone) and the automatic analysis performed by the backend.

3.3.4.1 Description from a User's Perspective

Jane, being a diabetic, must measure her blood glucose level three times a day. The system reminds her to do so with an alarm. IoT smart devices, in this case Jane's IoT-Phone, allow users to easily interact with complex services and data structure (Jane's EHR). The IoT information model enables different technologies to interact on the same data.

Jane acknowledges the alarm, measures her blood glucose level and indicates the activity she was doing. The system calculates the amount of insulin she must inject. Jane injects the insulin and her record is updated. All the actions are simplified by the automatic interaction of the different IoT resources, such as the Blood Glucose reader, Jane's IoT-Phone, Jane's EHR and the hospital backend system. In this case an insulin control service interacts with Jane's insulin level record resource provided in her EHR to provide both Jane's and her physicians' with frontends for a quick translation of raw data in usable information. In addition Jane can interact with her Health Care frontend in order to update her EHR.

All measurements are stored in the system and an automatic analysis of the results is performed, notifying her doctor in case some values are outside the normal range. IoT technology can offer services to physicians to automatically monitor their patient conditions and to rise an alarm when critical conditions are detected.

3.3.4.2 Relation to the Domain Model

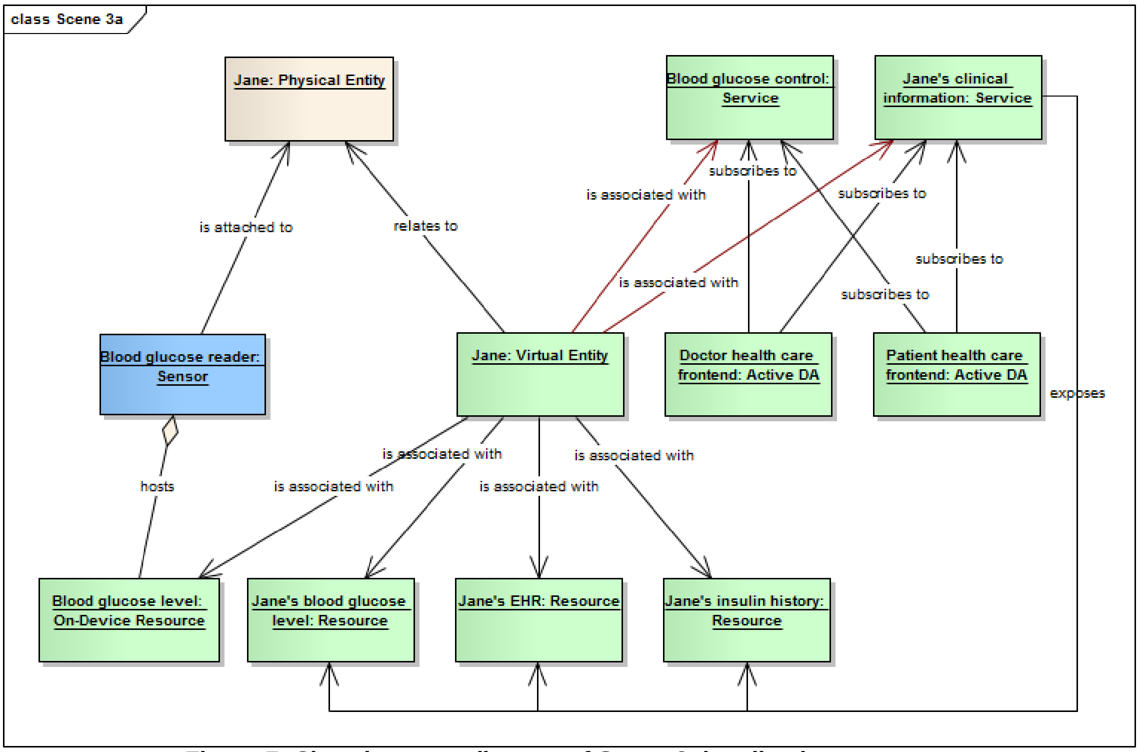

Figure 7: Class instance diagram of Scene 3: insulin alarm resources

The modeling of the different devices used in scene 3 for taking the blood glucose level and let the different actors (Jane and physicians) interact with the system and with one another is pretty simple. The patient, Jane, is modeled both as a *Physical Entity* with a corresponding *Virtual Entity* in the digital domain.
The blood glucose level monitoring device is modeled as a *Sensor*. The two IoT-Phone devices host an *Active Digital Artefact* each representing the two frontends to the backend system. In

this simplified model the frontends subscribe to two *Services*: the blood glucose control and Jane's clinical information.
The blood glucose reader *Sensor* monitors Jane's *Physical Entity* and hosts the related Blood Glucose level *Resource* to which, the blood glucose control *Services* is subscribed to.
Jane's EHR, blood glucose level and insulin history are represented as *resources* which have information about Jane's *Virtual Entity* and are exposed by Jane's clinical information *Service*.
The second picture illustrates the interactions between users and services. In particular, the diagram models both Jane and her doctor has *Human Users* who interact with Jane *Physical Entity* and subscribe to the two aforementioned *Services.*

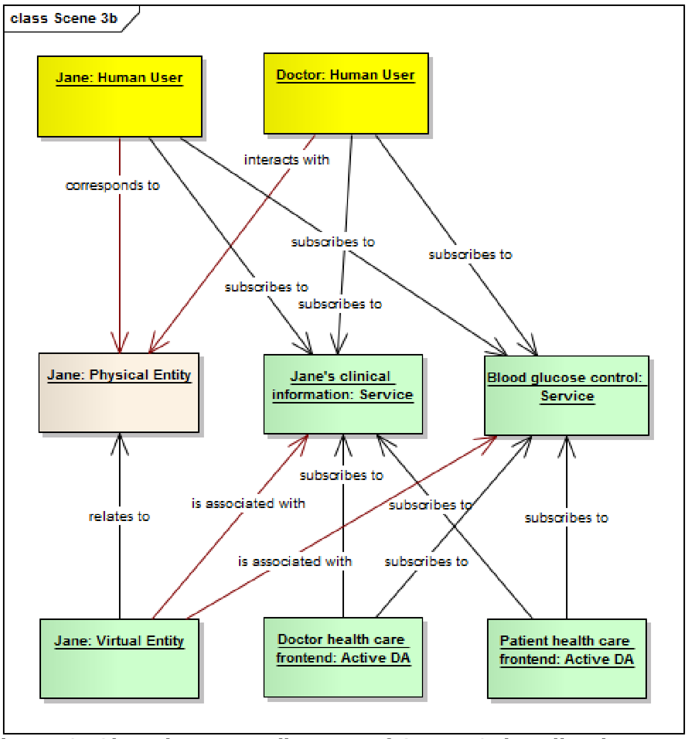

Figure 8: Class instance diagram of Scene 3: insulin alarm users

3.3.4.3 Implementation details

The following table contains software, hardware and components which are planned to be used for the implementation of this scene. Details on the individual items may be seen in chapter 6.

Applications	Devices	Functional Components
SW.CFR.1 (backend software)	HW.CFR.1 (blood glucose reader)	C.CFR.1 (Communication Unification)
SW.CFR.2 (Jane IoT-Phone application)	HW.CFR.2 (gateway)	C.CFR.2 (Device Interoperability)
SW.CFR.3 (Physicain IoT-Phone application)	HW.CFR.3 (backend server)	C.CFR.3 (Communication Reliability)
SW.CFR.4 (sensor nodes firmware)	HW.CFR.4 (patient and physician mobile)	
SW.CFR.5 (gateway device firmware)		

Table 4: Applications, devices and functional components planned for scene 3 implementation

3.3.5 Scene 4 – Low Insulin Supply (link to Retail UC) (ALU-BE)

This scene illustrates the link between the Health and the Retail use case. The medicine supply of the patients is continuously monitored and in case the supply has lowered below the threshold, automatic actions can be triggered to assure that the patient is never out of medicines.

Scene summary:

- Location: this scene originates at Robert's house; the event is forwarded to the IoT-Phone of Salomée.
- Covered topics:
  - Home Care
  - Remote health assistance
  - Smart Objects
  - Link to Retail use case
- Key benefits:
  With IoT-A technology, it is easy to have cross application domain use cases, something that is extremely difficult to obtain with a traditional approach.

3.3.5.1 Description from a User's Perspective

Each ampoule of insulin is tagged with a RFID tag. The ampoules are stored in a medicine cupboard which is also equipped with an RFID reader.

Robert takes the last ampoule of insulin out of his medicine cupboard. The system detects that the ampoule is removed and will update Robert's EHR.

In case the EHR shows that this was the last ampoule of insulin and there are no more ampoules present in the cupboard, a notification to the IoT-Phone of Salomée is sent so she can stop by the pharmacy and buy some more insulin.

3.3.5.2 Relation to the Domain Model

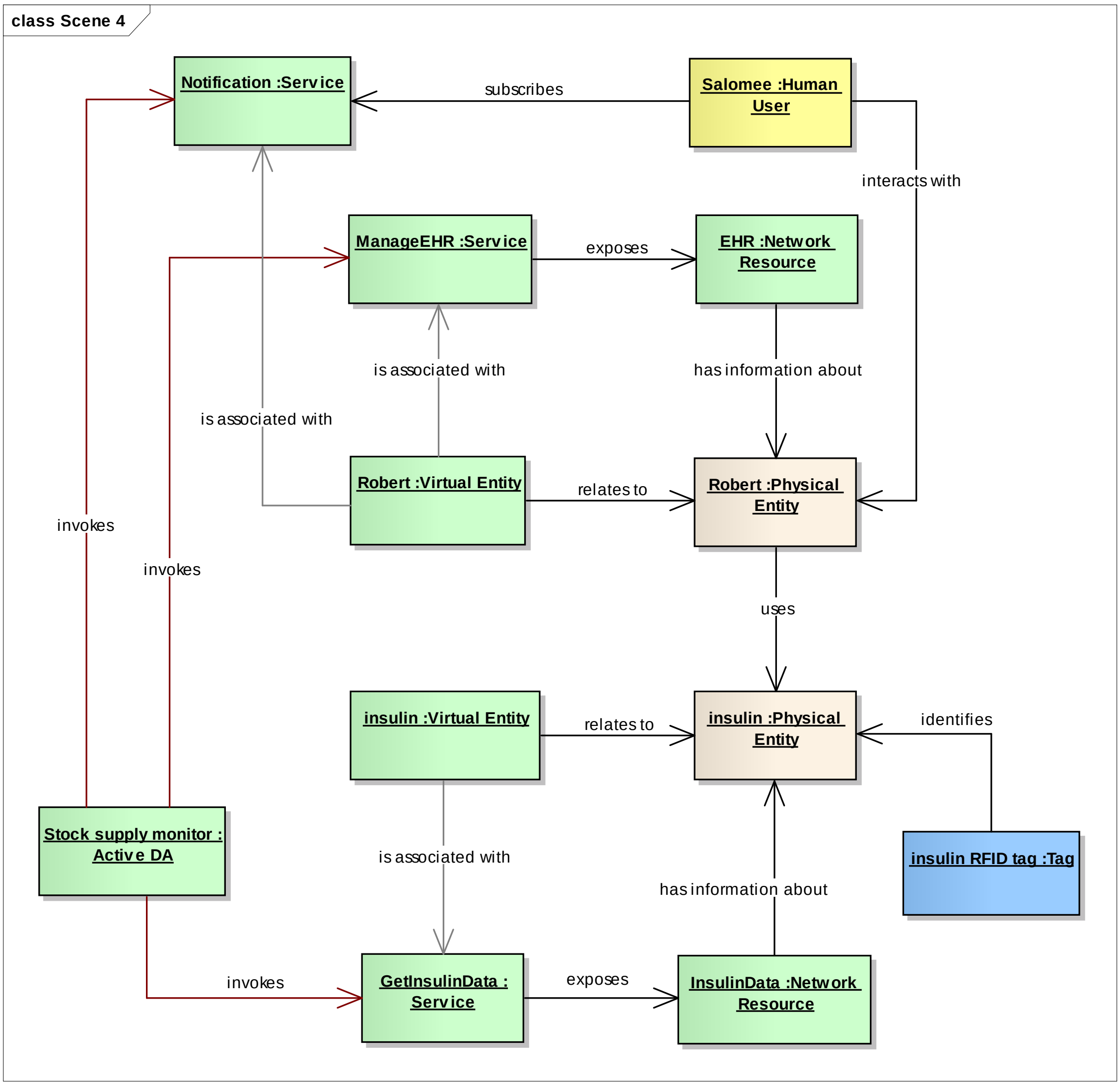

Figure 9: Class instance diagram of Scene 4: low insulin supply

Each ampoule of insulin is tagged with an RFID *Tag*, that identifies the *Physical Entity* of the ampoule. The insulin is represented in the digital domain with the insulin *Virtual Entity*.
The information about the insulin is stored in the *Network Resource* InsulinData, that is accessed via the GetInsulinData *Service*.
The Stock supply monitor *Active Digital Artefact* invokes the GetInsulinData service to obtain more information on this Virtual Entity.
The Stock supply monitor *Active DA* will as well update the EHR of Robert in case a box of insulin is taken out of the medicine cupboard (not modeled).
The EHR is modeled as a *Network Resource* which is accessible via the EHR *Service*.
Part of the EHR is the amount of insulin that Robert still has available.
When the Stock supply monitor *Active DA* reads from the EHR that the amount of insulin has dropped below the threshold, the Notification *service* associated to Robert's *Virtual Entity* will be invoked. Salomée, modeled as a *Human User*, is subscribed to this service and will receive the notification that she needs to go for more insulin.

3.3.5.3 Implementation details

The following table contains software, hardware and components which are planned to be used for the implementation of this scene. Details on the individual items may be seen in chapter 6.

Applications	Devices	Functional Components
SW.ALUBE.4 (Notification App)	HW.ALUBE.1 (IoT-Phone)	C.ALUBE.1 (Virtual-entity Resolution)
SW.ALUBE.5 (EHR App)	HW.ALUBE.8 (RFID reader)	C.ALUBE.2 (IoT Service Resolution)
SW.ALUBE.6 (Stock monitor App)		
SW.ALUBE.7 (RFID DB App)		
SW.ALUBE.8 (Stock alarm Mobile App)		

Table 5: Applications, devices and functional components planned for scene 4 implementation

3.3.6 Scene 5 – Remote Patient Panic Event (ALU-BE)

Some patients subscribed to the Remote Patient Care plan are equipped with a medical emergency response system. This system is recommended for elderly patients, physically disabled or mentally disabled persons or person living alone that want to have the comfort of calling for medical care by just pressing a button.
In this scene, it is illustrated how the medical situation is assessed after receiving a panic button event and close by relatives are contacted to look for the patient and see if emergency medical care is needed.

Scene summary:

- Location: at Robert's house
- Covered topics:
  - Home Care
  - Remote health assistance
  - Remote health monitoring
  - Smart Objects
  - Emergency event handling
- Key benefits:
  With the IoT-A technology shown in this scene, a very flexible notification system can be set up that can use any device or service integrated, even if these services are not part of the application domain.

3.3.6.1 Description form a User's Perspective

Robert, being a diabetic, always runs the risk of hypoglycemia so he is wearing a bracelet with a fall detector, a pulse monitor and a panic button.

One morning, Robert suddenly starts to feel dizzy and lightheaded. He presses the panic button of his bracelet.

After receiving the panic event, the system checks the EHR entry for Robert and finds that Jane, Robert's mother, is first on the list of persons to contact in case of a panic event so the system contacts Jane on her personal IoT-Phone and informs her of the panic button pressed.

Jane is close by so she looks for Robert and sees he is already doing better after he ate the candy bar he is always carrying with him.

3.3.6.2 Relation to the Domain Model

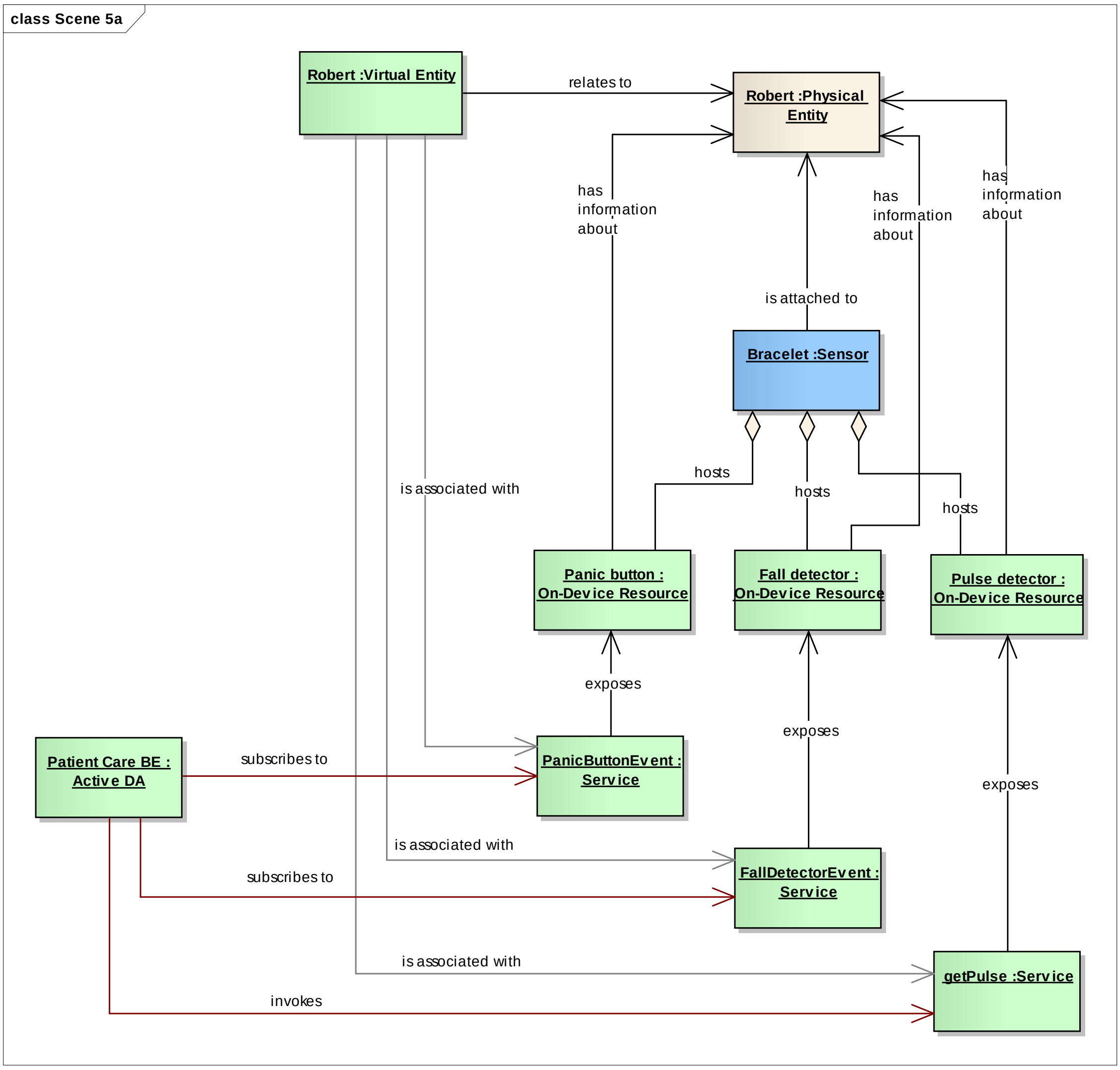

Figure 10: Class instance diagram of Scene 5: panic event

The modeling of this scene is quite straightforward and is basically three times the same. The bracelet, which is a *Device* hosting 3 *Sensors*, contains three *On-Device Resources*: the panic button, the fall detector and the pulse detector. Each of these *On-Device Resources* is exposed by a *Service*.
The backend application Patient Care BE, modeled as *Active Digital Artefact*, is subscribed to the events of these three services.

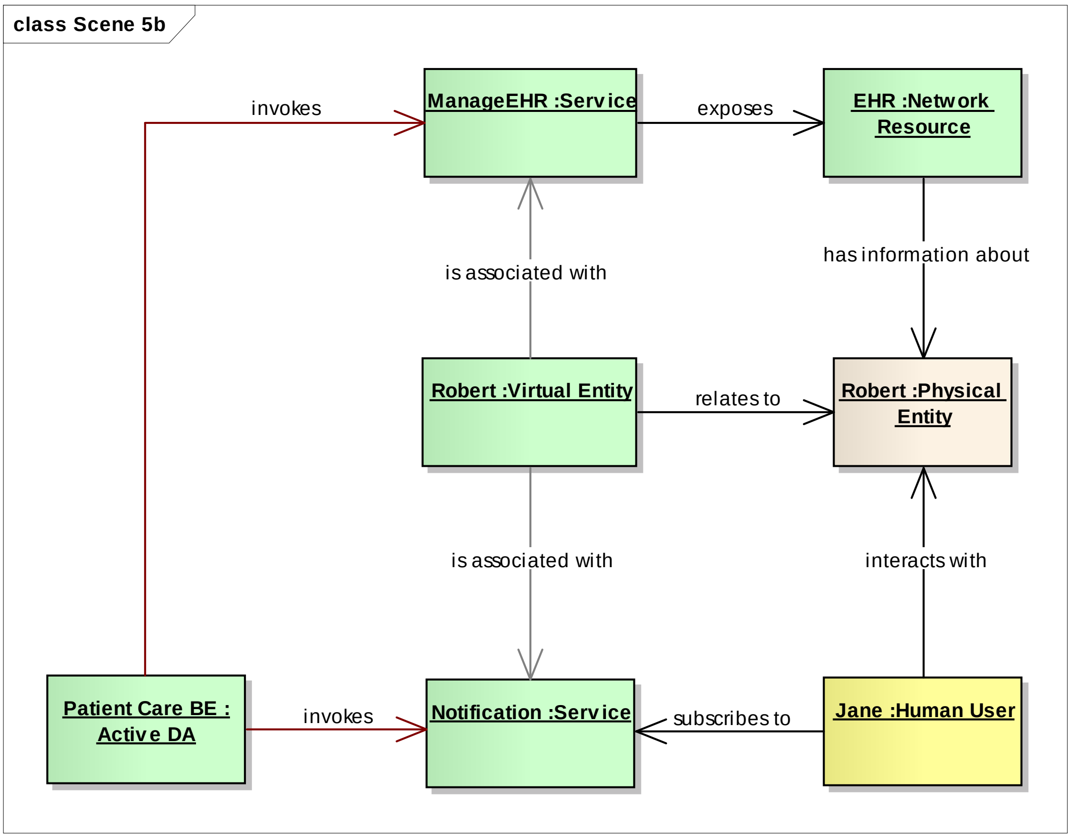

Figure 11: Class instance diagram of Scene 5: panic event

For the event handling, the EHR of Robert, modeled as a *Network Resource* and exposed by a *Service* is consulted. The first to contact in the list in case of emergency is part of the EHR. The Patient Care BE will then invoke a Notification *Service*, to which Jane, of type Human *User* is also subscribed.

3.3.6.3 Implementation details

The following table contains software, hardware and components which are planned to be used for the implementation of this scene. Details on the individual items may be seen in chapter 6.

Applications	Devices	Functional Components
SW.ALUBE.1 (Patient Care BE App)	HW.ALUBE.1 (IoT-Phone)	C.ALUBE.1 (Virtual-entity Resolution)
SW.ALUBE.2 (Patient Care Mobile App)	HW.ALUBE.7 (Emergency bracelet)	C.ALUBE.2 (IoT Service Resolution)
SW.ALUBE.4 (Notification App)		
SW.ALUBE.5 (EHR App)		

Table 6: Applications, devices and functional components planned for scene 5 implementation

3.3.7 Scene 6 – Accident and hospitalization: car accident (CFR)

This scene illustrates a sudden critical condition, such as a car accident. In particular, this scene addresses the interactions among sensors hosted on Robert's body network and those hosted on Robert's IoT-Phone. Automatic services process the data in order to identify actual dangerous conditions from simple anomalies on the sensor readings.

Scene summary:

- Location: on the street, Robert driving his car
- Covered topics:
  - Care on the move
  - Remote health monitoring
  - Smart Objects
  - Emergency event handling
- Key benefits:
  This scene is intended to show the reactiveness of the system in case of sudden anormal conditions, the capabilities of recognizing dangerous event from outliers and the promptness in providing assistance to the accident victim. In addition, this scene requires the interaction of most of the different technologies in IoT systems: IoT-Phone accelerometers are used to monitor Robert's movement, while Robert's body sensor network provides readings for his baseline vital parameters, such as the breath rate and the heart beat rate. Finally *Active Digital Artefacts* are hosted both in Robert's mobile and in the hospital backend system in order to quickly connect the emergency room staff with accident victims.

3.3.7.1 Description from a User's Perspective

Robert is wearing body sensors that continuously monitor vital signs and that can detect life threatening situations. He is also wearing his bracelet with panic button. Multiple IoT smart devices interoperate in order to provide a combined analysis of Robert's vital parameters. In such a way, it is easier to recognize false alarms from emergency situations. *Active Digital Artefacts* can subscribe to patient vital parameter *Services* and dispatch alarms to Robert's physician and/or the E.R. staff. In particular, the IoT-Phone accelerometers are used to detect abrupt Robert's movement and may trigger emergency alarms; also, alarm are trigger only if the emergency situation is verified by the other wearable sensors and Robert does not confirm he is safe by pressing a button sequence on the smartphone within a given time. For instance, during the car accident, the IoT-Phone is detecting the car hitting something as a sudden and strong acceleration (well beyond Robert's everyday actions); also, the body sensors will detect Robert's panic through an increasing breath rate and heart beat rate due to the increasing adrenalin level.
His body sensors detect a life threatening situation and immediately dispatch emergency medical care. In addition to periodic monitoring, wearable sensors can be programmed to rise alarm on critical condition. Alarms can be processed by on-device services to identify situation where immediate assistance is needed.

3.3.7.2 Relation to the Domain Model

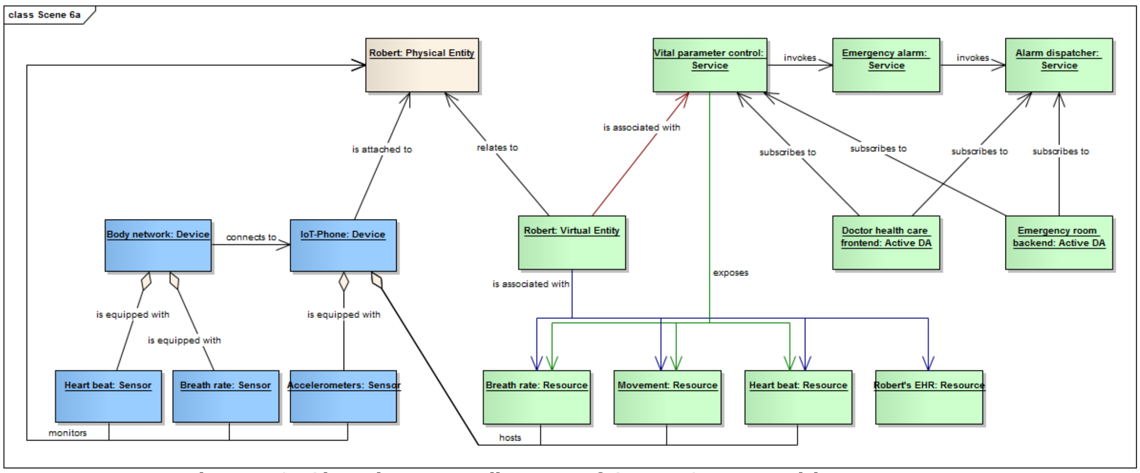

Figure 12: Class instance diagram of Scene 6: car accident resources

In this scene Robert is represented as the usual *Physical* and *Virtual Entities* couple. Robert's *PE* is being monitored by accelerometers (*Sensor*) hosted on his IoT-Phone (*Device*) and by other *Sensors* such as breath rate reader and heart beat monitor hosted on Robert's body network (*Device*). In addition, three *Services* are modeled: the vital parameter control is in charge of exposing the main *Resources* and periodically monitoring their values, the emergency alarm create the alarm messages, which are then sent by the alarm dispatcher. Two *Active Digital Artefacts* model the doctor and the emergency room frontends. Finally, in the second figure, the doctor is modeled as a *Human User*, who interacts with Robert's *Physical Entity* and subscribes to the vital parameter control and the alarm dispatcher *Services*.

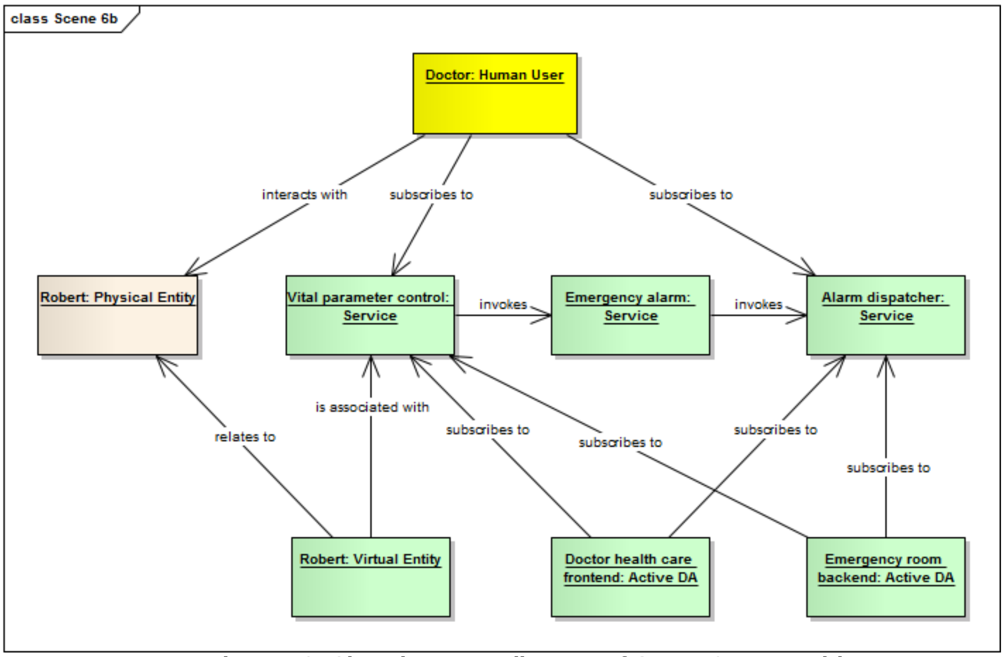

Figure 13: Class instance diagram of Scene 6: car accident users

3.3.7.3 Implementation details

The following table contains software, hardware and components which are planned to be used for the implementation of this scene. Details on the individual items may be seen in chapter 6. In order to demonstrate this scene, the changes in the vital readings will be simulated by adding synthetic signals to the actual sensor readings. This will make it possible to observe the system reactions in a safe environment.

Applications	**Devices**	**Functional Components**
SW.CFR.1 (backend software)	HW.CFR.1 (Robert's body network)	C.CFR.1 (Communication Unification)
SW.CFR.2 (Robert's IoT-Phone application)	HW.CFR.2 (gateway)	C.CFR.2 (Device Interoperability)
SW.CFR.3 (Physic /ER Staff IoT-Phone application)	HW.CFR.3 (backend server)	C.CFR.3 (Communication Reliability)
SW.CFR.4 (sensor nodes firmware)	HW.CFR.4 (Robert's and physician mobile)	
SW.CFR.5 (gateway device firmware)		

Table 7: Applications, devices and functional components planned for scene 6 implementation

3.3.8 Scene 7 – Expedited Checking Into a Hospital (HSG)

This scene shows how ubiquitous sensors in consumer goods - such as the IoT-Mouse - when paired with the Internet of Things Architecture, could speed up hospital check-in and make initially acquired information available in later applications. As a frame of reference, a hospital in a medium-sized European city has 100000 check-ins per year. The time saved can be a big cost and productivity improvement, and data accuracy is improved over manual entry.

In addition to fast hospital check-in, the application saves personnel time & improve data accuracy vs. manual entry. In particular, unlike many scenes which focus on machine-to-machine interaction, the scene shows what a direct human-to-machine interaction with IoT-A would be like. As such, the overlap between the existing internet and the future internet of things is shown.

Scene Summary:

- Location: Hospital
- Covered topics:
    - Hospital quick check-in enabled by IoT-A
    - Integrated information capture
    - Improved data accuracy
- Key Benefits:
    - IoT-A's resolution services are applied to enable fast hospital check-in, save personnel time & improve data accuracy, vs. manual entry

3.3.8.1 Description from a User's Perspective

After the accident, Robert is taken to a hospital. Although Robert is unconscious, the staff finds his Citizenship ID. They do not find any health Identification.

The clerk scans them with an integrated IoT Mouse & NFC reader, which calls up the Internet of Things Architecture Resolution service. The service is called to identify Robert, and request further hospital-relevant information about him from other data sources.

3.3.8.2 Domain Model of Scene

In this section, we show the interaction diagrams and the class diagrams of the Hospital check-in scene.

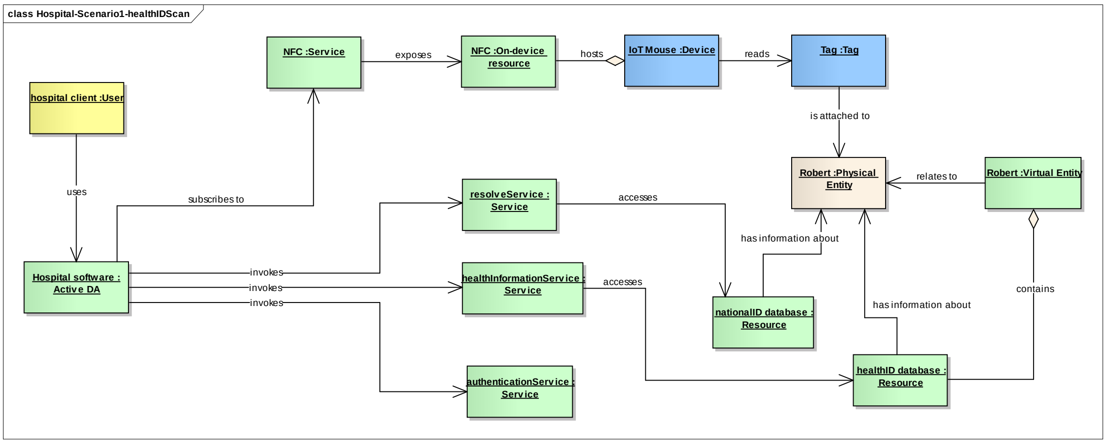

Figure 14: Class instance diagram of Scene 7: hospital check-in

The figure shows the different implemented classes and their relationships for the given use case in order to achieve the desired functionality. Each class shown caters to a particular function in the overall picture. For e.g. class *local hospital software* is the main entry class which controls the GUI and initiates different services. The labels on top of the arrows depict what kind of relationship exists between two classes.

Hospital user client uses the *Hospital software*. The *Hospital software* subscribes to *NFC service*. The service listens to the *NFC tags* in the vicinity and communicates with them. These *NFC tags* are read using the *IoT mouse* which is operated by the *hospital user client*. NFC tag reader is integrated in the *IoT mouse*. NFC tags are attached to the patients (Robert in our case) and contain the healthcare information of the patients.

After a NFC tag is read, *hospital software* calls the *authenticationService* to get access to the database where information about the patient is stored. Once the client is granted the permission to access the database, a *resolveService* is executed to resolve the health ID (part of healthcare information read from the NFC tag) into a unique ID of the patient.

The unique ID leads to fetch more information about the patient from different connected/distributed databases (e.g. driver license database, health database etc.). Finally the *healthInformationService* is called to get all the healthcare information (address, information of the last visit, prescriptions, doctor details etc.). The health EHR has all the relevant information about the patients.

After getting the relevant information from the health database, the *hospital software* displays it in the hospital GUI form.

3.3.8.3 Implementation Details

The following table contains software, hardware and components which are planned to be used for the implementation of this scene. Details on the individual items may be seen in chapter 6.

Software	Hardware	IoT-A components
SW.HSG.1 (Local Hospital Software)	HW.HSG.1 (IoT-Mouse)	C.ALUBE.1 (Virtual-entity Resolution)
	HW.HSG.2 (Local Computer)	C.ALUBE.2 (IoT Service Resolution)
	HW.HSG.3 (Tagged ID Cards)	C.HSG.1 (AuthN)
		C.HSG.2 (AuthS)

Table 8: Applications, devices and functional components planned for scene 7 implementation

3.3.9 Scene 8 – Environment and Patient Remote Monitoring (CATTID)

During the stay of the patient at the hospital, the monitoring of the patient is one of the important tasks. The hospital in this scene is equipped with a Hospital Information System (HIS). HIS is software at the hospital which monitors the environmental conditions in the room and the medical conditions of patients' through sensors. Upon an issue is detected by HIS, defined controls are done by the system automatically and responsible person is notified. HIS also has a mobile version which runs on the hospital personals' tablets. Communication between HIS and the hospital staff is established through those tablets. One of the key benefits of the HIS which is described in this scene is that IoT-A enables remote monitoring of the patients' and the environmental conditions through sensors. The routine temperature control of the patient is performed remotely and upon the measurement of high patient body temperature, the HIS starts the defined examination of the environmental features to explore the reasons of high body temperature. Interoperability between the sensors and IoT-A services enables the monitoring of the patient and the sudden recognition of a possible problem. Once the problem is recognized services communicate to diagnose the problem.

Scene summary:

- Location: This scene is at a HIS equipped hospital room
- Covered topics:
    - Remote environment monitoring
    - Remote health monitoring
    - Smart Objects
- Key benefits:
  The motivation of the scene is to use the IoT-A services with smart objects. Patient and environment equipped with sensors enable the Hospital Information System to use IoT-A Services to control the conditions remotely, figure out the problem and inform the hospital staff on the performed analysis.

3.3.9.1 Description from a User's Perspective

Robert is hosted in a twin room with another patient. The room is equipped with sensors to measure the humidity and temperature of the room. When Robert is under care, his body is equipped with body sensors to monitor his body temperature. Although the responsible nurse controls the patient's condition in person periodically, those sensors enable the patient being under a continuous monitoring.

The nurse defines the routine temperature monitoring parameters of Robert such as the time, the frequency, temperature limits and she confirms the data through the HIS.

As defined by the nurse, in the morning, before the nurse starts her visit to the patients, the body temperature measurement is done by the sensors. Unfortunately, the measurement value is a little bit higher than the defined upper threshold. This fewer indication is recognized by HIS and system automatically starts to do defined controls of the possible causes. The control of the room temperature shows that the room temperature is lower than the defined value. The responsible hospital staff is informed by HIS about the possible failure in the heating system. The responsible person detects the failure and repairs it. After the repairs have been acknowledged, HIS adjusts the room temperature to the defined degree. The nurse is notified that Robert's body temperature was low during the morning routine and it was due to the failure in the heating system but nothing vital.

3.3.9.2 Relation to the Domain Model

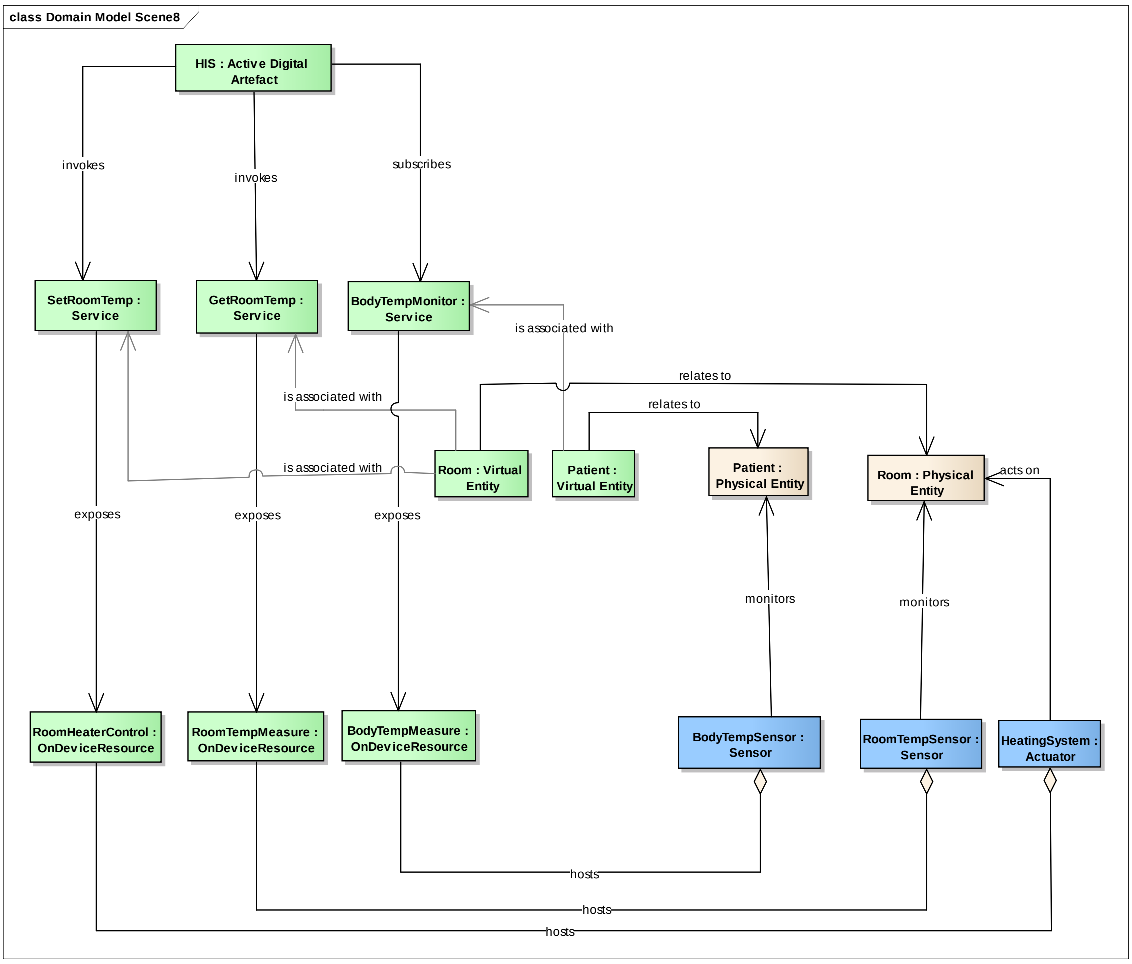

Figure 15: Class instance diagram of Scene 8: environment and patient remote monitoring

In the domain modeling of the scene Room and Patient are modeled as *Physical Entity* and *Virtual Entities* are related with them. For the temperature measuring there are *Sensors* for both body of the patient and the room. Resources are modeled as *OnDeviceResources* and there are 3 services which expose them accordingly. HIS (Hospital Information System) is modeled as *DigitalArtefact* which is the user of the system.

BodyTempMonitor *Service* exposes the BodyTemp *OnDeviceResource* to do the reading of the body temperature sensor value. This service is subscribed by the HIS and by subscription the HIS is notified on every changes of the value.

GetRoomTemp is a *Service* which is invoked by HIS in case of examination of the environmental features is needed. Service exposes RoomTemp *OnDeviceResource* to do the reading of the room sensor.

SetRoomTemp is a *Service* which is invoked by HIS. It exposes RoomHeater *OnDeviceResource* which hosts the actuator HeatingSystem. The *Actuator* changes the room temperature to the defined settings after the fix of the problem in the heating system.

3.3.9.3 Implementation details

The following table contains software, hardware and components which are planned to be used for the implementation of this scene.

Applications	Devices	Functional Components
SW.CATTID.1 (HIS)	HW.CATTID.1 (Computer)	C.CATTID.1 (Application)
SW.CATTID.2 (Program on Sensor Node)	HW.CATTID.2 (Sensor)	
SW.CATTID.3 (Program on Sensor Node)	HW.CATTID.3 (Sensor)	

Table 9: Applications, devices and functional components planned for scene 8 implementation

3.3.10 Scene 9 – Medication Control (CATTID)

During their stay in the hospital, patients are administered medication whose types and doses are decided by their doctor. These planned medications are forwarded to the pharmacy of the hospital, and the responsible nurse does the medication to every patient.

In this scene, the medication type and dose is registered to the Electronic Health Record (EHR) of the patient. At the check-in the NFC patient bracelet is attached to the patient and after the reading of the NFC bracelet with a tablet application the related data is accessed by the user. Every medicine ampoule has an RFID tag. The dosage and additional information about the medicine in the database is accessed after the reading of the ampoule's RFID tag. The nurse does the reading of both the patient's bracelet and tag of the ampoule before administering the medicine. After readings are submitted to the EHR of the patient, automatic data analysis will take place and if needed, an alarm rises.

Scene summary:

- Location: this scene is at hospital
- Covered topics:
    - Detection of possible undesirable effects of medications
    - Smart Objects
- Key benefits:
  The responsibility of the hospital staff is decreased and any wrong type or dosage of medication is prevented. The mobile application and smart objects are used to access and control the data at the EHR.

3.3.10.1 Description from a User's Perspective

The doctor decided that Robert needs to be medicated 2 times a day, once in the evening and once in the morning. This data and the related data of the medicine (like dosage) are registered to the EHR of Robert.
For the evening medication the medicine within the tagged ampoule arrives from the hospital's pharmacy. The nurse takes her tablet with the installed software to read both the NFC bracelet of the patient and the RFID tag of the ampoule.
When she reads the bracelet of Robert with the tablet application she has access to his EHR information. For the medication she must read the RFID tag of the ampoule. Upon the read with the tablet, an automatic data analysis is triggered. An alarm is raised on the screen. The defined dosage of the medicine is not compliant with the defined dosage in the EHR of Robert. The

alarm is cancelled by the nurse. She does not administer the medicine and contacts with the pharmacy to inform them about the wrong dosage.

3.3.10.2 Relation to the Domain Model

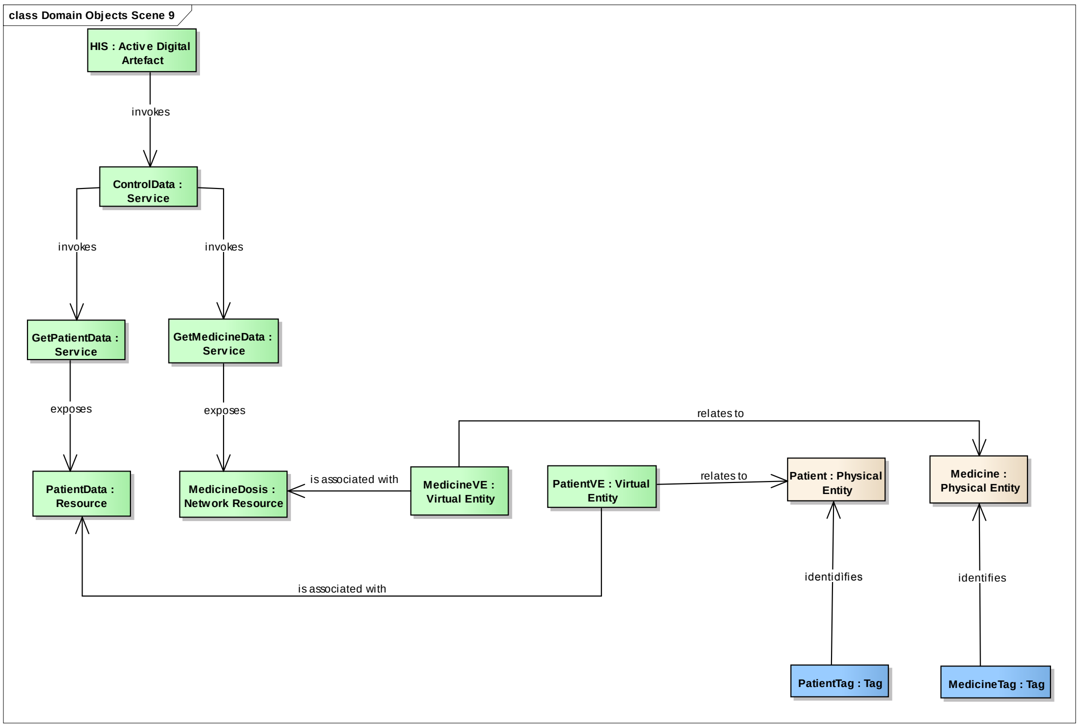

Figure 16: Class instance diagram of Scene 9: medication control

The patient and the medicine are modeled as *Physical Entities* and the *Virtual Entities* are related. The *Tag* modeling of NFC bracelet of the patient and the RFID tag of the ampoule are straightforward.
GetMedicineData is a *Service* which exposes the *NetworkResource* modeled MedicineDosis which is associated with the *Virtual Entity* MedicineVE.
GetPatientData is a *Service* which exposes *Resource* modeled PatientData which is associated with the *Virtual Entity* PatientVE.
ControlData is a *Service* which invokes two other services, GetPatientData and GetMedicineData. ControlData is invoked by the *DigitalArtefact* modelled user HIS.

3.3.10.3 Implementation details

The following table contains software, hardware and components which are planned to be used for the implementation of this scene.

Applications	Devices	Functional Components
SW.CATTID.4 (HIS Tablet Application)	HW.CATTID.4 (Tablet)	C.CATTID.2 (Application)
	HW.CATTID.5 (RFID Tag)	C.CATTID.3 (Device connectivity and communication)
	HW.CATTID.6 (Gateway)	

Table 10: Applications, devices and functional components planned for scene 9 implementation

4. Use Case II: Retail

4.1 Defined Storyline

Nearly every single one of us constantly has to go shopping and collects experiences with the retail industry; therefore it is a part of our daily life. As technological innovations permeate other parts of our lives, retailing is also getting more and more penetrated by different technologies, supporting and helping us in many different situations, for example, smart mobile phones equipped with shopping applications managing our shopping lists or our dietary information. This leads the retailers to a new challenge to integrate their business into the consumer's world and vice versa.

On the one side, information which is generated by the customers (e.g. product ratings) might be of high interest to the retailers, especially the product and category managers. On the other side, information which is owned by a retailer is not always just of interest for the retailers themselves, like looking at the traceability of individual goods, real-time queries on a mobile device for the customer or the availability of products in a certain store.

To realize traceability of individual goods during the whole product life cycle and to create transparency all along the supply chain, everything starts with the serialization of each single item, which is happening at some point in time during the manufacturing process. Adding additional sensors to the items to collect various environmental information as well as counting the emitted carbon by the products itself helps increase transparency.
Privacy concerns which arise from tracking the items of customers outside of the store with the help of tagged items may be addressed by solutions that help both, the supply chain and the customers.

Other examples of serialized objects include an NFC tag to a laptop to track the ownership and add built-in accelerometers to it for recalling physical transport damages. Many kinds of different data is recorded on the way from the Far East to the point of destination and is transmitted during the transport or on the handover and could be made visible by the system at any time.

Using smart mobile phones in combination with RFID- or NFC-tagged products does not only provide advantages for manufacturers, retailers, and customers but also for delivery and anyone involved in logistic processes concerning these products. With this scenario in mind, the future internet of things applied to retail could unfold as follows.

Ted, the delivery man for the gardener, uses his IoT-Phone to manage transport orders, scan tagged items or load carriers and receive status messages from sensors added to the items he is currently moving. This way, he can know about the circumstances of the products without the need to do a visual inspection (i.e. stop his truck, which would mean a delay).

When arriving at the local supermarket, Ted lets the load carriers he delivered pass an RFID gate which automatically recognizes them. After briefly talking to John, the store manager, Ted sends the sensor record history saved on his IoT-Phone to John's IoT-Phone via NFC. The manager can now see that on the way to the store, there was a critical rise in temperature at one point, which causes him to visually inspect the orchids and decide if he still wants to accept the delivery. Since John identifies the orchids as fine, he sends a message of approval to Ted's IoT-Phone.

For taking a look on the customer's perspective of our scenario, we switch to Salomée, a young woman representing a customer.
This Saturday, Salomée decides to try out the new supermarket (where John is the store manager) that opened recently. She is a young attorney at law and a single mother. Salomée just started her first position at a big law firm, and therefore has to put in long hours, leaving her son in day care. Balancing work and family time is difficult. Therefore, she usually does all of

her shopping on Saturday morning even though she hates the long queues that usually form then. As a single parent she is very price-conscious, but she still wants her child to get healthy nutrition and she also cares about the environment, preferring local products with a small carbon footprint.
As she enters, she is positively surprised by its spaciousness and its calm atmosphere. Salomée has a shopping application installed on her IoT-Phone that allows her to receive information about products upon scanning them or when the store's backend system recognizes a certain behavior or circumstances. The software also keeps track of Salomée's shopping behavior in order to provide more personalized and thus more efficient suggestions.

Today, Salomée is looking for cheese, so she enters the refrigerated section. Once she found a packet of cheese that caught her interest, she reads its NFC tag to have her virtual Shopping Assistant give her additional information on it and compare it to other kinds of cheese she has bought before. With the support of the application she is able to quickly find the cheese she wants to buy.

Now that Salomée has found a cheese she likes, she wants to buy her favorite wine. The Shopping Assistant on her IoT-Phone can now tell her about the prices of wines she has bought before and if one of these wines is out of stock, a recommendation for a similar wine is shown.

Today Salomée has to acknowledge that there are no affordable wines available that she likes so she starts leaving. This and the fact that she has got cheese in her shopping cart causes a big TFT display to show an announcement for a 30% discount of wine for buyers of cheese. Salomée is glad to return and buy a bottle of wine she had considered too expensive earlier.

In parallel, John and his crew have to struggle with the always busy Saturdays. They need to replenish the empty shelves and need to know what the customers need next. Cameras on the ceiling and other ways of understanding the customers support them to be more efficient and provide the best services to the customer.

Some automatic processes simplify the staff's tasks. The orchids have sensors attached to them that monitor environmental features critical to the quality of the flowers and send this information to the price tags to enable automatic adjusting of prices according to product quality. Since the air condition in the store is currently not correctly set up, the orchids' price is lowered by 10% due to a rise in temperature. Continuing her shopping, Salomée passes the orchids and recognizes the beauty of these and to her surprise, realizes there is discount on them as displayed on the electronic shelf labels. She immediately takes one as a present for her neighbour who loves flowers. As the supermarket is crowded today, she uses her IoT-Phone to participate in a virtual queuing system at the checkout, so that she can browse through the shelves while already being in the queue for checkout.

After Salomée is done grocery shopping and about to go back home, she receives a notification on her IoT-Phone that Robert, her father, used his last ampoule of insulin and recommends to her to stop by a pharmacy to buy new medication. Salomée is glad to see that her IoT-Phone can show her the location of the closest pharmacy around.

She enters the pharmacy and picks a package of insulin ampoules. The clerk scans the medication and is asked by the local pharmacy software to ask for Salomée's health ID to verify that she is allowed to buy this kind of drug. Salomée hands over her health ID and when the clerk scans it, the software displays that Salomée is permitted to buy it.

In summary the regarded storyline will give us an impression of how IoT-A components can help consumers and retailers to handle or manage daily challenges. It shows that IoT affects the whole supply chain. Beginning at the production site across the transport and retail part up to the customer, IoT is able to facilitate the whole process and to improve the service.

4.2 Demonstrated concepts

This section gives an overview on what kind of concepts of the ARM may be demonstrated by the defined storyline of the retail use case. Therefore a mapping of the UC's single scenes, which are detailed in section 4.3, to the functional components of the ARM's functional view was made.

Figure 17 shows the current development status[3] of the functional groups and functional components after IR1.4 (Nettsträter, et al., 2011) but before the delivery of D1.3. It was decided to use an intermediate version of the functional view, because a fundamental change had already been done and the version out of IR1.4 or even D1.2 would have been too different. The version shown in the figure will be closer to the final functional view of D1.3.

[3] The delivery date of this document

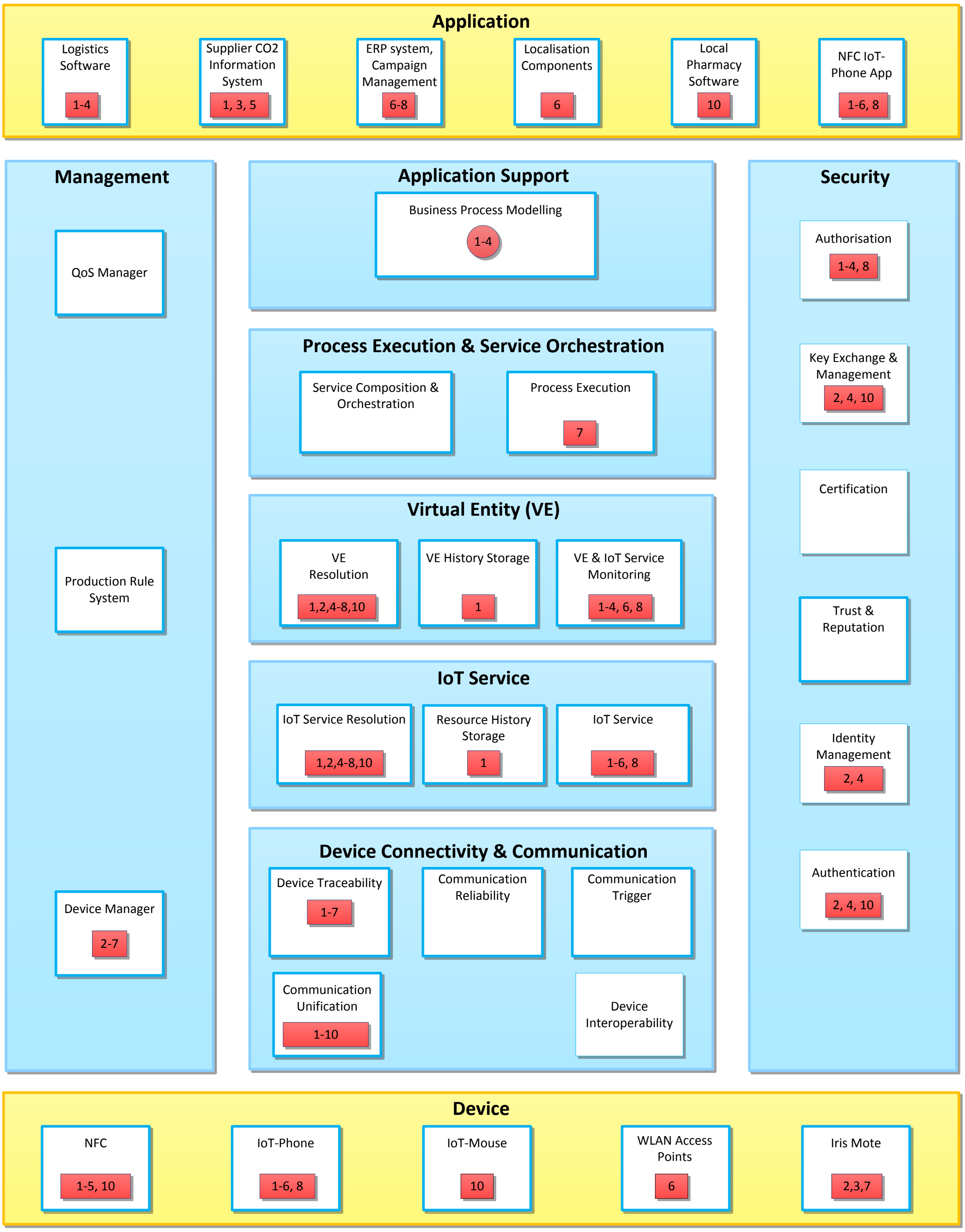

Figure 17: Functional component mapping of defined Retail use case scenes

The figure shows the scenes 1 to 10 mapped to functional components of the ARM. A scene is shown as red box, while a functional component is shown as white box. The numbers in the red box state a specific scene number or a range of scenes which are mapped. Please note that a mapping does not necessarily imply that a mapped component is needed for a later implementation, but instead shows what ARM concepts could be demonstrated.

4.3 Description of Scenes

4.3.1 Users and Roles

Ted	Truck driver for manufacturer of orchids	Ted (m, 34) is a delivery man working for an orchid plantation. His job is facilitated by the IoT-A functions of his IoT-Phone.
John	Shop owner	John (m, 51) is, among other tasks, in charge of dis-/approving of incoming deliveries at his shop.
Carlo	Shop assistant	Carlo (m, 46) works for John at the shop. He takes care of incoming deliveries.
Salomée	Relative of Robert	Salomée (f, 29) represents the younger demographic who benefits from IoT-A enabled applications. This demographic is also tech savvy and highly proficient with new technologies.
Robert	Patient and relative of Salomée	Robert (m, 55) is suffering from type II diabetes, hypertension and faints from time to time by acute syndromes. He also suffers from high blood pressure and is a bit overweight. Apart from this he is generally in a healthy condition and mobile. Robert is living together with his mother Jane.

Table 11: Users and Roles in the retail use case

4.3.2 Scene 1 – Provisioning of Meta Information for Products (TID)

This scene shows how an application running in an IoT-Phone, in conjunction with reading tags attached to products, is able of provisioning some meta information (product's owner, location and CO_2 product's figure generated during production process) into a backend server where it can be later updated or consulted.

The IoT-Phone's application is in charge of reading product's tag info, fill additional information about the product, and provisioning this info to the Resource History Storage server.

Scene summary:

- Location: Production site
- Covered topics
  - Product identification
  - Provision of product's Information in IoT Resource History Storage Server
- Key benefits:
  Products identified with tags + provision product's information in backend helps later processes

4.3.2.1 Description from a User's Perspective

The scene takes place at the production facility of any of the products we are interested in (wine, cheese, and orchids). The scene takes place when the production of each product has finished, and each product is tagged with RFID tags on item level.

The person in charge of the process uses an IoT-Phone to read each product's tag. Later he is asked to fill a form with information about the product (type, owner). The application obtains GPS location, and calculates the CO_2 generated by the product during the production process). Finally it connects to the backend server to update this information (RFID Id, owner (manufacturer), location (production facility) and CO_2 product's figure) to a backend sever where later it can be updated during the travel into the local store or when consulted by any interested service.

4.3.2.2 Relation to the Domain Model

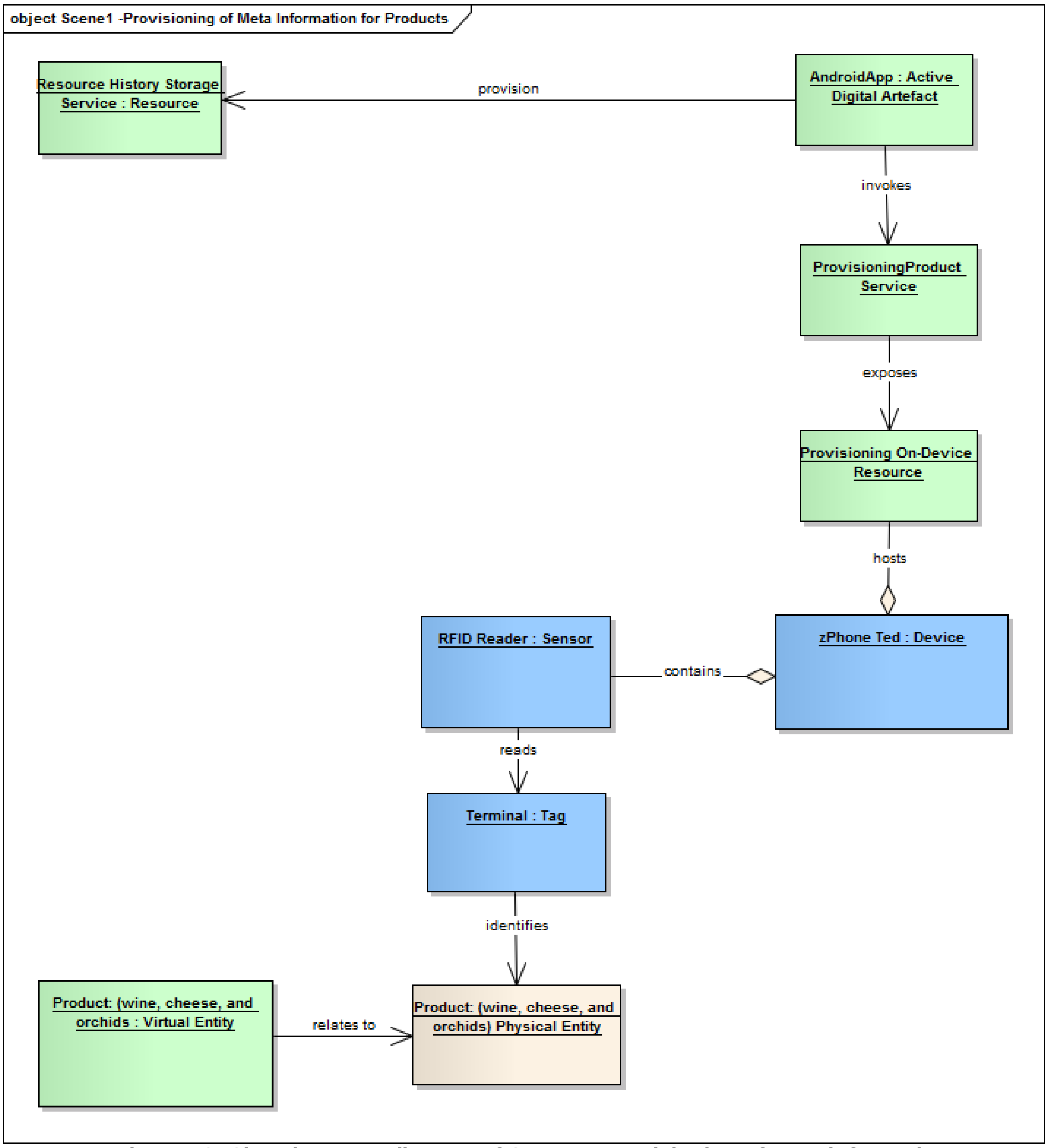

Figure 18: Class instance diagram of Scene 1: provisioning of meta information

The figure shows the modeling of the different components used in scene 1.
The Android application as an *Active Digital Artefact* which is embedded in the IoT-Phone *Device* is interacting with the embedded RFID reader *Sensor for* reading the terminal *Tag*. This *Tag* identifies the *Physical Entity* of the product with its corresponding *Virtual Entity*. Information [Resource] such as CO2 output during production process, owner (manufacturer), location (production facility) is created. The ID [identifier] of the RFID tag may be used to retrieve the URL [resolution service] of the service accessing the resource [Resource]

4.3.2.3 Implementation details

The following table contains software, hardware and components which are planned to be used for the implementation of this scene. Details on the individual items may be seen in chapter 6.

Applications	Devices	Functional Components
SW.TID.1 (Mobile application)	HW.TID.1 (IoT-Phone)	C.ALUBE.1 (Virtual-entity Resolution)
	HW.IML.6 (RFID label)	C.ALUBE.2 (IoT Service Resolution)
		C.TID.1 (Resource History Storage)

Table 12: Applications, devices and functional components planned for scene 1 implementation

4.3.3 Scene 2 – NFC supported check in and assisted loading (FHG IML)

This scene shows how a personal assistant like the IoT-Phone can prevent mistakes in loading goods by automatically comparing loaded items with an order stored in an IT system. By scanning the attached barcodes the IoT-Phone registers for sensor events of the corresponding sensor node.

Scene summary:

- Location: Production site
- Covered topics
    - Automatic check in with NFC
    - Paper-less loading
    - Loading control with personal assistant
- Key benefits:
  Assisted and paper-less loading makes loading faster and protects against errors.

4.3.3.1 Description from a User's Perspective

Ted, the truck driver, arrives at the gardener's production site to pick up some goods he has to transport. Up to now he does not know anything about his task for this day. To get more information he checks in by holding his IoT-Phone above the check in terminal of the manufacturer which is located in front of the entrance barrier. The NFC reader inside the terminal reads the tag located in his IoT-Phone and sends a notification to the manufacturer's ERP system. This retrieves the transport order which has been assigned to Ted from the transport order database and sends all information including the gate number to Ted's IoT-Phone.

After he got the information Ted drives to the appropriate gate and starts loading the intelligent load carriers containing the orchids into his truck. All load carriers are equipped with a sensor node which measures temperature and humidity. Every time Ted puts a carrier into the truck, he uses his IoT-Phone to scan the load carrier's barcode to mark it as loaded and sign up for sensor events of the attached sensor node. After he finished loading, Ted confirms it to the manufacturer, receives his shipping order, and starts driving.

4.3.3.2 Relation to the Domain Model

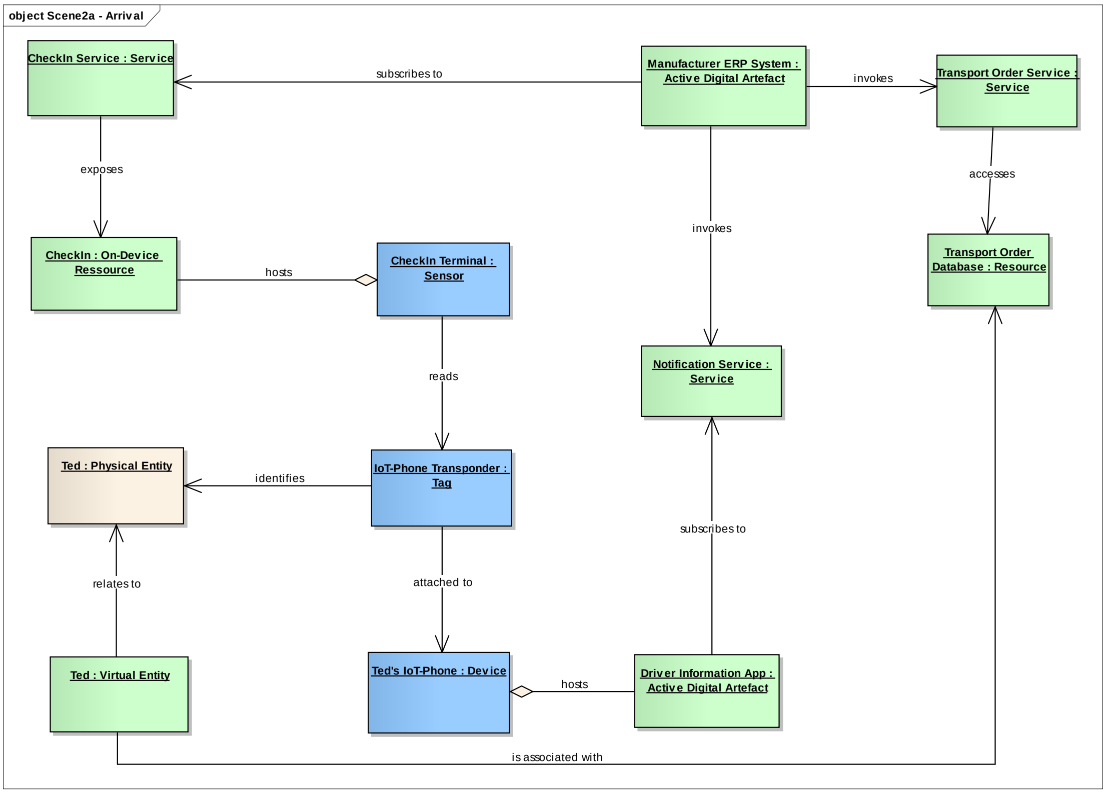

Figure 19: Class instance diagram of Scene 2: arrival and check-in at production site

Figure 19 shows the modeling of the different components used in arrival part of scene 2.
The CheckIn Terminal *Sensor* reads the transponder *Tag* which is attached to Ted's IoT-Phone *Device*. Over the embedded CheckIn *Service* the notification data is sent to the Manufacturer ERP System which is defined as a *Digital Artefact*. This invokes the Transport Order *Service* for searching the transport orders which are associated with Ted's *Virtual Entity*.
In the next step the found transport order information is sent to the Notification *Service* which has been subscribed by the Driver Information App an *Active Digital Artefact* running on Ted's IoT-Phone.

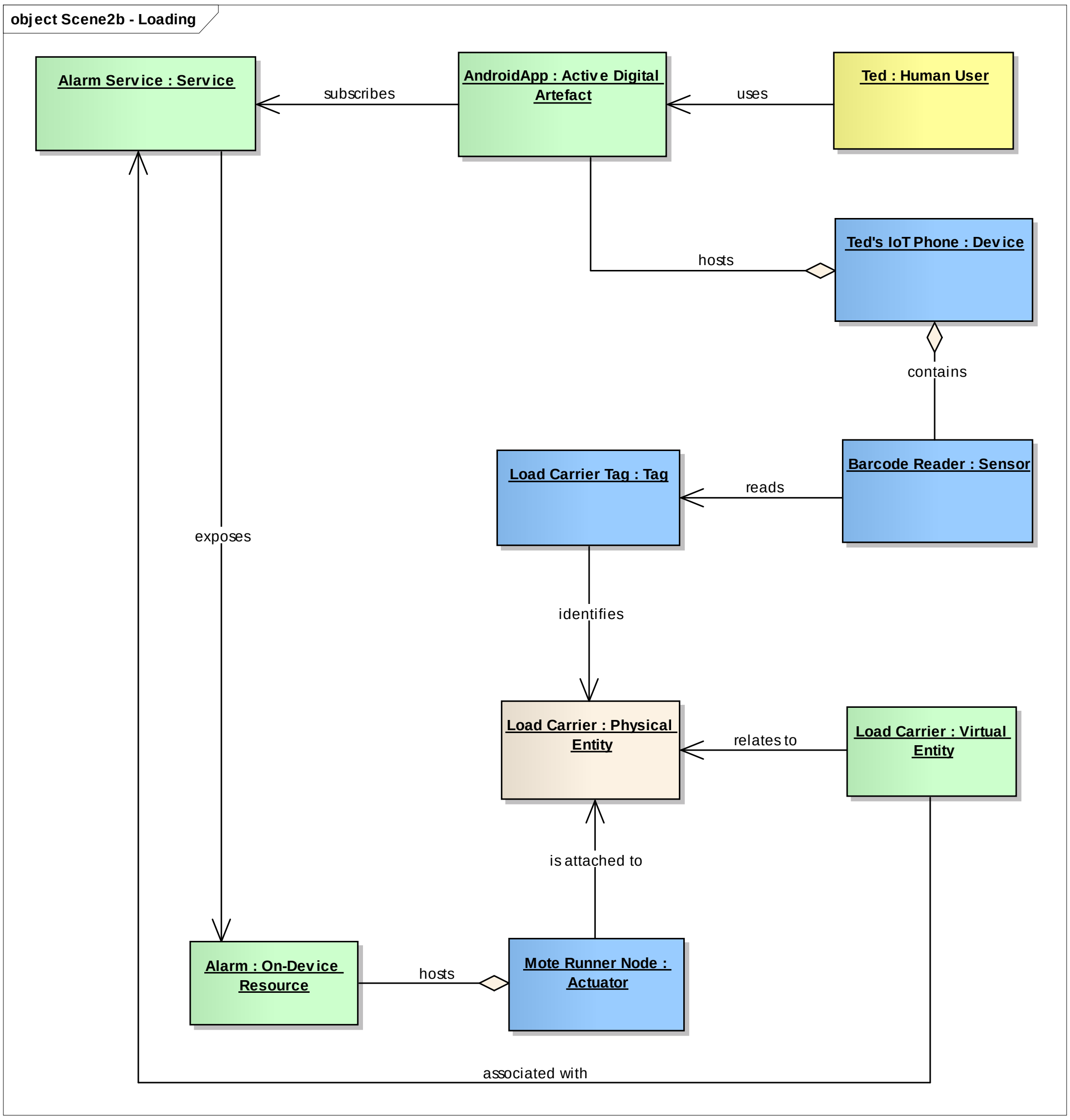

Figure 20: Class instance diagram of Scene 2: Loading goods

In Figure 20 the class instance diagram of the loading process is shown.
The sensor node *Device* which is attached to a load carrier *Physical Entity* hosts a measurement *Service* as an *On-Device Resource* which will be subscribed by the AndroidApp (*Active Digital Artefact*) for getting live measurement data. By scanning the Load Carrier *Tag* with the embedded RFID-Reader *Sensor* of Ted's IoT-Phone *Device*, which identifies the *Physical Entity* of the Load Carrier, the AndroidApp gets access to the Alarm *Service* which corresponds to the *Virtual Entity* of the Load Carrier.

4.3.3.3 Implementation details

The following table contains software, hardware and components which are planned to be used for the implementation of this scene. Details on the individual items may be seen in chapter 6.

Applications	Devices	Functional Components
SW.IML.1 (Mobile application)	HW.IML.1 (IoT-Phone)	C.ALUBE.1 (Virtual-entity Resolution)
SW.IML.2 (Manufacturer backend)	HW.IML.4 (RFID/Barcode label)	C.ALUBE.2 (IoT Service Resolution)

Applications	Devices	Functional Components
SW.IML.4 (Sensor node application)	HW.IML.6 (NFC device)	
SW.IML.5 (Terminal application)		

Table 13: Applications, devices and functional components planned for scene 2 implementation

4.3.4 Scene 3 – Transport monitoring with Smart Load Carriers (FHG IML)

This scene shows how live sensor monitoring of smart load carriers can prevent the transported goods from being damaged due to environmental influences. In this context the load carrier is called smart because it is equipped with sensors and can communicate with other devices in terms of wireless radio technology. With this hardware every load carrier continuously measures its environmental parameters and sends all measurements by the embedded event service to Ted's IoT-Phone which has subscribed to this service.

Scene summary:

- Location: Road
- Covered topics
    - Smart load carrier
    - Monitoring environmental features
- Key benefits:
  Better quality control while transportation makes it easier to prevent and realize damages in transit.

4.3.4.1 Description form a User's Perspective

Ted is driving his truck to the destination. While he is driving, Ted gets hungry and decides to stop and have lunch. He parks the truck at a resting spot, turns off the engine and goes into a nearby restaurant. Unfortunately, Ted forgot that by turning of the engine, air condition for the orchids shuts off too, and since it is a very hot day, the temperature inside the truck starts rising. When temperature reaches a predefined critical level inside one of the load carriers, one of its sensors notices this and its node sends an emergency signal to Ted's IoT-Phone.
On the IoT-Phone's display, Ted can now see that the orchids in load carrier number 6 are in danger due to high temperature so he rushes back to the vehicle and turns the air condition back on. The IoT-Phone also keeps track of any alert messages it receives from the load carriers and saves this message history for future inspection.

4.3.4.2 Relation to the Domain Model

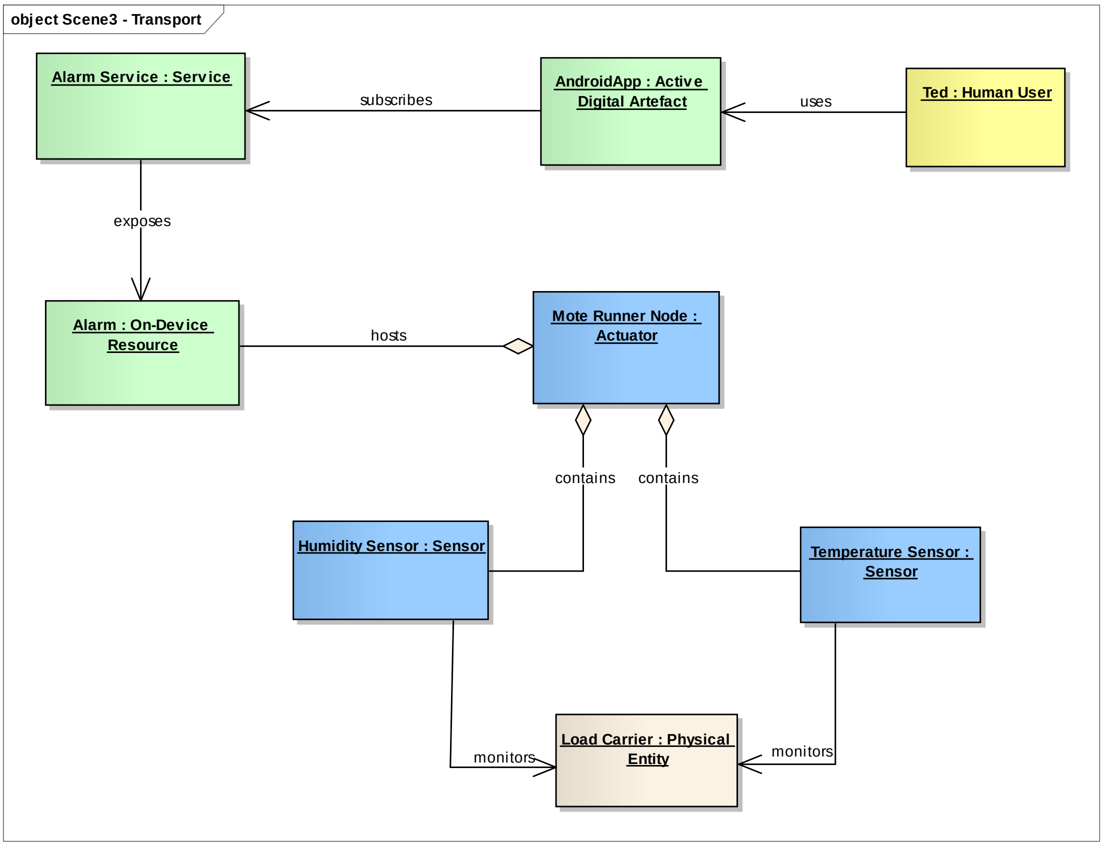

Figure 21: Class instance diagram of Scene 3: load carrier monitoring

The modeling of scene 3 is shown in the above figure. The sensor node *Device*, which is connected to a temperature and a humidity *Sensor*, monitors the corresponding load carrier *Physical Entity*. The measured data is sent to the Android application which is an *Active Digital Artefact* over the integrated measurement *Service* as an *On-Device Resource*.

4.3.4.3 Implementation details

The following table contains software, hardware and components which are planned to be used for the implementation of this scene. Details on the individual items may be seen in chapter 6.

Applications	Devices	Functional Components
SW.IML.1 (Mobile application)	HW.IML.1 (IoT-Phone)	none
SW.IML.4 (Sensor node application)	HW.IML.3 (Sensor node)	
	HW.IML.8 (Gateway)	

Table 14: Applications, devices and functional components planned for scene 3 implementation

4.3.5 Scene 4 – Assisted quality check and digital signature (FHG IML)

This scene shows the capabilities of ownership transfer of goods and digital signature of supply orders in combination with an assisted quality check within the unloading process in the store.

Scene summary:

- Location: Store
- Covered topics
    - Smart transport items
    - Ownership transfer
    - Digital signature
    - Quality check
- Key benefits:
  Paper-less signature and assisted quality check protect against human errors and manipulations. These features are able to raise the quality of shipped goods.

4.3.5.1 Description from a User's Perspective

Ted is continuing to drive his truck to the destination. Without further incidents, finally Ted reaches the store. Once there he drives up to the store's NFC terminal and scans it to let the staff at the store know of his arrival. After a short time Ted receives a message on his IoT-Phone, letting him know where he needs to go to unload.
When Ted arrives at the gate, he is welcomed by Carlo, a shop assistant. They unload the load carriers together and have them pass through an RFID gate that recognizes them automatically. Then they wait for John, the shop owner to have him confirm the delivery.
When John arrives, they briefly talk and then hold their IoT-Phones close to each other to transfer the sensor history report of the intelligent load carriers from Ted's IoT-Phone to John's. John's IoT-Phone then draws his attention to a recorded rise in temperature during transportation, so John does a visual inspection on the orchids. Once he has assured himself that all the flowers are still in good quality, he confirms the delivery and holds up his IoT-Phone to send a message of approval to Ted's IoT-Phone.

4.3.5.2 Relation to the Domain Model

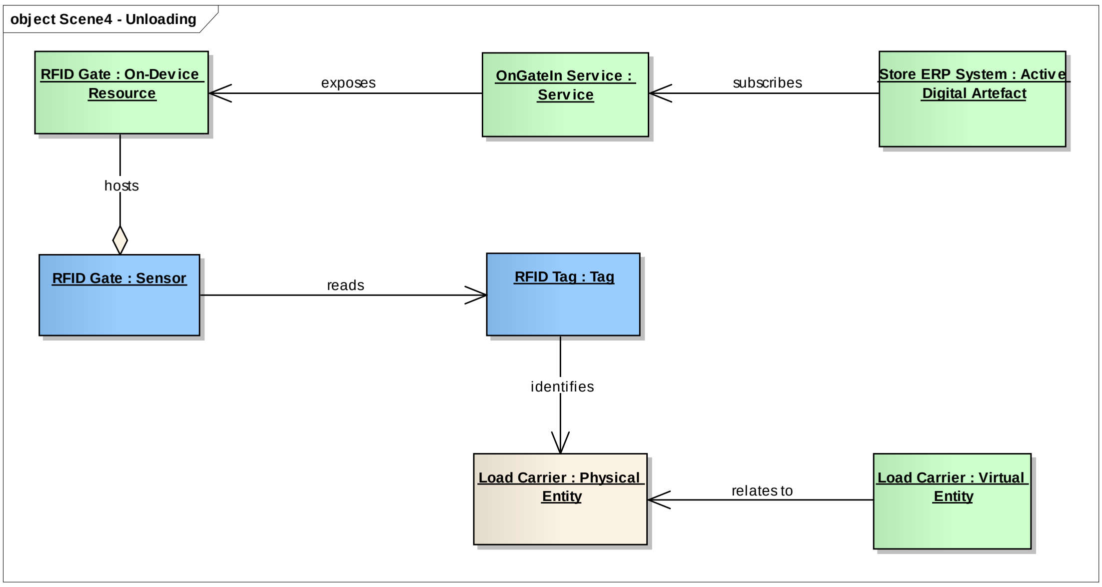

Figure 22: Class instance diagram of Scene 4: goods-in at store

The unloading of scene 4 is modeled in the above figure. The RFID gate which was defined as *Sensor* reads the RFID *Tag* which identifies the unloaded load carrier *Physical Entities*. The integrated on-gate-in *Service* sends the identified load carrier information to the store system *Virtual Entity*.

4.3.5.3 Implementation details

The following table contains software, hardware and components which are planned to be used for the implementation of this scene. Details on the individual items may be seen in chapter 6.

Applications	Devices	Functional Components
SW.IML.1 (Mobile application)	HW.IML.1 (IoT-Phone)	C.ALUBE.1 (Virtual-entity Resolution)
SW.IML.2 (Manufacturer backend)	HW.IML.2 (IoT-Phone)	C.ALUBE.2 (IoT Service Resolution)
SW.IML.3 (Store backend)	HW.IML.4 (RFID/Barcode label)	
	HW.IML.5 (RFID gate)	
	HW.IML.7 (RFID label)	

Table 15: Applications, devices and functional components planned for scene 4 implementation

4.3.6 Scene 5 – NFC based Shopping Assistant (SAP)

This scene shows how IoT technologies like sensor technologies built into consumer electronic devices and NFC tags coupled with the Internet of Things Architecture can provide useful meta-information to the customer to enhance the overall shopping experience.

The scene demonstrates how IoT enabled products can be identified by NFC tags and product related meta-data is provided. Furthermore, consumer sensitive product recommendations can be delivered. The scene shows what a direct human-to-machine interaction with IoT-A would be like. As such, the overlap between the existing internet and the future internet of things is shown.

Scene summary:

- Location: Retail Store
- Covered topics
  - Usage of a Mobile Shopping Assistant
  - Carbon Footprint Information
  - NFC Reading for Product Identification
  - Determination of over- / underperformers
- Key benefits:
  Mapping of Physical Entities to Backend Information accessed on mobile devices in order to improve the customer experience and support his purchasing decisions

4.3.6.1 Description from a User's Perspective

Salomée goes to the refrigerated section and searches for some nice cheese. She takes a package of cheese and reads the supplier's information on the backside. But she wants more information, takes out her IoT-Phone and reads the NFC tag on the backside of the case.

Her virtual Shopping Assistant, a nice application on her IoT-Phone, tells her not only some hard facts about the cheese such as the CO_2 foot print or the best before date but also some additional useful information like recipes. For example she could add all of the items needed to cook that recipe to her personal electronic shopping list at once.

It also compares the product with other cheeses she has bought in the past and suggests another cheese matching her common taste more. She looks for that cheese and takes it.

4.3.6.1 Relation to the Domain Model

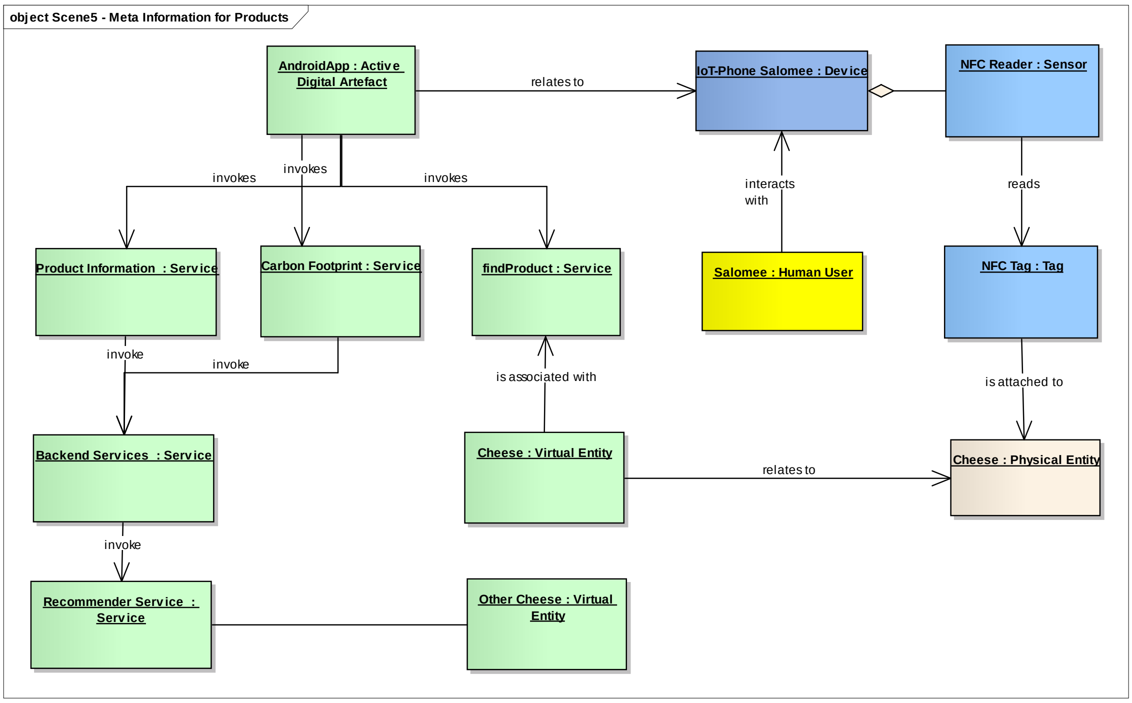

Figure 23: Class instance diagram of Scene 5: shopping assistant

The domain model diagram, i.e. the process expressed in terminology and concepts of the IoT-A reference model (see Figure 23), shows how Salomée, who is a *Human User*, goes to the store for some cheese and takes a package of cheese which is a *Physical Entity*, and reads the information printed on the wrapping. To get more information she reads the NFC-Tag, which is a *Sensor*, with her IoT-Phone which is in itself a *Device*. Her Virtual Shopping Assistant is modeled as a *Digital Artifact* which invokes a plethora of services that tell Salomée some hard facts about the product and some additional meta-data like recipes corresponding to that product and comparable products depending on her shopping behavior.

4.3.6.1 Implementation details

The following table contains software, hardware and components which are planned to be used for the implementation of this scene. Details on the individual items may be seen in chapter 6.

Applications	Devices	Functional Components
SW.SAP.1 (Mobile Consumer Application)	HW.SAP.1 (Tagged Cheese)	C.ALUBE.1 (Virtual-entity Resolution)
SW.IML.3 (Store backend)	HW.SAP.2 (IoT-Phone)	C.ALUBE.2 (IoT Service Resolution)

Table 16: Applications, devices and functional components planned for scene 5 implementation

4.3.7 Scene 6 – Location Based Services (SAP)

This scene shows the possible interaction of location tracking solutions – such as Wifi triangulation – with the Internet of Things Architecture to enable location aware applications which identifie the customer’s position in a store and provide location based offerings.

In this way, systems can provide context and consumer sensitive information to users.
The scene shows what a direct human-to-machine interaction with IoT-A would be like. As such, the overlap between the existing internet and the future internet of things is shown.

Scene summary:

- Location: Retail Store
- Covered topics
  - Digital Signage triggered by positioning technologies
  - Recommendations based on previous purchases (cheese -> wine)
- Key benefits:
  Usage of Positioning technologies in the store combined with targeted information helps the store to address the specific needs and interests of the consumer. The utilization of the previous purchase history allows for contextually optimized purchase recommendations.

4.3.7.1 Description from a User's Perspective

Salomée proceeds with her shopping. The Shopping Assistant on her IoT-Phone informs that her favourite wine is out of stock, as it checks the ATP (availability to promise) of her favourite wine, since she has wine on her shopping list. Due to the fact that her favourite wine is just sold out, the Shopping assistant recommends another wine based on ratings to Salomée. It recommends her another similar Italian wine that is rated higher instead, but more expensive. Anyhow, she enters the wine area to look for alternatives.

Since she does not find any adequate wine, she decides not to buy any wine. While she is about to leave the wine area of the store, because she has cheese in her shopping cart, a big TFT display, which is an Active Digital Entity, announces a 30% sale on all wines in the "Italian wine section" for buyers of cheese. To personalize the discount anonymously, Salomée's shopping list is checked for selected products and cheese is found. The discount is set valid only for buyers of cheese. The 30% discount ad is displayed on the screen.

She smiles, walks back and takes the wine that was recommended by the Shopping Assistant.

4.3.7.2 Relation to the Domain Model

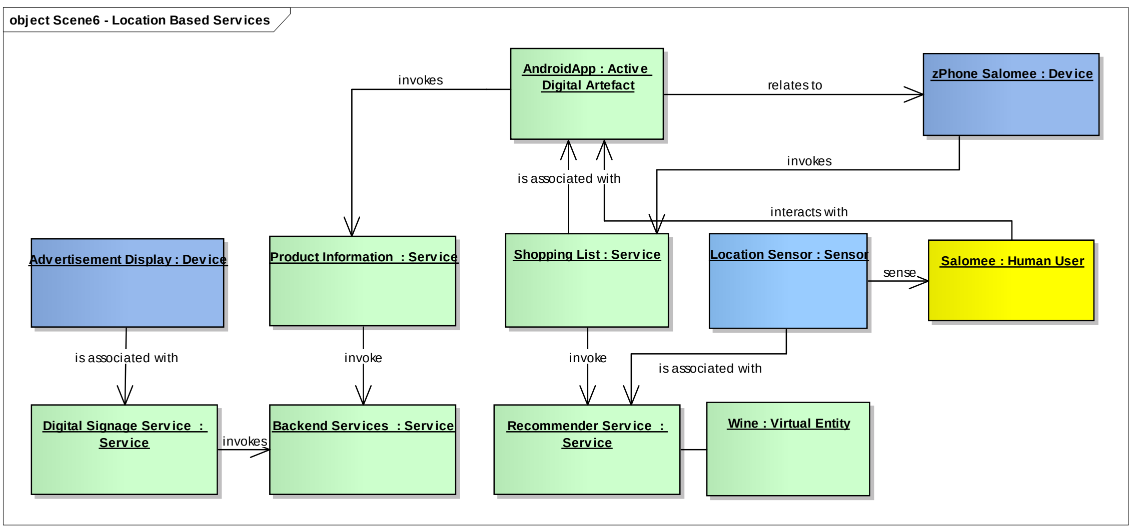

Figure 24: Class instance diagram of Scene 6: location based services

The diagram above (see Figure 24) illustrates the related components of the scene, expressed in terminology and concepts of the IoT-A domain model.

As in the previous scene, the Shopping assistant on Salomée's IoT-Phone is modeled as a *Device*. The Shopping Assistant is modeled as an *Active Digital Artifact* that invokes the respective backend *Services* in the store.

Salomée is recognized by the location system which is modeled as a *Sensor* in the wine area while she is entering. The technical detail of who actually is recognized (Salomée or her phone) is not fully determined at the time of this writing. In any case, the IoT-Phone or Salomée as a *Human User* is now registered for the wine area in the store backend system.

The advertisement shall be displayed on the big TFT display which is naturally modeled as a *Device*. The recommendation and price reduction process is modeled with several service calls between the *Active Digital Artefact* (her IoT-Phone app) and the backend *Services* in the store.

4.3.7.3 Implementation details

The following table contains software, hardware and components which are planned to be used for the implementation of this scene. Details on the individual items may be seen in chapter 6.

Applications	Devices	Functional Components
SW.SAP.1 (Mobile Consumer Application)	HW.SAP.2 (IoT-Phone)	C.ALUBE.1 (Virtual-entity Resolution)
SW.IML.3 (Store backend)	HW.SAP.3 (Advertisement Display)	C.ALUBE.2 (IoT Service Resolution)
SW.SAP.2 (Store Digital Signage)	HW.SAP.4 (Location Sensing Technology)	
SW.SAP.3 (Instore Positioning Software)		

Table 17: Applications, devices and functional components planned for scene 6 implementation

4.3.8 Scene 7 – Sensor Based Quality Control (SAP)

This scene shows how sensors monitor perishable goods in a store. The sensor infrastructure measurements are used for estimating the quality of a rare and expensive form of Chinese orchids. Depending on the luminance, humidity, and temperature of the environment, the estimated future quality of the orchids is determined and prices are reduced, even before a perceivable degradation of quality occurs. By applying this sensor based quality control and combining it with dynamic pricing, it is ensured that the goods are sold before quality degradation is likely to occur.

From a business and industry perspective, the scene demonstrates two important retail related concepts: Dynamic pricing and quality control of perishable goods. Dynamic pricing as a real-time tool for price optimization strategies has always been crucial for profit maximization. In contrast to the state of the art, dynamic pricing in the featured use case is not performed on static information such as best before end dates in the transaction data of the backend ERP system, but it is based on real time IoT data gathered from a sensor infrastructure.
As about 20% of perishable goods never reach the consumer, but are disposed of before, either in the store or in the supply chain, the utilization of IoT sensors is also an interesting concept to implement quality control of perishables and thus reduce waste and increase profits at the same time.

Scene summary:

- Location: Retail Store
- Covered topics
  - Sensor Networks Measuring Environmental Parameters
  - Campaign Management for Dynamic Pricing
  - Electronic Shelf Labels updated in Real-Time
- Key benefits:
  IoT technologies help optimize pricing and product quality by measuring the real-world environmental parameters that influence the quality of perishable goods. This is a strong benefit for both consumers and retailers, as consumers receive potential discounts whereas retailers reduce waste and optimize in-store logistics.

4.3.8.1 Description from a User's Perspective

This Saturday, Salomée decides to try out the new supermarket that opened recently. As she enters, she is positively surprised by its spaciousness and its calm atmosphere. Her mobile shopping application points her to a special offer of non-food items, namely rare and fragile orchids from China.

She immediately thinks of her neighbor Heinrich who loves flowers and would appreciate them as a gift from her. Just as she approaches the shelf with the orchids, she realizes their price going down by 10%. Happy about the price reduction, she immediately picks an orchid and continues shopping.

4.3.8.1 Relation to the Domain Model

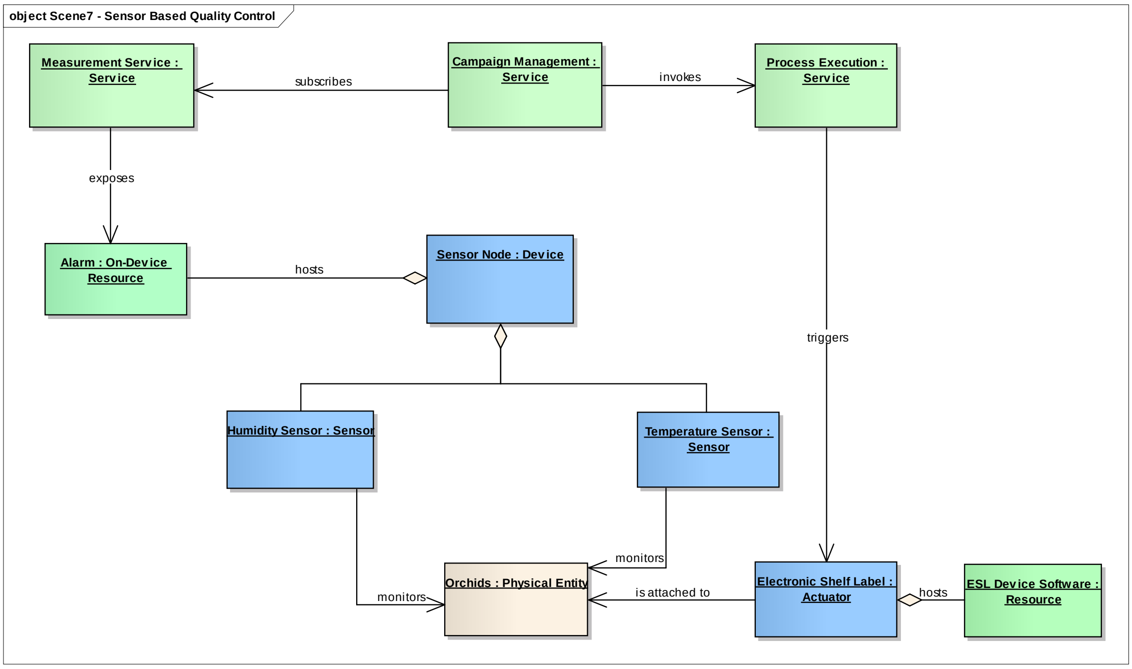

Figure 25: Class instance diagram of Scene 7: sensor based quality control

The modeling of the scene is closely connected to scene 3 (Transport), as in both cases smart *Sensors* continuously monitor certain environmental parameters crucial for the quality of the fragile orchids which are modeled as *Physical Entities*, as e.g. the humidity level must be in a certain range, so that neither dryness nor inundation, neither warm nor chill, nor non-optimal luminance harms the quality of the orchids. As it is as difficult to maintain optimal conditions in a supermarket as on a truck, the *Sensors* would only send alerts on severely inappropriate conditions, but the average conditions over time are still measured to calculate the estimated point in time for a perceivable degradation in quality.

Based on an estimation of environmental measures from humidity and temperature *Sensors*, a threshold of quality degrading environmental parameters is reached and the platform (modeled as *Services*) automatically reduces the price at the POS (the price display is an *Actuator*, but it also hosts a *Resource* for storing it) in order to foster the sales of the orchids. Data from the Store Platform is propagated to the cashier system (not modeled) and also wirelessly to electronic shelf labels (ESLs) that function as complementary *Actuators* to the *Sensors* that measure the environmental conditions of the orchids.

This scene also involves process execution which is modeled as a *Service* that the campaign management triggers whenever relevant price changes occur that need o be propagated to the ESL *Actuator*.

4.3.8.1 Implementation details

The following table contains software, hardware and components which are planned to be used for the implementation of this scene. Details on the individual items may be seen in chapter 6.

Applications	Devices	Functional Components
SW.SAP.4 (Campaign Management Software)	HW.SAP.5 (Electronic Shelf Labels)	C.ALUBE.1 (Virtual-entity Resolution)
SW.IML.3 (Store backend)	HW.SAP.6 (Sensor Devices)	C.ALUBE.2 (IoT Service Resolution)
		C.SAP.1 (Process Execution)

Table 18: Applications, devices and functional components planned for scene 7 implementation

4.3.9 Scene 8 – Queue Management Checkout Assistance (SAP)

This scene shows how the checkout process in the store can be enhanced with the IoT-Phone presented in the other scenes. Mobile solutions for retail environments usually aim at improving the shopping experience, but they do not address a factor perceived as cumbersome by consumers: queuing. This scene shows how respective shopping applications can be extended to improve the checkout experience. The scene is based on a checkout queue virtualization system that makes waiting more convenient for customers and which is likely to increase the retailer's revenue. The customer takes a virtual ticket on her smart phone and is informed when she can proceed to the checkout.

Scene summary:

- Location: Retail Store
- Covered topics
  - Combine display and mobile devices to monitor and manage the supermarket customers' waiting time
  - Integration within the mobile shopping application already used in other scenes
- Key benefits:
  IoT technologies help optimize waiting times for customers at the cashier desk

4.3.9.1 Description from a User's Perspective

When Salomeé has finished her shopping, she presses a checkout button on her IoT-Phone. The application receives a virtual ticket and the position in the virtual queue is displayed along with the estimated waiting time. The waiting time is an approximate value, which is computed by taking an empirical average dispatch time per customer. It is always possible to cancel a ticket when the customer decides to continue shopping.

After Salomeé has pressed the checkout button, a recommendation and targeting engine recommends goods that are usually bought during checkout. The virtual queuing system is an elegant way to determine when an individual customer is ready to check out. In marketing, such purchases are called impulsive purchases. The restriction that those items have to be located next to the counters vanishes, which gives more flexibility to the shop on how and where to place the items.

While Salomeé is waiting in the virtual queue, she wanders around the shelves and watches advertisement being displayed on the digital signage systems in the store, before she is finally informed by her IoT-Phone that her time for checkout has now come. Accordingly, she is able to quickly pay and leave the store with her purchases.

4.3.9.2 Relation to the Domain Model

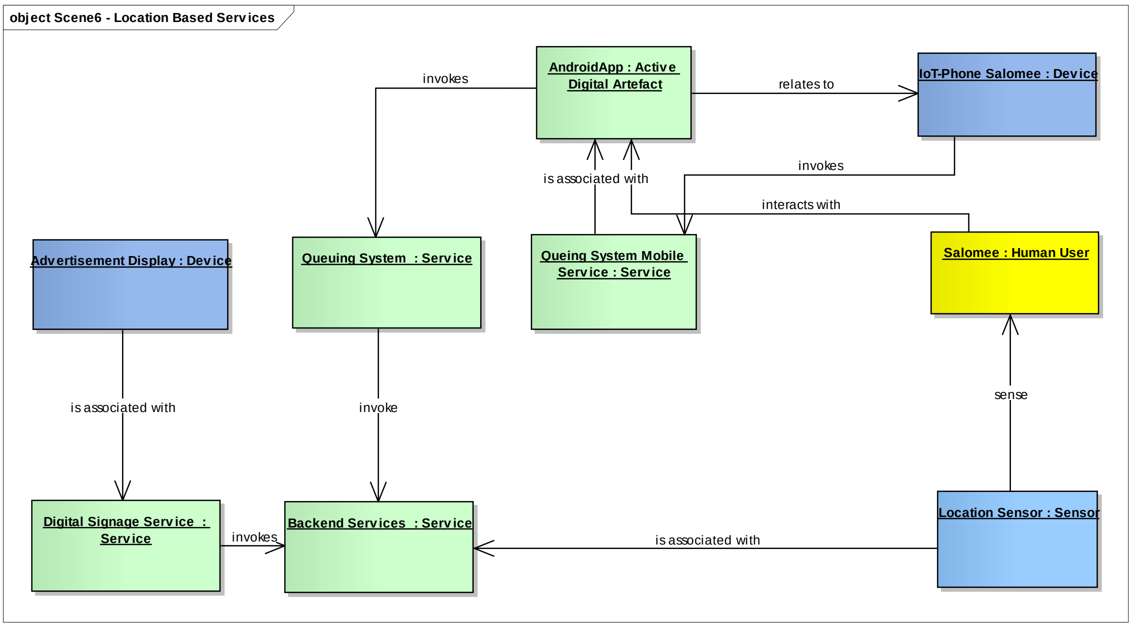

Figure 26: Class instance diagram of Scene 8: Queue Management Checkout Assistance

As figure 26 shows, the Shopping assistant on Salomée's IoT-Phone is modeled as a *Device*. The Shopping Assistant is modeled as an *Active Digital Artifact* that invokes the respective backend *Services* in the store.

Salomée is recognized by the location system which is modeled as a *Sensor*, so that the advertisement display can be used according to her location by the backend system. The technical detail of who actually is recognized (Salomée or her phone) is not fully determined at the time of this writing. In any case, the IoT-Phone or Salomée as a *Human User* is now registered for the wine area in the store backend system.

The advertisement shall be displayed on the big TFT display which is naturally modeled as a *Device*. The recommendation and price reduction process is modeled with several service calls between the *Active Digital Artefact* (her IoT-Phone app) and the backend *Services* in the store.

4.3.9.3 Implementation details

The following table contains software, hardware and components which are planned to be used for the implementation of this scene. Details on the individual items may be seen in chapter 6.

Applications	Devices	Functional Components
SW.SAP.1 (Mobile Consumer Application)	HW.SAP.2 (IoT-Phone)	C.ALUBE.1 (Virtual-entity Resolution)
SW.IML.3 (Store backend)	HW.SAP.3 (Advertisement Display)	C.ALUBE.2 (IoT Service Resolution)
SW.SAP.2 (Store Digital Signage)	HW.SAP.4 (Location Sensing Technology)	

Table 19: Applications, devices and functional components planned for scene 8 implementation

4.3.10 Scene 9 – Low Insulin Supply (link to Health UC) (ALU-BE)

This scene illustrates the link between the Health and the Retail use case. The medicine supply of the patients is continuously monitored and in case the supply has lowered below the threshold, automatic actions can be triggered to assure that the patient is never out of medicine.

Scene summary:

- Location: this scene originates at Robert's house; the event is forwarded to the IoT-Phone of Salomée.
- Covered topics:
    - Home Care
    - Remote health assistance
    - Smart Objects
    - Link to Retail use case
- Key benefits:
  With IoT-A technology, it is easy to have cross application domain use cases, something that is extremely difficult to obtain with a traditional approach.

Further details on this scene are described in detail in the Health use case, paragraph 3.3.5.

4.3.11 Scene 10 – The Selling of Controlled Substances at the Pharmacy (HSG)

This scene shows how the linked information between persons and objects in the future Internet of Things can be used to regulate the sale of controlled substances. This adds a layer of safety for the consumer. The technological entry points into this scene are ubiquitous sensors such as the IoT Mouse and the ubiquity of tags in objects, such as in identification cards.

In this scene, IoT-A's resolution services are applied to identify and notify when a controlled substance can be sold, within legal & safety frameworks. The information structures suggested by the IoT-A Reference Model are demonstrated here to show its use in improving the available information of a given entity, such as the ampoules and Salomée, and ensuring that there are no access rights violated in every day transactions.

Scene summary:

- Location: Pharmacy

- Covered topics:
  - Selling controlled pharmaceutical items
  - Safety
- Key Benefits:
  - IoT-A's resolution services are applied to identify and notify when a controlled substance can be sold, within legal & safety frameworks

4.3.11.1 Description from a user's Perspective

Salomée goes to the pharmacy to buy the necessary medication for Robert. When she takes the medication to the clerk, the clerk reads a tag on the ampoules with a nearby sensor.

Upon scanning, the local pharmacy software requests information about the ampoules and the local software becomes aware of the restrictions. The clerk asks for some personal identification and gets Salomée's health ID to determine Salomée's eligibility to buy the drug. Permission is found and the drug is sold. Ownership of the drug transfers to Salomée.

4.3.11.2 Domain Model of Scene

In this section, we show the class diagrams of the Pharmacy ampoule scanning and patient verification scene.

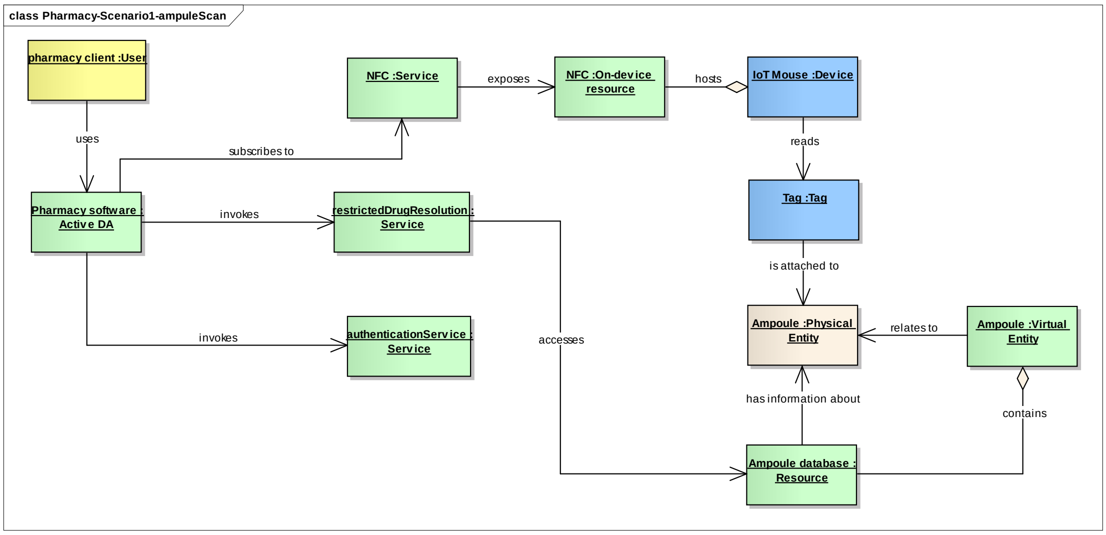

Figure 27: Class instance diagram of Scene 10: ampoule scanning

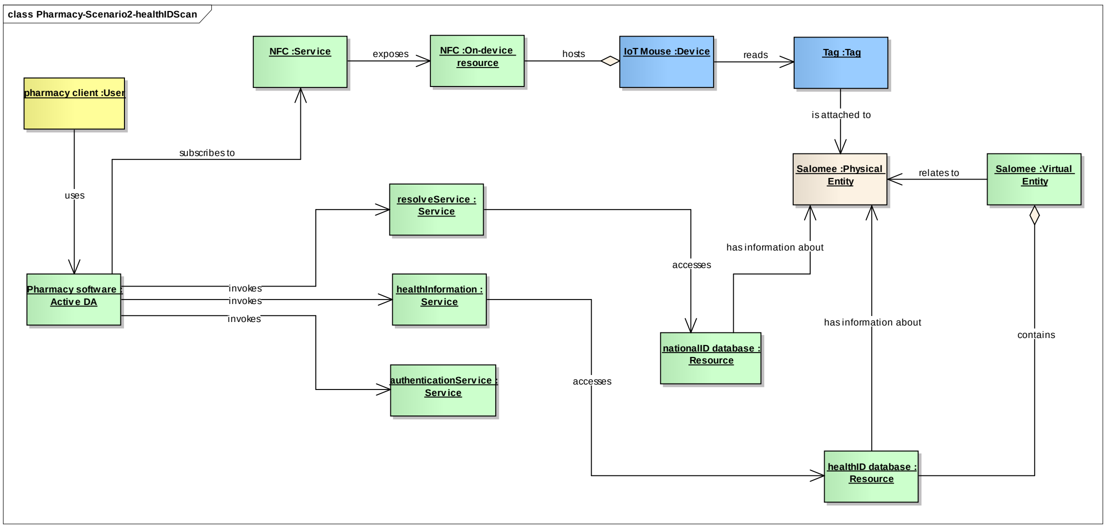

Figure 28: Class instance diagram of Scene 10: Salomée's Health ID scanning

Figure 27 and Figure 28 show the different implemented classes and their relationships for the given use case in order to achieve the desired functionality. Figure 27 shows the classes and their relationships when an ampoule/pill is scanned. It checks whether the scanned ampoule/pill comes under the restricted sale and if yes then asks the patient to show his/her identification. Figure 28 follows up the Figure 27 with the patient's ID scan and verifying whether the patient is allowed to buy the restricted ampoule/pill or not. Each class shown caters to a particular function in the overall picture. For e.g. *Pharmacy software* controls the local GUI and initiates different services. The labels on top of the arrows depict what kind of relationship exists between two classes.

Pharmacy user client uses the *Pharmacy software*. The *Pharmacy software* subscribes to *NFC service*. The service listens to the *NFC tags* in the vicinity and communicates with them. These NFC tags are read using the *IoT mouse* which is operated by the pharmacy user client. NFC tag reader is integrated in the IoT mouse. NFC tags are attached to the ampoules/pills and contain the information of the ampoules/pills.

After scanning an ampoule/pill, it is checked whether the ampoule/pill comes under restricted sale. If yes then the pharmacy user client asks for a valid prescription and health information identification of the patient. This health information is provided as a separate NFC tag which is attached to the patient (Robert in our case) and contains the healthcare information of the patient.

After a NFC tag is read, the *Pharmacy software* calls the *authenticationService* to get the access to the database where information about the patient is stored. Once the client is granted the permission to access the database, a *resolveService* is executed to resolve the health ID (part of healthcare information read from the NFC tag) into a unique ID of the patient.

The unique ID leads to fetch more information about the patient from different connected/distributed databases (e.g. driver license database, health database etc.). Finally the *healthInformationService* is called to get all the healthcare information (address, information of the last visit, prescriptions, doctor details etc.). The EHR resource has all the relevant information about the patients.

After getting the relevant information from the health database, *Pharmacy software* displays it in the pharmacy GUI form.

4.3.11.3 Implementation Details

The following table contains software, hardware and components which are planned to be used for the implementation of this scene. Details on the individual items may be seen in chapter 6.

Software	Hardware	IoT-A components
SW.HSG.2 (Local Pharmacy Software)	HW.HSG.1 (IoT-Mouse)	C.ALUBE.1 (Virtual-entity Resolution)
	HW.HSG.2 (Local Computer)	C.ALUBE.2 (IoT Service Resolution)
	HW.HSG.3 (Tagged ID Cards)	C.HSG.1 (AuthN)
	HW.HSG.4 (Tagged Ampoules)	C.HSG.2 (AuthS)

Table 20: Applications, devices and functional components planned for scene 10 implementation

5. Security and Privacy Considerations

5.1 Vulnerabilities and Threats

The main objective of the different devices used in the use cases are the traceability of objects and/or people. This goal is easily reached but with an important drawback on the security and especially on the privacy of users. Indeed the difference between a simple barcode on an item in a supermarket and an RFID tag or even a sensor is that it emits a unique identifier (named UID). For example, two identical products like two orchids will have exactly the same barcode but two different UIDs. As a consequence, traceability of objects and not only of a certain kind of products is feasible. Moreover, this UID is a pointer on a database where a huge amount of data on the object can be stored. Those data may also be linked to the consumer who bought the item like his credit card number. The two features of RFID systems (Unique identifier and the fact that this UID points towards a database) is an important advantage for logistics but a threat for privacy of users.

This threat is possible due to vulnerabilities of RFID systems. The first one is that communications of RFID tags or sensors can easily be eavesdropped (up to 4 meters) since they are low resources devices which makes encryption of transactions difficult. The second vulnerability is that these devices are passive (remotely powered) and many do not have an on/off switch so they can be interrogated by everybody. Moreover, this request may be done without the consent or knowledge of the user. For the same reason but also due to the mandatory presence of anti-collision protocols, denials of service which can even lead to the destruction of the device are simple to carry out. Finally, eavesdropping and skimming can be used together to perform a relay attack on contactless devices. This attack is able to circumvent any kind of cryptography solutions and as a consequence it may jeopardize a transaction. Those different attacks are exhaustively described in deliverable D5.1 (Savry & Moering, 2011).

In the retail use cases, Ted's IoT-Phone can be remotely interrogated without his knowledge or his communication with the terminal can be eavesdropped and replayed enabling an attacker to replace him. When Salomée has bought cheese, wine or orchids, personal data like her name or her credit card number can be stored in the shop database. Thus, if it is not possible to disable the tags at checkout and if she enters later in the same shop with a tagged item (it is difficult to understand it with cheese or wine because they are edible products but a tagged shirt or the ampoules of insulin for example can be more relevant), she can be traced by the readers of the shop and she can be profiled jeopardizing her privacy.

5.2 Security and Privacy Requirements

The different vulnerabilities for security and privacy in RFID systems can be summarized in a few points:

- each tagged item has a unique identifier
- the UID is a pointer on an huge database
- the tag communication can be eavesdropped
- the tag can be skimmed
- the passive feature and presence of anti-collision functionality is a backdoor to denials of service

One simple requirement to give back to the user the control of his privacy is to ask him to validate any request made on his own objects (tags, contactless cards, sensors, NFC phone). While this solution can avoid skimming and can be implemented on smartphones with keyboards, it is not obvious to develop it for RFID tags or contactless cards which have no human interface. One main requirement to preserve privacy is then to ensure that only the owner of the object or an authorized entity (authorized by the owner) is able to read his own tags or sensors. The owner of the cheese, wine or orchids in our use cases is the manufacturer at the beginning of the scenario, then the retailer and finally Salomée. To preserve her privacy, only Salomée after the checkout must be able to read her tags.

5.3 Countermeasures and Possible Integration

The security and privacy requirements can be reached by ensuring an authentication of the reader by the tag or sensor and by sending a blurred UID. The deliverable D5.2a (van Kampen & Savry, 2011) describes an RFID reader which is able to send noise during the response of the tag to prevent eavesdropping on the communication by a spying probe. Since the reader knows the noise it has sent, it can subtract it to retrieve the tag message while the spying probe is still blurred. Nevertheless, this solution requires an authentication of the reader by the tag or else any kind of commercial reader could be able to interrogate the tag and to retrieve its UID. This authentication can be performed with a challenge-response protocol supported by an asymmetric or symmetric cryptography. The first solution requires a lot of computing resources and might not be commercially available for low cost tags before a certain number of years. NXP develops this solution; further details may be read in D5.2a (van Kampen & Savry, 2011). The second solution enables the use of lightweight cryptographic primitives (like the algorithm PRESENT) but create difficulties to manage the keys since tag and reader must have a common shared cryptographic key.

In the case of an authentication of the reader based on symmetric encryption, it must be possible in the system to modify the cryptographic key in the tag when the owner has changed. For example, when Salomée passes the checkout an ownership transfer must be performed between the retailer and the consumer. The difficulty of the ownership transfer is to try to load in the tag the key of the new owner without being able to know it and without disclosing to the new owner the key of the previous owner. To enable this transfer, Salomée can send a temporary key to the terminal of the checkout (it can be done securely with a noisy reader) with her NFC-capable phone (the IoT-Phone), then the checkout (the current owner) can provide to the tag a command which enable to modify its key with this new key. As a consequence, the tag is in a temporary state where it can be read by the retailer and Salomée without disclosing the two crucial keys. After the checkout, Salomée may by using her NFC-capable phone also modify the temporary key with her own key with the same kind of command (an ownership transfer protocol will be described in detail in the deliverable D5.2b).

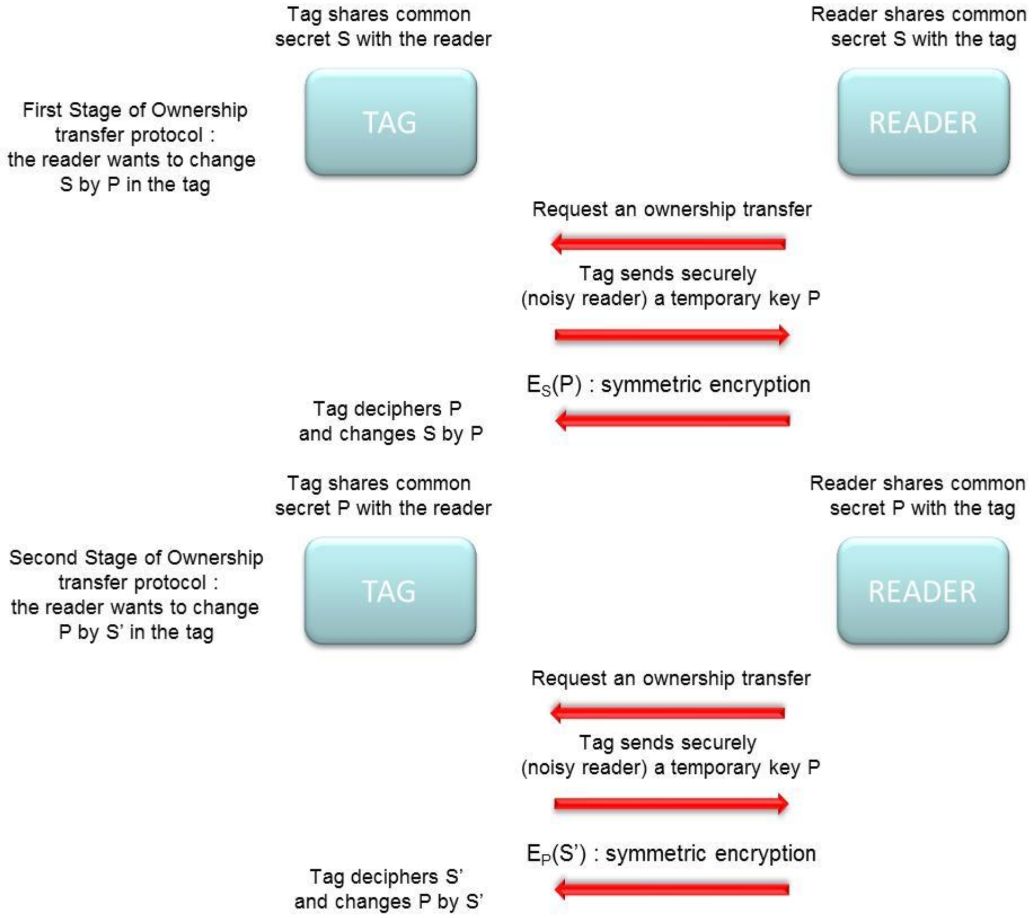

Figure 29: Description of an ownership transfer protocol which enables to securely change the cryptographic key in the tag.

At this stage of the ownership transfer, Salomée can read the UIDs of her tags but she must be authorized to retrieve the data in the database which are linked to the UIDs. Thus, the same mechanism driven by the NFC-capable phone or the RFID reader must be carried out with the database.

One or possibly more scenes will be picked to take into consideration the security and privacy solutions described above. This will enable a comparison between the same scene with and without the privacy solution, thereby illustrating the above security ideas.

6. Implementation Details

6.1 Planned Applications

The following table lists the software applications which are planned to be used for implementation in both health and retail use case in no specific order.

Application ID	Category	Description
SW.ALUBE.1	Backend Application	Patient Care Application - database with patients to monitor - guidance of the patient through measurements - event handling of medical emergencies
SW.ALUBE.2	Mobile Application	Android application for interaction with the patient - frontend for SW.ALUBE.1

Application ID	Category	Description
		- gateway for medical sensors
SW.ALUBE.3	Backend Application	Data analysis application - cross checking of her data against thresholds - interface with notification service
SW.ALUBE.4	Backend Application	Notification application - handling of incoming notifications - distribution of notification events to subscribed users
SW.ALUBE.5	Backend Application	Electronic Health Record database - storage of all medical history related to the patient
SW.ALUBE.6	Backend Application	Stock Monitor Application - receive events from RFID reader - check RFID tag against tag database - update patienherEHR - interface with notification service
SW.ALUBE.7	Backend Application	RFID Database - database with all metadata corresponding an RFID tag
SW.ALUBE.8	Mobile Application	Stock alarm mobile application - display incoming events related to low insulin stock
SW.CATTID.1	Backend Application	Software at the hospital which: monitors the environmental conditions in the room and the conditions o patients' and controls the medications to the patients to avoid wrong medication.
SW.CATTID.2	Program on Sensor Node	Software which senses specific sensor limits and sends out alarms
SW.CATTID.3	Program on Sensor Node	Software which senses specific sensor limits and sends out alarms
SW.CATTID.4	Backend Application	Automatic analysis of submitted measurements on tablet
SW.CFR.1	Backend Application	Patient Care Web Application - database with patients to monitor - web interface - map integration
SW.CFR.2	Mobile Application	Android App for patient to interact with their WBAN - frontend for SW.CFR.1 - gateway to sensors
SW.CFR.3	Mobile Application	Android App for physician to interact with patient data - frontend for SW.CFR.1 - gateway to sensors
SW.CFR.4	WBAN firmware	Embedded software to control sensor HW and provide services to SW.CFR.1/2/3
SW.CFR.5	Gateway firmware	Embedded software to bridge WBAN with Internet using either Ethernet or WiFi, depending on the deployment
SW.HSG.1	Local Hospital Software	Software for processing ID card scanning, and calls a service which seeks permission for retrieving information about scanned human. heral EHR is updated from the results of the service call
SW.HSG.2	Local Pharmacy Software	Software for checking-out pharmacy goods like ampoules, and can look-up any buying

Application ID	Category	Description
		restrictions associated with the sold product. Also executes processing ID card scanning, and calls a service which seeks permission for retrieving information about prescriptions of the shopper.
SW.IML.1	Mobile Application	Android App for Logistics (Loading, Transport notification, Unloading, Handover).
SW.IML.2	Manufacturer Backend System	ERP system at manufacturer - retrieve orders - manage truck driver dock door handling - forward orders to truck driver - link products with load carriers
SW.IML.3	Store Backend System	ERP system at store - create order for products - handle delivery notes
SW.IML.4	Program on sensor node	Software which senses specific sensor limits and sends out alarms.
SW.IML.5	Terminal Application	Application running on the NFC device. Hosts the CheckIn service.
SW.SAP.1	Mobile Consumer Application	Android App for the consumer that comprises product information services, shopping lists, and other value added services.
SW.SAP.2	Store Digital Signage	ERP system at manufacturer - retrieve orders - manage truck driver dock door handling - forward orders to truck driver - link products with load carriers
SW.SAP.3	Instore Positioning Software	Locate user activities in different zones of the retail store
SW.SAP.4	Campaign Mangament Software	Used to calculate prices based on real-world environmental parameters
SW.TID.1	Mobile Application	Android App. to collect info for each product and send it to the C.TID1 Resource Story Storage

Table 21: Application list of components planned for use cases implementation

6.2Planned Devices

The following table lists the hardware modules and components which are planned to be used on implementation in both health and retail use case in no specific order.

Device ID	Category	Description
HW.ALUBE.1	Smart Phone	TazPad Android tablet with - NFC - Zigbee interface - Biometric sensor (fingerprint reader)
HW.ALUBE.2	Zigbee device	Cleode Zplug: - smart metering switch - used to turn on/off a light
HW.ALUBE.3	Light	Light that is switched on/off by HW.ALUBE.2
HW.ALUBE.4	Bluetooth device	A&D wireless blood pressure meter
HW.ALUBE.5	Bluetooth device	A&D wireless weight scale
HW.ALUBE.6	Bluetooth device	Entra wireless blood glucose meter
HW.ALUBE.7	Zigbee device	Cleode Zcare medical bracelet with

Device ID	Category	Description
		- Fall detector - panic button - pulse sensor
HW.ALUBE.8	RFID reader	NXP reader
HW.CATTID.1	Computer	Runs HIS
HW.CATTID.2	Sensor Node	Body Temperature Sensor
HW.CATTID.3	Sensor Node	Room Temperature Sensor
HW.CATTID.4	Tablet	Runs mobile HIS and Tag Reader
HW.CATTID.5	RFID Tag	Patient Bracelet
HW.CATTID.6	Gateway	IEEE 802.15.4 -to- Ethernet
HW.CFR.1	Sensor Node	Constrained device with IEEE 802.15.4 RF and 6LoWPAN/CoAP protocol stack Provides lightweight webservices using CoAP
HW.CFR.2	Gateway	Custom device connecting the IEEE 802.15.4 devices with the Internet. Depending on the final implementation might also host SW.CFR.1
HW.CFR.3	Server	PC hosting SW.CFR.1 if HW.CFR.2 has limited capabilities
HW.CFR.4	Android device	Either a smartphone or a tablet with up to date Android OS, running SW.CFR.2/3
HW.HSG.1	IoT-Mouse	Mouse with multiple sensors
HW.HSG.2	Local Computer	Laptop/desktop running local hospital software
HW.HSG.3	Tagged ID Cards	ID cards with NFC tags on them
HW.HSG.4	Tagged Ampoules	Ampoules with NFC tags on them
HW.IML.1	Smartphone	Used by Ted to check in at manufacturer and store, for loading, monitoring the transport goods and the digital signature in the store.
HW.IML.2	Smartphone	Used by John to do the quality check and the digital signature with Ted.
HW.IML.3	Sensor node	Device which takes the measurements inside the load carrier.
HW.IML.4	RFID/Barcode-Label	Tag which identifies the load carrier.
HW.IML.5	RFID gate	Reads the load carrier tags while unloading the load carriers at the store.
HW.IML.6	NFC Device	Device for handling the CheckIn at the manufacturer. Acts as an active NFC terminal.
HW.IML.7	RFID label	Location tag store. Is applied on a passive terminal.
HW.IML.8	Gateway	Gateway which is communicating with the sensor nodes inside the truck.
HW.SAP.1	Tagged Cheese	RFID-tagged cheese to be read with mobile phone NFC reader
HW.SAP.2	Consumer Mobile Phone	Used by the consumer in the store in order to read NFC tags and to run the SW.SAP.1 software component
HW.SAP.3	Advertisement Display	Hardware device that runs the digital signage software SW.SAP.2 and reacts with different contents to changes in consumer locations in the store
HW.SAP.4	Location Sensing Technology	The hardware equivalent to the SW.SAP.3 software that is used to determine positions of consumers in the store.
HW.SAP.5	Electronic Shelf Labels (ESLs)	Dynamically adaptive displays in a store that can update price changes in real time.
HW.SAP.6	Sensor Devices (MoteRunner)	Wireless devices used for measuring temperature and humidity of products.

Device ID	Category	Description
HW.TID1	IoT-Phone	Used in scene 1 to gather info for each product

Table 22: Device list of components planned for use cases implementation

6.3 Identified Functional Components

The following table lists common software components and services which are planned to be used on implementation in both health and retail use case in no specific order.

Component ID	ARM Functional component	Description
C.ALUBE.1	Virtual-entity resolution	Draft implementation of the VE Resolution functional component
C.ALUBE.2	IoT Service Resolution	Draft implementation of the IoT Service Resolution functional component
C.CATTID.1	Business ModelProcessing	Hospital Information System (HIS)
C.CATTID.2	Business ModelProcessing	Tablet Application of Hospital Information System (HIS)
C.CFR.1	Device connectivity and communication	Communication Unification
C.CFR.2	Device connectivity and communication	Device Interoperability
C.CFR.3	Device connectivity and communication	Communication Reliability
C.HSG.1	AuthN	Draft implementation of the Authentication functional component
C.HSG.2	AuthS	Draft implementation of the Authorization functional component
C.SAP.1	Process Execution	Draft implementation of the IoT augmented Process Execution functional component
C.TID.1	Resource History Storage	Storage for storing information related to product's ownership and CO_2 footprint

Table 23: Components planned for use cases implementation

7. Conclusion

The use cases presented in this deliverable demonstrate the evolutionary approach that work package 7 has taken in the development and specification of the core tangible results of the IoT-A project. While the initial definition of the use cases as outlined in D7.1 (Hagedorn, et al., 2011) primarily served as a collection of rather unrelated ideas, a convergence towards a coherent and domain spanning storyline could be reached in this deliverable.

This convergence manifests itself both within each domain and at the interfaces between the domains. While health and retail naturally represent very different, but each in its own right highly relevant fields that profit with IoT technologies we even provide some basic domain spanning integration that contributes to demonstrating the value of the Internet of things paradigms.

Due to the progress that the project has made during the last year, this deliverable D7.2 also shows how integration between the different work packages of the project is taking place and is progressing continuously. This becomes most obvious by the relationships to the domain model that are presented for each and every scene of the document. This domain model mapping ensures that the use cases are highly integrated with the overall vision and architecture of the project. The use of domain model related figures demonstrates that what we show in this document - and with the realization of the use cases - is the manifestation and implementation of the reference architecture.

Correspondingly, most functional components of the reference architecture are actually used within the scenes of the use cases. In that respect, this deliverable also demonstrates that the applications, devices, and functional components are closely linked to the other work packages, as, for instance, virtual entity resolution as a functional component is provided by work package 4 and is used by practically every scene detailed in this document. Also, the process execution of work package 2 will be used in some of the retail scenes, although to a lesser degree. First implementations of some of the scenes already utilize hardware provided by work package 5 such as mote devices powered by the Moterunner platform.

In addition to these obvious links to the contributions of the domain model and the reference architecture as well as the individual technical work packages, the role of security and privacy takes a special focus of the document, as it is relevant as a vertical function throughout the entire project. Consequently, this deliverable dedicates an entire chapter to security and privacy considerations.

While the first deliverable D7.1 (Hagedorn, et al., 2011) of the work package had a focus on demonstrating the added value of the use cases from a business and process perspective, this justification was omitted from this document, as the rationale behind it has not changed and would detract from the new information provided. Instead, sufficient implementation details are systematically provided in order to serve as a reference frame for the distributed work among partners of implementing the use cases. From that perspective, D7.2 can be seen as a hinge between the more abstract and less focused initial definition of the use cases provided in D7.1 and the actual implementation of the use cases that follows the delivery of this document.

In order to ensure the external validity of the use cases presented, each scene comes with a short section that outlines the key benefits, central concepts, and technologies used. This kind of executive summary is also reflected in a corresponding presentation and slides set that will be presented to the IoT-A stakeholder group at the forthcoming stakeholder workshop. This process was initiated in order to ensure that prior to the finalization of the use cases, the domain specific experts within the stakeholder group have the chance to comment on and improve upon the selected scenes, so that the key objectives and benefits of the scenes can be identified also from a domain- and not only from a technical perspective. This iterative approach is a key pillar of the entire IoT-A project and is naturally also reflected in this deliverable.

Our current plans for implementation includes the preparation of the IoT Week conference in summer 2012 where we seek to gather feedback from expert visitors and other important stakeholders about first prototype implementations that already show some of the concepts discussed in this document. In parallel to that, we will also conduct a dedicated stakeholder workshop that will be more structured towards a systematic analysis of the selected scenes. After careful evaluations of the feedback we will get at IoT Week and the stakeholder workshop we will finalize the demonstrations of both storylines in appropriate locations such as a simulated retail store or a hospital. With these implementations we hope to be able to prove the value of a domain spanning reference architecture for the Internet of Things, both from a technical and from a business perspective.

8. References

Hagedorn, P., Magerkurth, C., Meyer, S., Haller, S., Sperner, K., Ho, E., et al. (2011). *Project Deliverable D7.1 - Initial definition of Use Cases 1/2.*

Nettsträter, A., Bauer, M., Bui, N., Carrez, F., Giacomin, P., Haller, S., et al. (2011). *Project Deliverable IR1.4 – Initial Architectural Reference Model for IoT.*

Savry, O., & Moering, W. (2011). *Project Deliverable D5.1 - Energy aware IoT architecture.*

van Kampen, P., & Savry, O. (2011). *Project Deliverable D5.2 - Secure and Privacy compliant interactions with IoT objects: Concept study and architecture.*

Walewski, J. W., Bauer, M., Bui, N., Giacomin, P., Gruschka, N., Haller, S., et al. (2011). *Project Deliverable D1.2 – Initial Architectural Reference Model for IoT.*

A. Appendix

The Appendix includes a loose collection of diagrams which were already made regarding the upcoming implementation of both use cases. The intention is to give the reader a more in depth view on the details of the single scenes.

A.1 Health Use Case Diagrams

A.1.1 Scene 1 – Remote Patient Notification (ALU-BE)

The interaction diagram for Scene 1 can be found in Figure 30.
In a first step, the Patient Care BE application wants to sound the alarm on the IoT-Phone of Robert. It needs the service URL of the Alarm application to do so. Since the Alarm service is associated to the VE of Robert, the backend application first performs a lookup of this association. The result of this lookup is a Service ID. This Service ID can then be resolved to a URL and the alarm service invoked.
In the use case the alarm is not acknowledged so the backend application will look for lights in the vicinity of Robert's last known location.
Robert's VE is associated to a service which exposes Robert's location. This service gets its data from the GPS receiver on Robert's IoT-Phone.
So first the association between Robert's VE and the location service is determined. The result is a Service ID which can then be resolved again to a URL.
Now that the location of Robert is determined, lights in the vicinity can be discovered with the discovery service of the resolution framework. This service accepts a Service specification where it will be indicated that all service of type Light within a certain perimeter of Robert's last location need to be found. The result of this call is a list of Service Descriptions (which include the URL of the service). The backend application then selects the best match and invokes the Light service switching on the light.
In a last step, Robert physically sees the light, looks for his IoT-Phone and acknowledges the alarm.

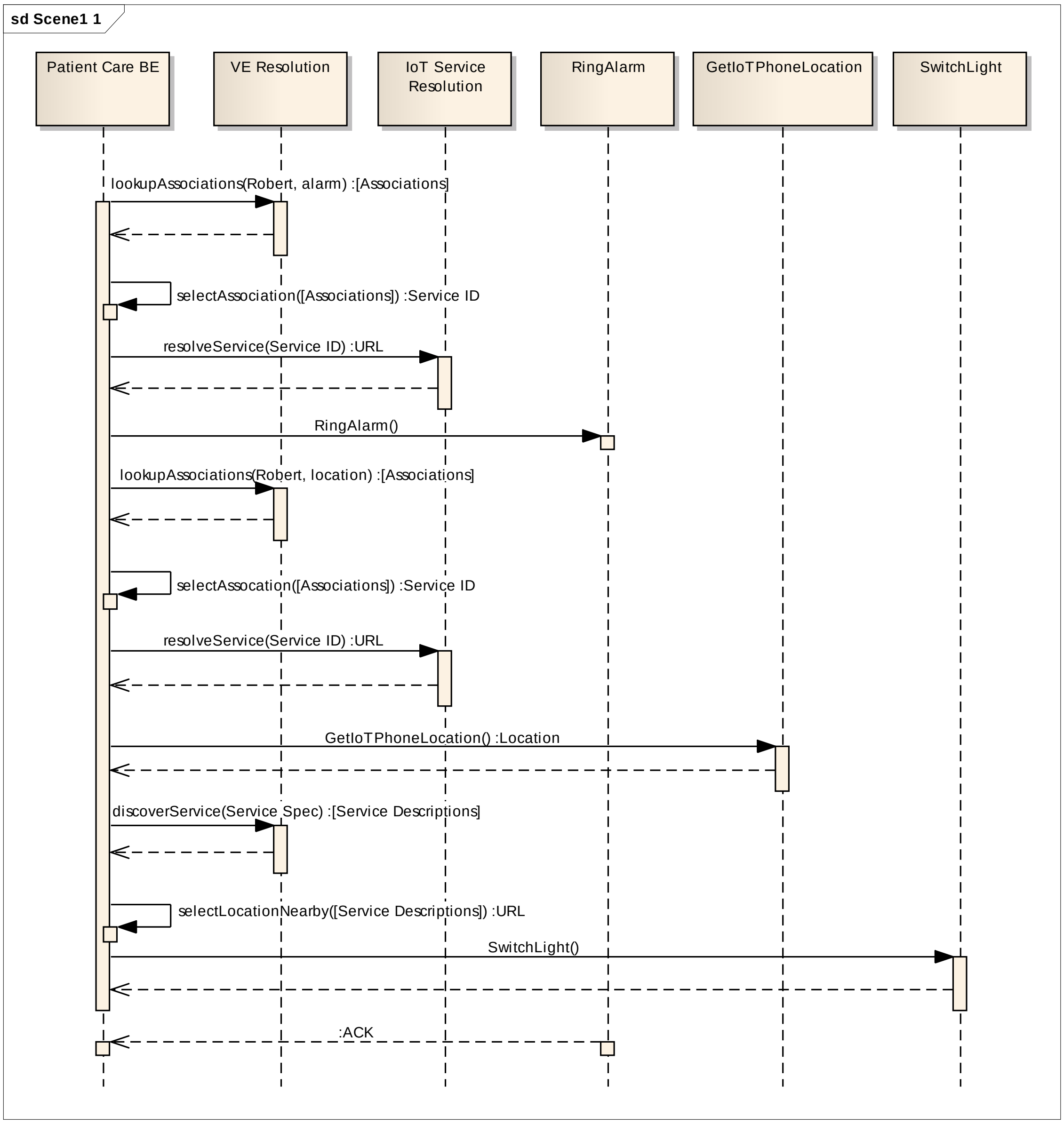

Figure 30: Interaction diagram of Scene 1

A.1.2 Scene 2 – Remote Patient Measurements (ALU-BE)

The interaction diagram of Scene 2 is depicted in Figure 31.
This interaction diagram is for worked out in detail for taking a weight measurement. Other measurements such as blood glucose level and blood pressure follow the same flow.
The general behaviour is that a certain service, such as the weight service is associated with the patient's VE. In case the backend application wants to take a weight measurement, it will invoke that service. The application behind the service will then start an application on the IoT-Phone guiding the patient in taking the measurement.
In a first step, the Patient Care BE application will perform a lookup for the association between Robert and a weight service. In case several services are found, the best match will be selected and the Service ID will be resolved to a URL.
The Patient Care BE application will then invoke the service on the IoT-Phone using that URL.

Following this invocation, the IoT-Phone will pop up a GUI and guide the user through the measurements. The service will return the value of the measurement to the backend application.
Now that the value is received, the backend application needs to identify the patient to make sure that no confusion between patient data is possible.
The backend application will look for the fingerprint service associated with our patient and invoke its service. To do that, it needs to lookup the association and resolve the service ID similar as explained above.
The fingerprint service will receive a service call and pop up an application on the IoT-Phone asking for confirmation of the data.
The patient confirms by putting his finger on the fingerprint reader.
The backend application will receive the biometric data from the IoT-Phone and verify it with the biometric data stored in the EHR of the patient.
In case both match, the backend application will store the measurement in the EHR of the patient.
There are some simplifications made in the interaction diagram. For instance, since the EHR is a network resource, each record in the EHR is a separate resource and has a dedicated service. This is easy implementable using REST. Each of these service is then associated to Robert's VE.

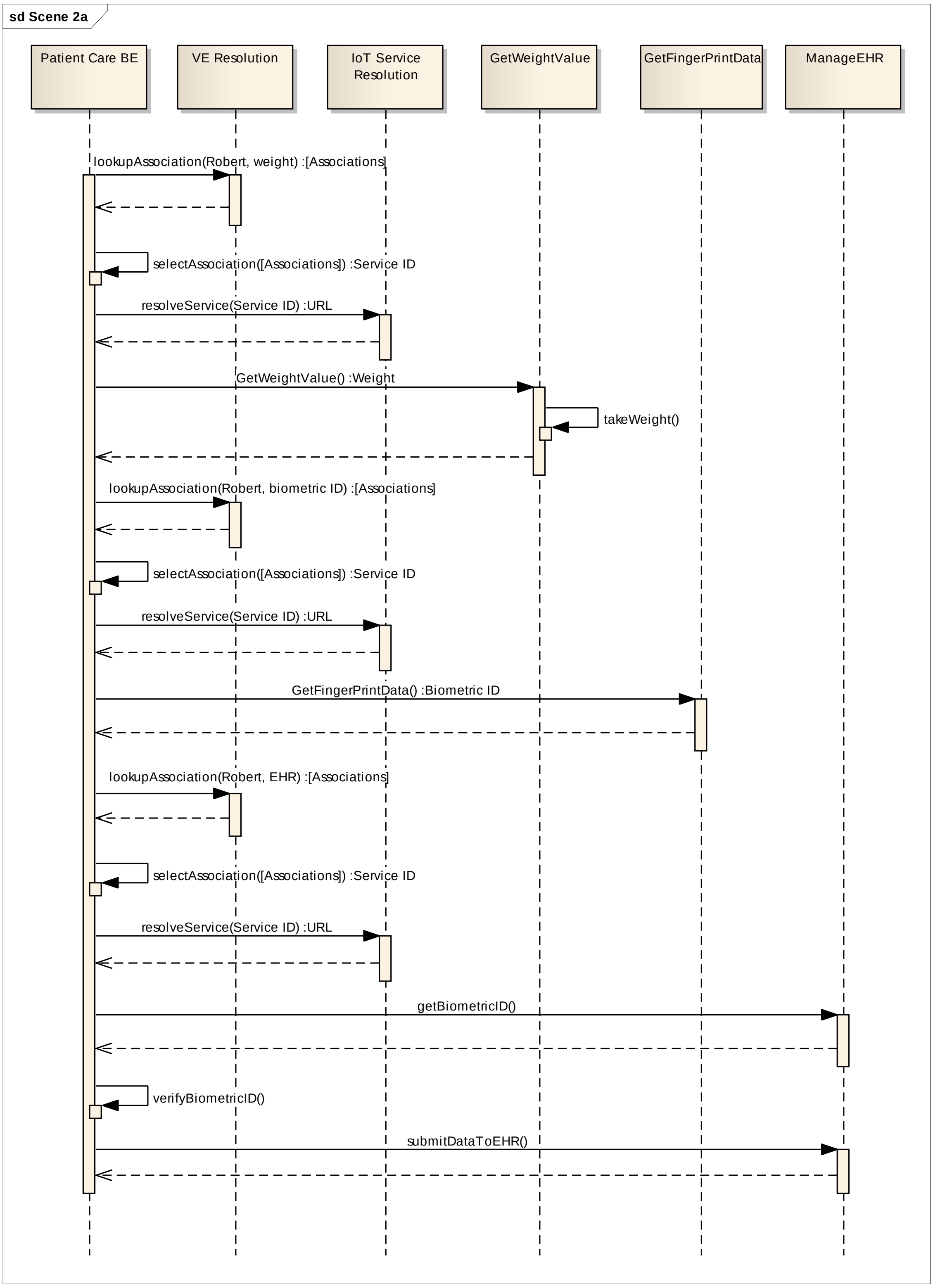

Figure 31: Interaction diagram of Scene 2 part 1

After the measurement data is submitted to the EHR, automatic analysis of the data is performed. This interaction diagram is depicted in Figure 32.

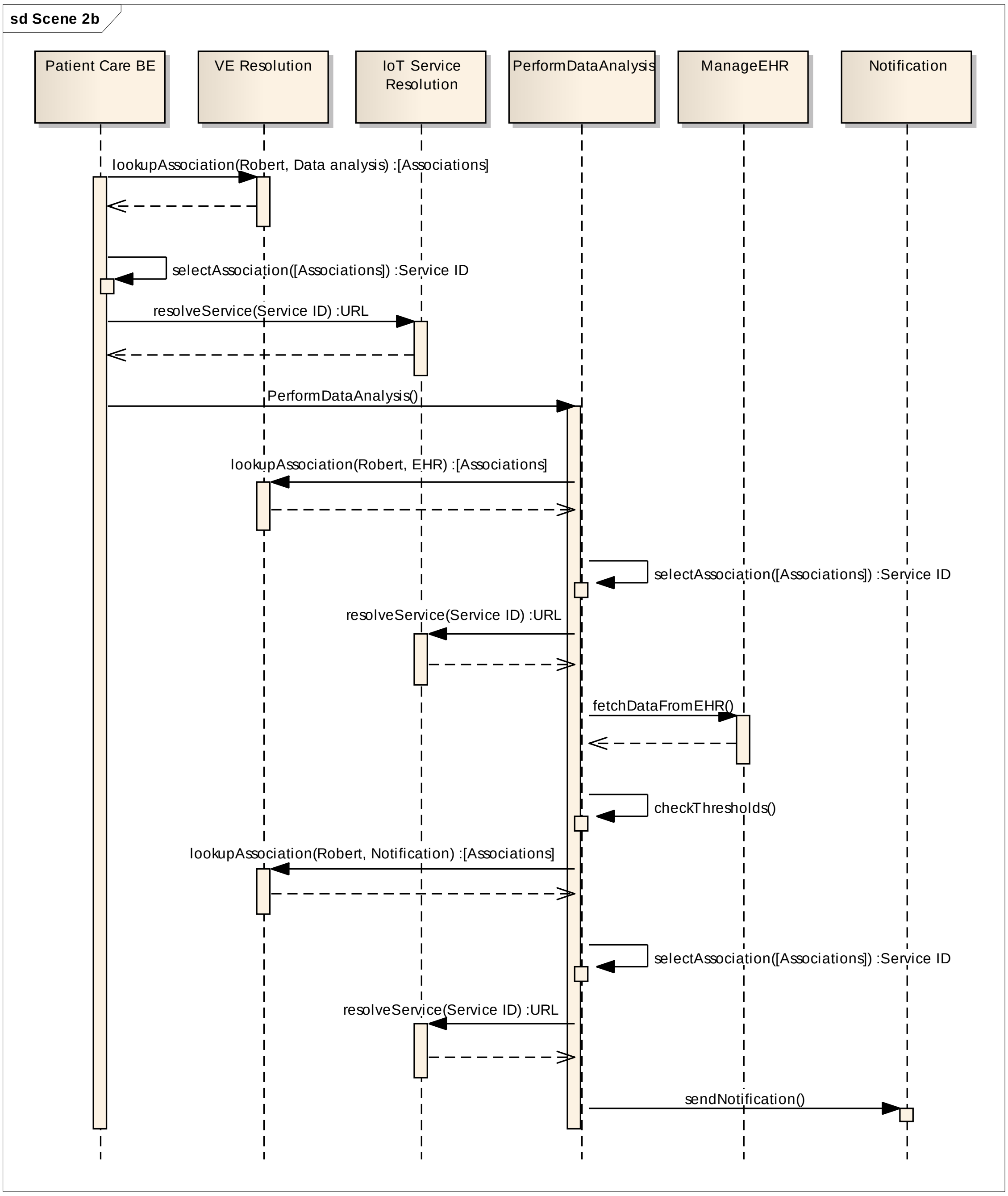

Figure 32: Interaction diagram of Scene 2 part 2

First, the Patient Care BE application will perform a lookup of the association between the VE of Robert and a data analysis service followed by a resolve service of the service ID. Next, the backend application will invoke the Data Analysis service to start the data analysis on the data of the EHR of Robert. The Data Analysis App will perform a find the EHR service associated to Robert and fetch the EHR data. Next the application will check if the data of the measurements is within the defined thresholds. In case not, the Data Analysis application will look for the Notification service (associated to Robert) and invoke it. The Doctor, which is subscribed to this Notification service will then receive a notification that the EHR data of Robert need attention.

A.1.3 Scene 3 – Insulin alarm (CFR)

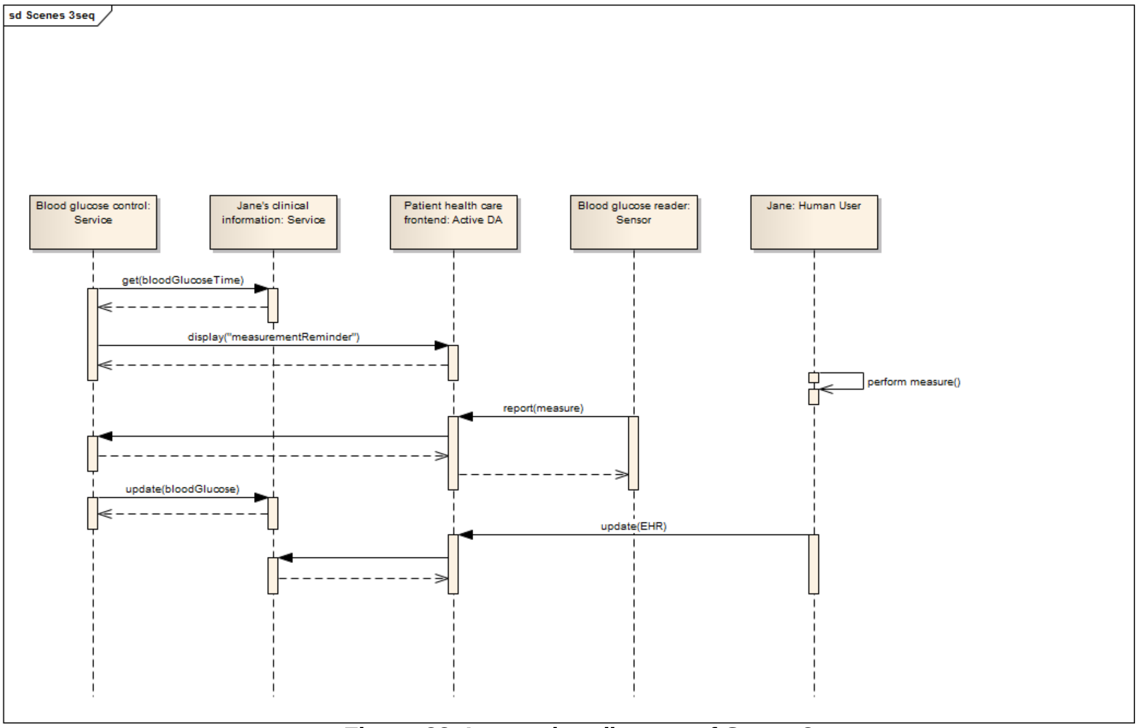

Figure 33: Interaction diagram of Scene 3

A.1.4 Scene 4 – Low Insulin Supply (link to Retail UC) (ALU-BE)

The interaction diagram for scene 4 is depicted in Figure 34.
The Stock Supply Monitor application receives an event from a RFID reader (not modeled) that a certain RFID tag is removed. This event starts the sequence diagram.
First, the application wants to access all information of this tag. So it performs a lookup of the associations between the VE of the tag (actually, the tag ID) and its network resource, select the correct association followed by the lookup of the URL of the service giving access to the network resource.
In the network resource, all details of the RFID tag are present.
From the description returned by the network resource, the application knows that the tag represents and ampoule of insulin.
The Stock Supply Monitor will now update the EHR of Robert. First, the application will look up the association between Robert's VE and an EHR service. Next, it will select out of the list of associations the correct association and resolve the Service ID to its URL.
The application will then submit to the EHR that an ampoule of insulin was removed from the medicine cupboard. The EHR application will update the EHR of Robert and return the amount of insulin still left.
The Stock Supply Monitor application will now check the amount of insulin left against a threshold. In case the supply is too low, a notification must be send so the application will look the associations between Robert's VE and a notification service, select the correct association and resolve the corresponding Service ID to its URL.
Finally, the application can send a notification to which the application on Salomée's IoT-Phone is subscribed.

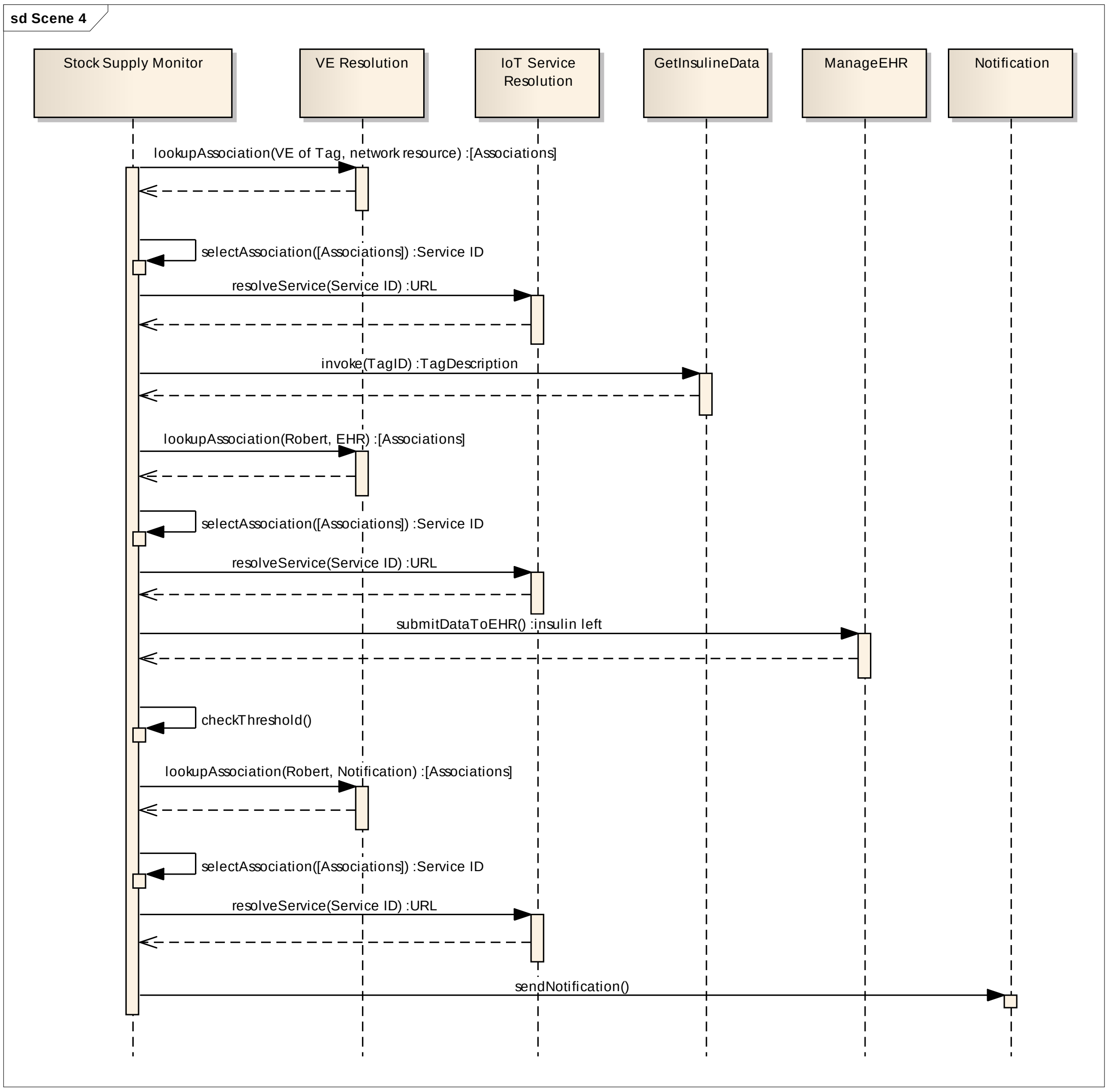

Figure 34: Interaction diagram of Scene 4

A.1.5 Scene 5 – Remote Patient Panic Event (ALU-BE)

The first part of the interaction diagram for Scene 5 can be found in Figure 35.
At startup, the Patient Care BE application will subscribe to events of the panic button service related to Robert. So first, it will lookup the associations between Robert's VE and the panic button service and select the correct association. Next, it will resolve the Service Id to the URL and send a subscribe message to the service.
If Robert presses the panic button, the Patient Care BE application will receive an event. The application will now check the pulse of Robert. To do that, it needs the URL of the service so it will lookup first the associations between Robert's VE and pulse service, select the correct association and resolve the corresponding Service ID to the URL. The application will next obtain the pulse value from the pulse service and check if there is still pulse.

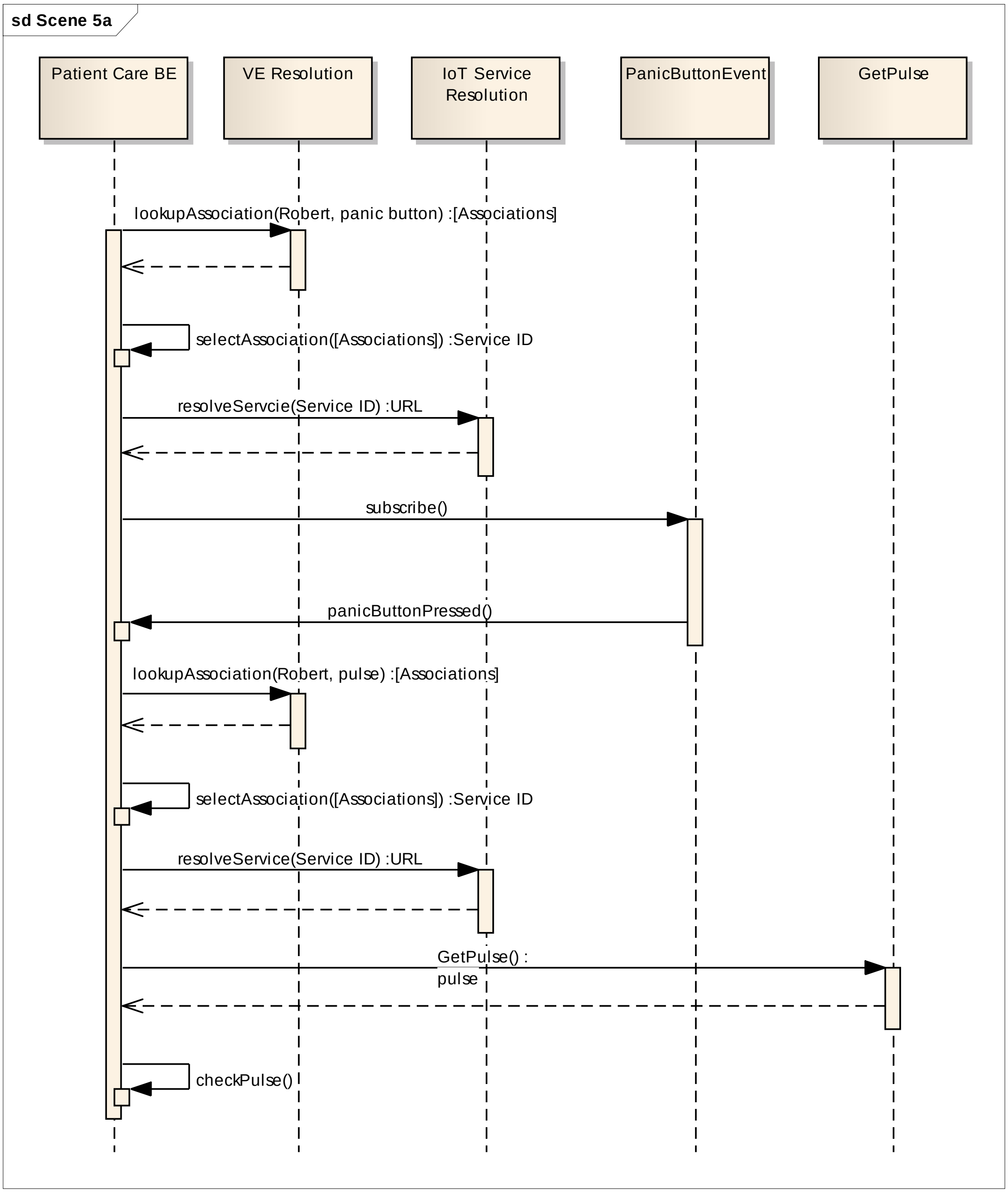

Figure 35: Interaction diagram of Scene 5 part 1

Since there is still pulse, emergency service are not necessary but Jane should be notified as can be seen in Figure 36.
The Patient Care BE application will obtain from Robert's EHR the first person to contact.
The application will therefore lookup the associations between Robert's VE and EHR services, select the correct association and resolve the corresponding Service ID to its URL.
The application can then obtain from the EHR the identification of the contact person.
Next, the Patient Care BE application needs to send a notification so it will lookup the associations between Robert's VE and notification services, select the correct association and resolve the corresponding Service ID to its URL. Finally, the Patient Care BE application sends a notification to the notification service to which Jane is subscribed.

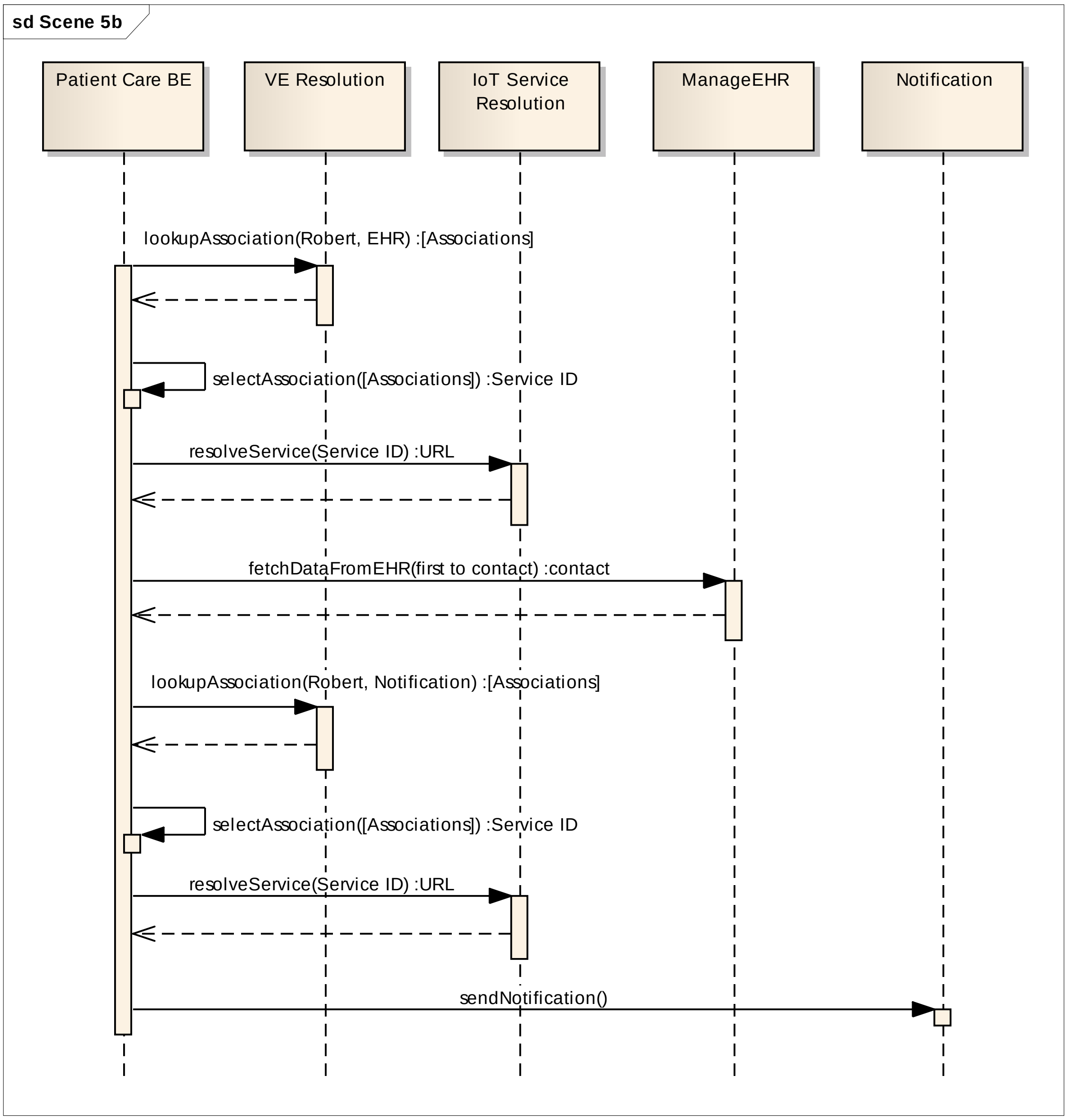

Figure 36: Interaction diagrams of Scene 5 part 2

A.1.6 Scene 6 – Car accident (CFR)

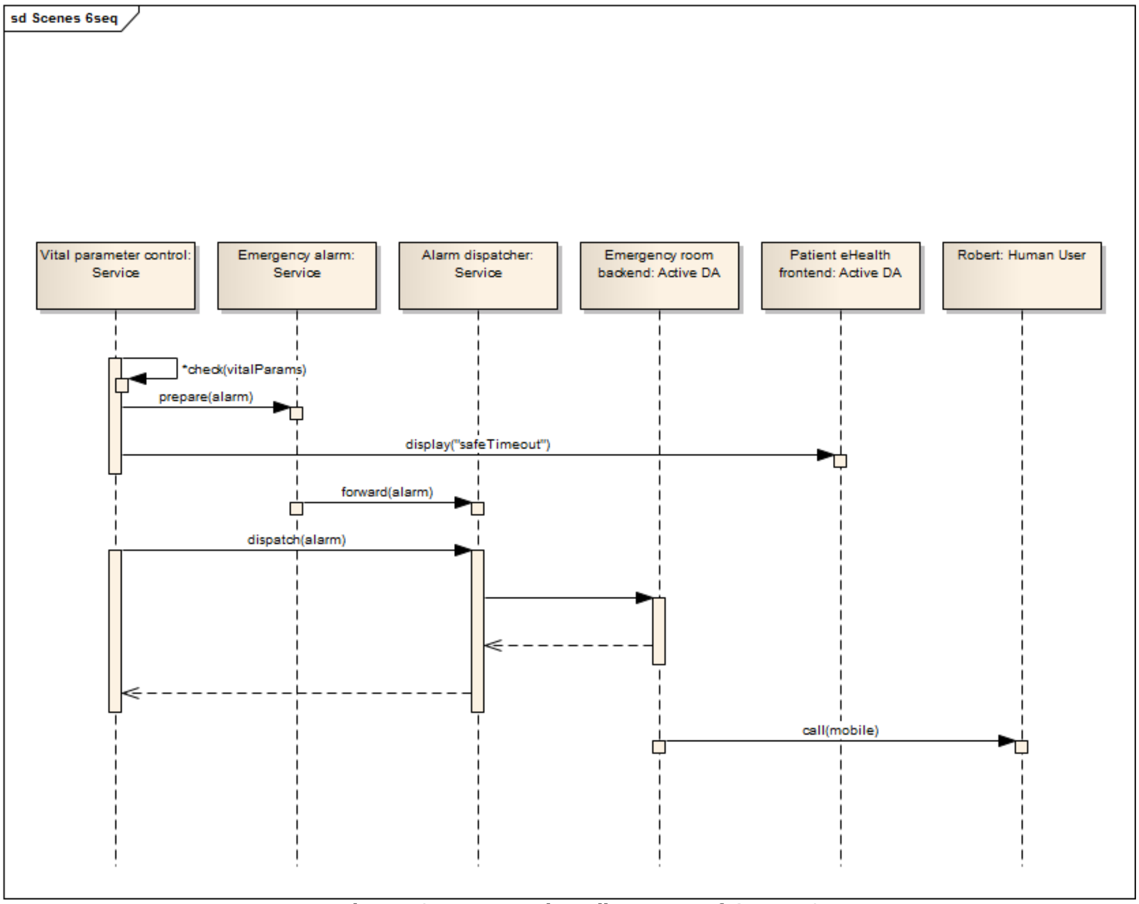

Figure 37: Interaction diagrams of Scene 6

A.1.7 Scene 7 – Expedited Checking Into a Hospital (HSG)

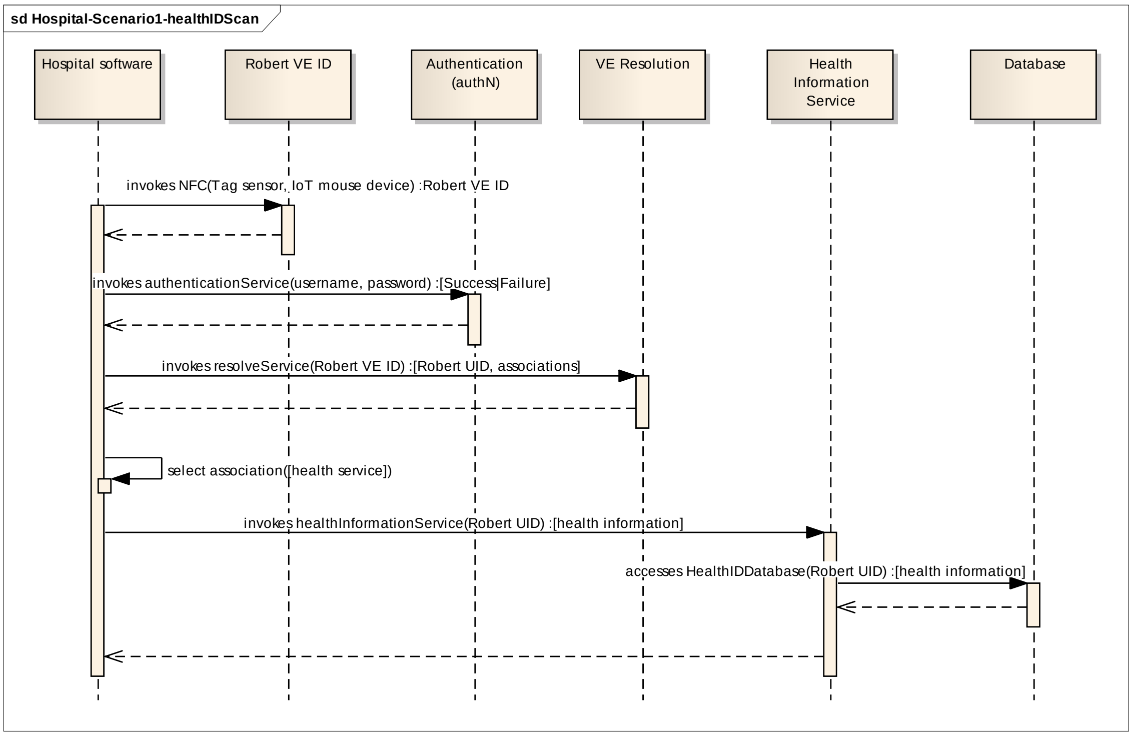

Figure 38: Interaction diagram of Scene 7, Health

The sequence of interactions can be described as follows. The Hospital user client uses the Hospital software. Then, the Hospital software subscribes to NFC service. The service listens to the NFC tags in the vicinity and communicates with them. These NFC tags are read using the IoT mouse which is operated by the hospital user client. NFC tag reader is integrated in the IoT mouse. NFC tags are attached to the patients (Robert in our case) and contain the healthcare information of the patients. After a NFC tag is read, hospital software calls the authenticationService to get the access to the database where information about the patient is stored. Once the client is granted the permission to access the database, a resolveService is executed to resolve the health ID (part of healthcare information read from the NFC tag) into a unique ID of the patient. The unique ID leads to fetch more information about the patient from different connected/distributed databases (e.g. driver license database, health database etc.).

Finally the healthInformationService is called to get all the healthcare information (address, information of the last visit, prescriptions, doctor details etc.). The database has all the relevant information about the patients. After getting the relevant information from the health database, hospital software displays it in the hospital GUI form.

A.1.8 Scene 8 – Environment and Patient Remote Monitoring (CATTID)

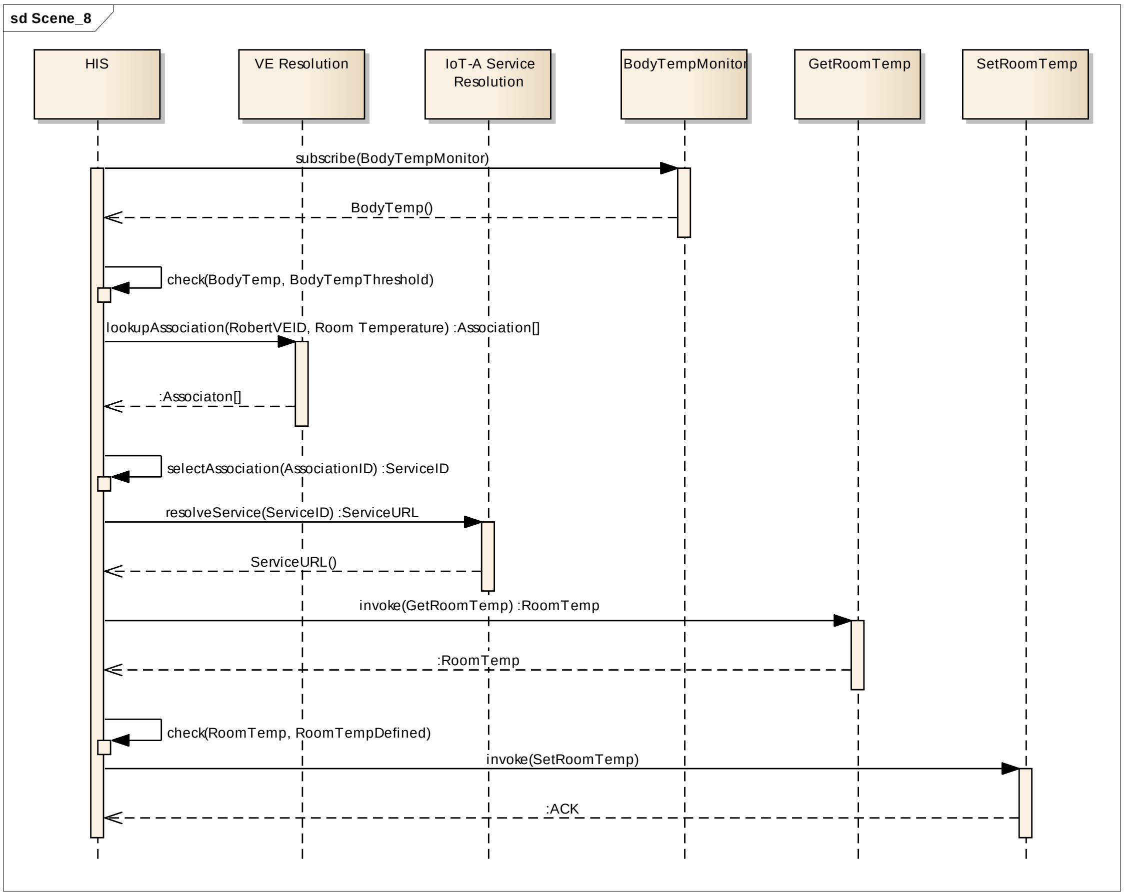

Figure 39: Interaction diagram of Scene 8

HIS is subscribed to Patient's body temperature measurement service and every result is checked if it is out of the defined threshold values. In the diagram, it is showed just for Robert but the same sequence is valid to obtain the fellow patient's body temperature.
Upon an excess of the defined body temperature, HIS starts to invoke the services to explore the problem which might cause the high temperature. GetRoomTemp is one of the defined services to be invoked. HIS lookup an association between RobertVEID and the room temperature service which is related with the Robert's VE. Association information includes the ServiceID, which is used to resolve the ServiceURL. HIS invokes the GetRoomTemp service to access the actual room temperature of Robert's room. After the retrieval of the data it is checked with the defined room temperature to explore if there is a problem. After the fix of the heating system problem by the responsible hospital staff, HIS invokes the SetRoomTemp service to regulate the room temperature to the defined value.

A.1.9 Scene 9 – Medication Control (CATTID)

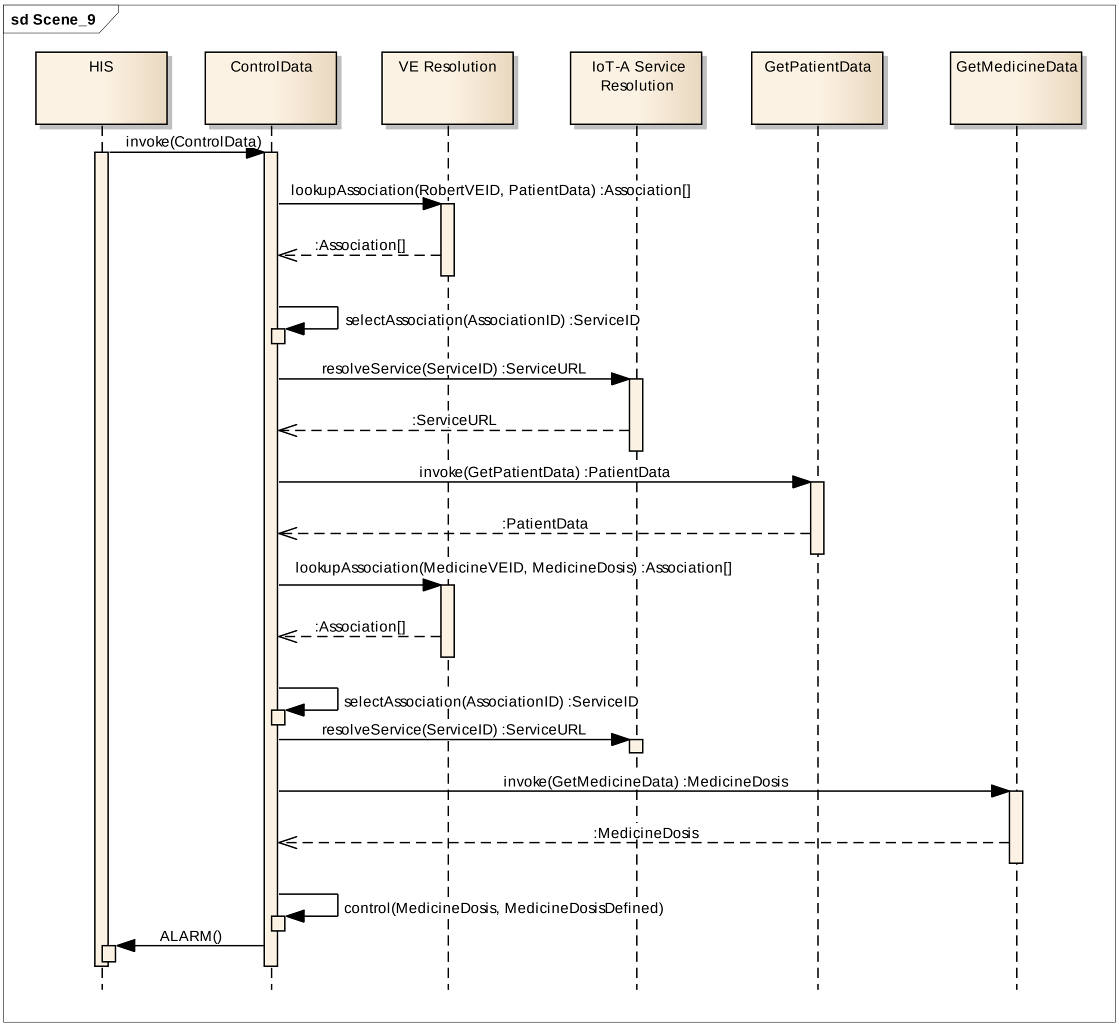

Figure 40: Interaction diagram of Scene 9

HIS has a ControlData service in order to prevent wrong medication of the patients. It is achieved through the control of defined patient medication information and actual data of the medicine which is prepared by the pharmacy of the hospital. Once the nurse has to administrate a medicine and chooses the related GUI from the HIS tablet application, ControlData service is invoked by HIS.

First, the nurse reads the tag of Robert through her tablet. ControlData service lookup an Association between RobertVEID and a service to access his registered data. After the resolution of ServiceID to ServiceURL, ControlData invokes GetPatientData to access the PatientData in the database. Afterwards the nurse reads the tag of the medicine which she brought with her to administrate to Robert. ControlData service lookup an Association between MedicineVEID and a service to access its dose information. After the resolution of ServiceID to ServiceURL, ControlData invokes GetMedicineData to access the MedicineDosis after the read of medicine ampoule’s tag.

ControlData service checks the registered medicine dose data of Robert and the actual dose of the medicine. It detects that the actual dose which is about to be administered to Robert is not the same as defined by the responsible doctor and alarms to the nurse.

A.2 Retail Use Case Diagrams

A.2.1 Scene 2 – NFC supported check in and assisted loading (FHG IML)

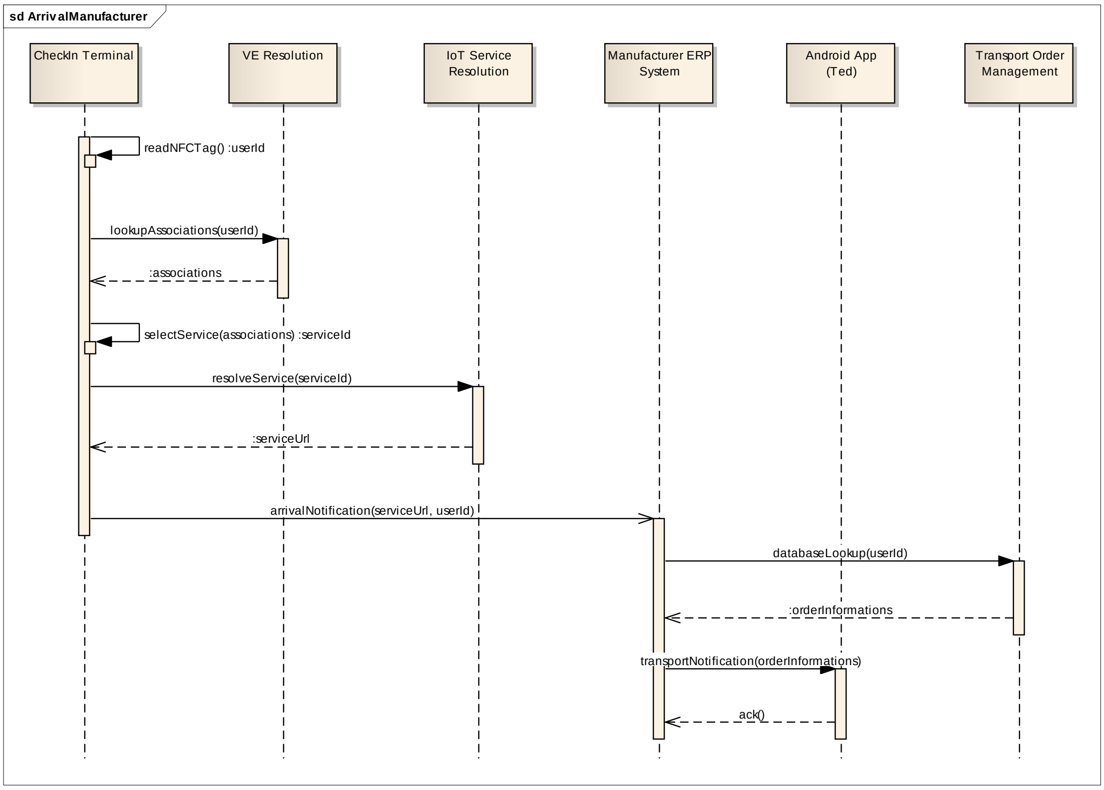

Figure 41: Interaction diagram of Scene 2 – Arrival at manufacturer

The Terminal scans the NFC tag of Ted's IoT-Phone to get his userId. After that it gets the corresponding associations over the resolution service. In the next step the transportNotification service is selected. The url of this service and Ted's userId is sent to the Manufacturer ERP system which does an lookup for with Ted associated transport orders. After all information (gate number, delivery note and additional transport information) has been loaded they are sent back to Ted's IoT-Phone using the transportNotification service.

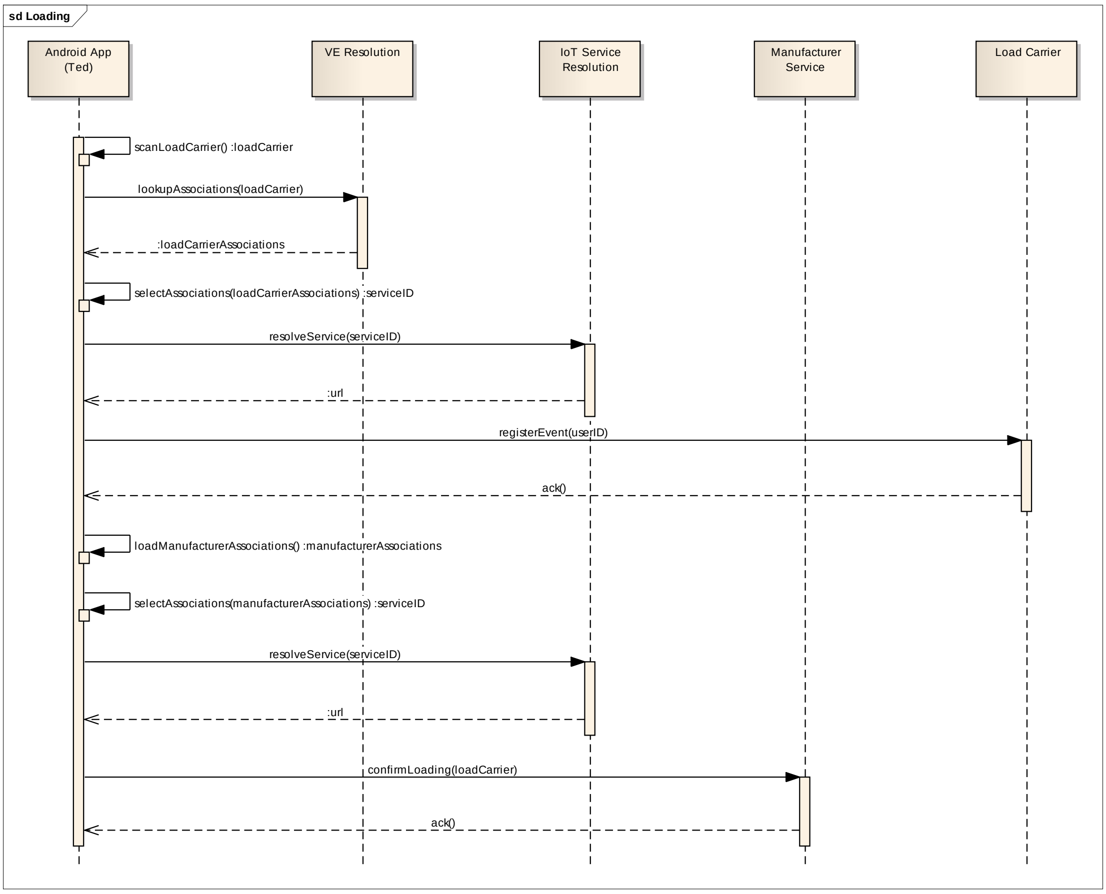

Figure 42: Interaction diagram of Scene 2 – Loading goods

By scanning a load carrier Ted's IoT-Phone gets access to the associated load carrier service which offers the registerEvent method. By invoking this method the IoT-Phone registers for the load carrier's sensor events for getting informed about any changes in temperature or humidity while the load carrier is transported by Ted.

In the second step Ted confirms the loading of the loaded carrier to acknowledge that the load carrier has been loaded successfully.

A.2.2 Scene 3 – Transport monitoring with Smart Load Carriers (FHG IML)

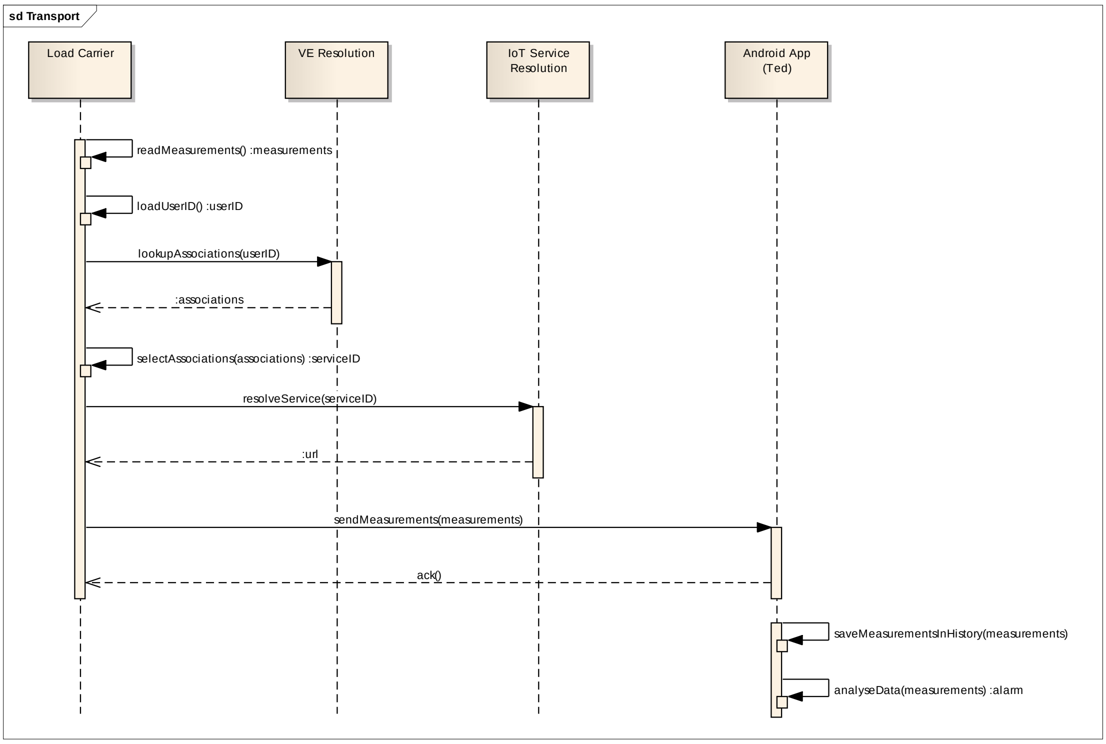

Figure 43: Interaction diagram of Scene 3 – Transport of goods

While the load carriers are transported by Ted, the embedded sensors continuously measure the environmental conditions within the truck. The measurement data is sent periodically over the sendMeasurements method of the android app running on Ted's IoT-Phone.
After responding this data the android app compares the measured parameters with the good specific environmental limits defined in the delivery note. If one limit has been exceeded an alarm is shown on the display of Ted's IoT-Phone.

A.2.3 Scene 4 – Assisted quality check and digital signature (FHG IML)

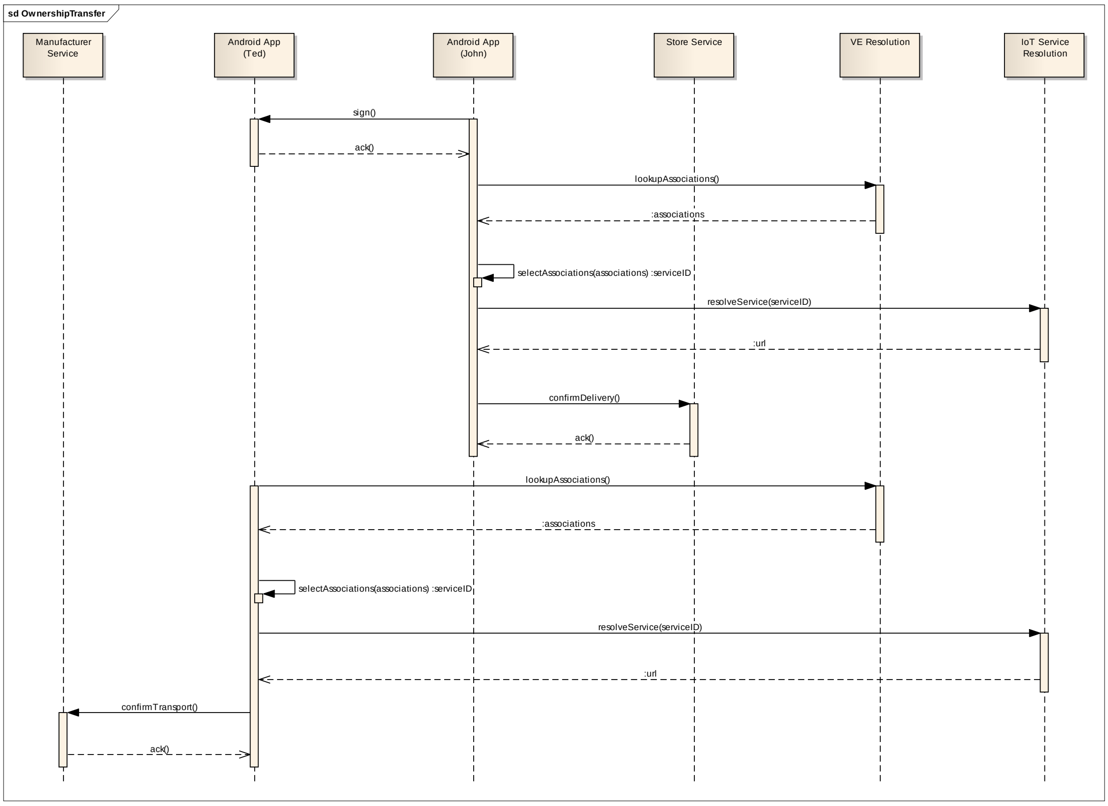

Figure 44: Interaction diagram of Scene 4 – Ownership transfer & digital signature

After John has checked the amount and quality of the delivered goods he uses the peer to peer functionality of his IoT-Phone to confirm the delivery digitally by sending a virtual signature directly to Ted's IoT-Phone.
After this John sends the delivery confirmation to the store's service by invoking the corresponding confirmDelivery method.
Ted confirms the successful transport by sending the digital signature of John to the manufacturer's service by invoking the corresponding confirmTransport method.

A.2.4 Scene 9 – The Selling of Controlled Substances at the Pharmacy (HSG)

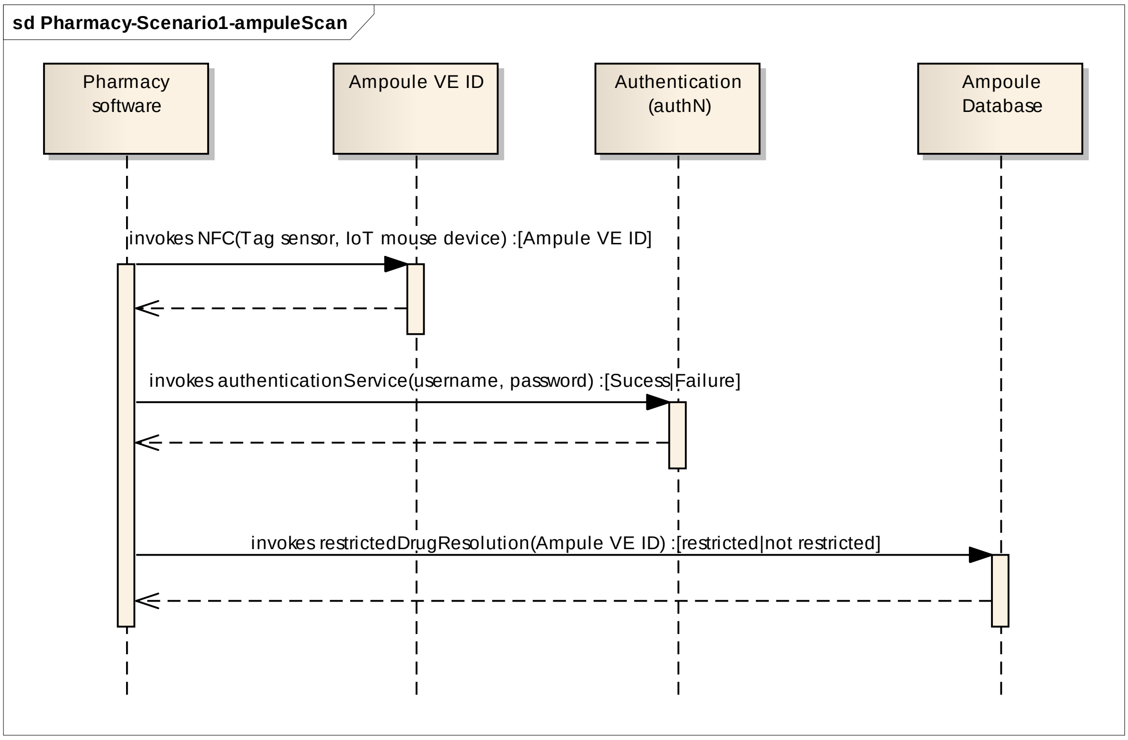

Figure 45: Interaction diagram of Scene 9a), ampoule scanning

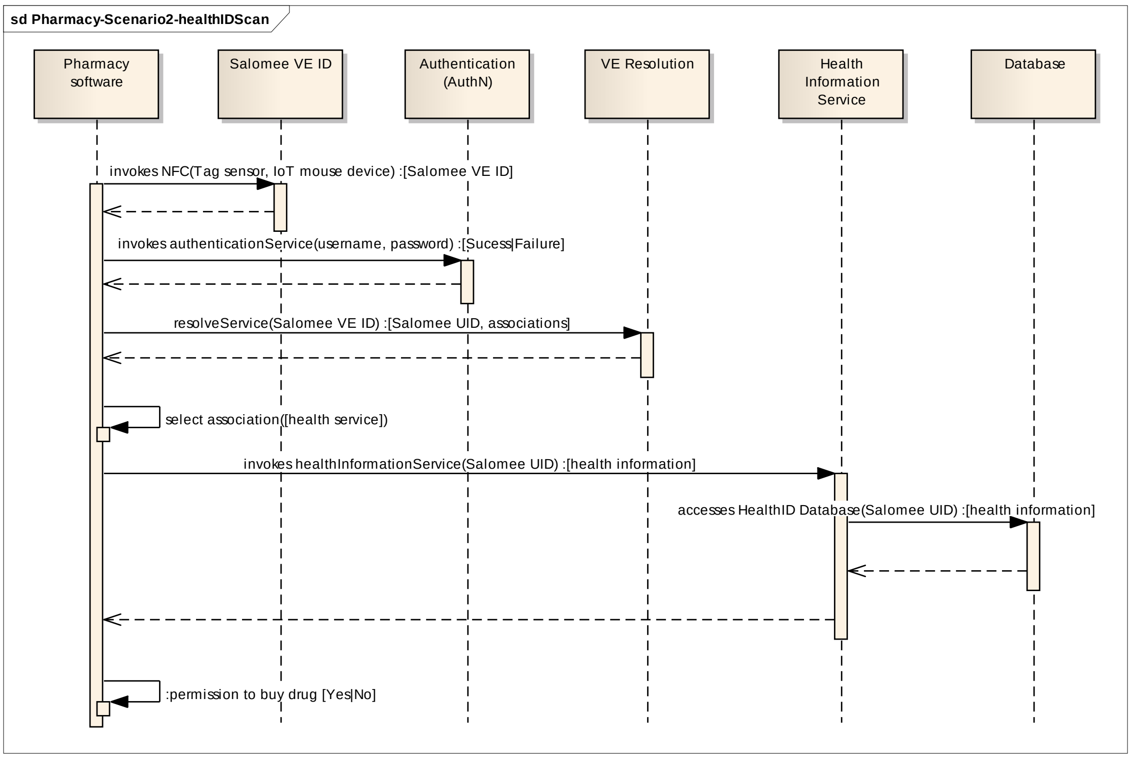

Figure 46: Interaction diagram of Scene 9b), health-ID scanning

In this scene, the sequence of interactions is as follows. Pharmacy user client uses the Pharmacy software. The Pharmacy software subscribes to NFC service. The service listens to the NFC tags in the vicinity and communicates with them. These NFC tags are read using the IoT mouse which is operated by the pharmacy user client. NFC tag reader is integrated in the IoT mouse. NFC tags are attached to the ampoules/pills and contain the information of the ampoules/pills.

After scanning an ampoule/pill, it is checked whether the ampoule/pill comes under restricted sale. If yes, then the pharmacy user client asks for a valid prescription and health information identification of the patient. This health information is provided as a separate NFC tag which is attached to the patient (Robert in our case) and contains the healthcare information of the patient.

After a NFC tag is read, the pharmacy software calls the authenticationService to get the access to the database where information about the patient is stored. Once the client is granted the permission to access the database, a resolveService is executed to resolve the health ID (part of healthcare information read from the NFC tag) into a unique ID of the patient. The unique ID leads to fetch more information about the patient from different connected/distributed databases (e.g. driver license database, health database etc.). Finally the healthInformationService is called to get all the healthcare information (address, information of the last visit, prescriptions, doctor details etc.). The database has all the relevant information about the patients.

After getting the relevant information from the health database, Pharmacy software displays it in the pharmacy GUI form.