Katholieke
Universiteit
Leuven

**Department of
Computer Science**

Shared Internet Of Things Infrastructure Platform:

Domain Analysis

Software Architecture (H09B5a and H07Z9a) – Part 1

Haveneers-Verreydt-Lemmens

Robin Haveneers (r0450702)
Stef Verreydt (r0456110)
Axel Lemmens (r0462440)

Academic year 2016–2017

Contents

1. Domain analysis

1.1 Domain models

This section shows the domain model(s).

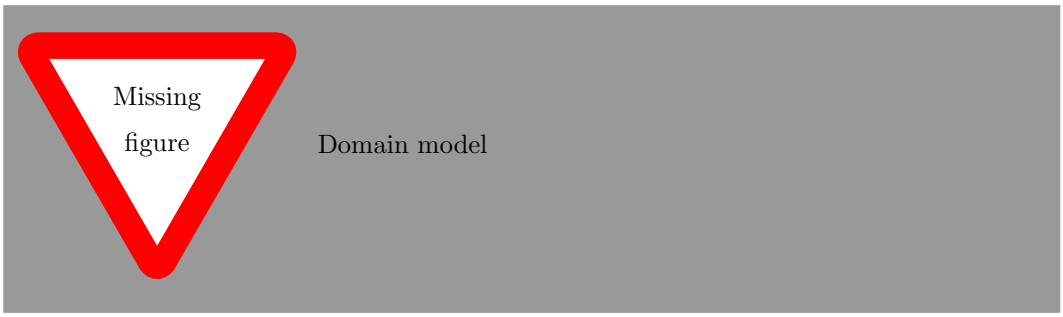

Figure 1.1: The domain model for the system.

1.2 Domain constraints

In this section we provide additional domain constraints.

- A customer organization can only subscribe to an application that is already registered with the system.
- A end-user only receives notifications from applications that the customer organization he works for is subscribed to. He only receives these notifcations if he/she is an individual who should receive notifications as configured in the application by the previously mentioned customer organization.
- The Online Service can only communicate with peripherals via their gateway.
- The connectivity between the network manager and the attached devices is done via 6LoWPAN.
- Application developers only receive rejection notifications about the applications they want to deploy.
- An application should be idle until all the necesarry hardware is available and if the hardware is available it should be activated automatically. For example, a fire application requires at least 2 specific sensors per room.
- If a component of the hardware fails, the system should automatically switch to a nearby, similar device if possible, based on the topology provided by the infrastructure owner.
- Only persons who are allowed to do so, can issue commands to an application.
- Gateways contain limited storage space. Whenever they're runnin low on space, the oldest information should be removed.
- When communicating with the system, customer organisations, infrastructure owners, system administrators and application providers each have their own, personal, dashboard to do so.

1.3 Glossary

In this section, we provide a glossary of the most important terminology used in this analysis.

- **API**: Application programming interface. The interface used by applications to communicate with the online system and gateways.
- **Accelerometer**: Type of sensor that measures the acceleration in 3 dimensions.
- **Actuator**: Hardware device that has actions associated with it (e.g. switch, light, buzzer ...)
- **Air Pressure Sensor**: Type of sensor that measures the air pressur in bar.
- **Air Temperature/Humidity Sensor**: Type of sensor the measures the air temperature in degrees Celcius and humidity in percentage.
- **Application Notification**: A type of notification send by the application to a relevant end-user.
- **Application Usage Information**: Data collected by applications and stored on the online service. Useful for providing insight in application usage and big data analytics.
- **Application provider**: Stakeholder that provides (develops and deploys) applications on the online system, to which one can subscribe.
- **Application**: Piece of 'software' running on the online service and gateway provided to customer organizations to subscribe to with the goal of simplifying their day to day business.
- **Building**: Place owned by infrastructure owner where customer organisations are housed. The infrastructure owners keeps a topology about the buidling they own.
- **Buzzer**: Actuator that is able to play a sound.
- **Connection Message**: THe message that is sent by the mote to the gateway whenever a new sensor or actuator is inserted.
- **Customer SLA**: The Service-Level Agreement between SIoTIP corporation and the customer organisation.
- **Customer organization**: The stakeholder in our system which subscribes to applications which optimize their workflow.
- **Data Message**: The message send by the mote from one of its connected devices to the gatewy which contains the value it's reading, the mote identifier and a timestamp. This message also specifies whether it was sent by a sensor/actuator which is necessary to be able to correctly interpret the data.
- **Development environement**: The environmennt available to application providers to freely test and debug their application before deploying it.
- **Distance Sensor**: Type of sensor which measure
- **End user**: A person involved in the customer organisation who interacts with applications.
- **Gateway identifier**:
- **Gateway**: A device that is connected to the online service and which provides access to sensors and actuators. It relays data from connected devices to the Online Service.
- **Hardware**: Umbrella concept used for all the connected devices (such as peripherals, gateways ...)
- **Heartbeat Message**: Periodical message sent by the mote to indicate it is alive. This message includes a list of connected sensors and actuators as well as a timestamp indicating when the message was generated.

- **Infrastructure owner**: Stakeholder in our system which manages the physical infrastructure as well as the hardware used as described in the toplogy. Customer organisations reside in these buidlings.
- **Library**: Set of methods that can be used by application providers in their applications to simplify development.
- **Light Sensor**: Type of sensor that measures the amount of light in LUX.
- **Llimited peripheral history**: Small set of peripheral history maintained by the gateway. Useful for application to, for example, learn certain behavourial patterns.
- **Message**: Umbrella term used for the different types of messages a mote can emit.
- **Microphone Sensor**: Type of sensor which allows detection of sound.
- **Mobile app**: One type of interface end-users can use to communicate with the online system.
- **Mote manufacturer**: Third party which produces the motes used by our system.
- **Mote**: Device that hosts sensors and actuators.
- **Notification**: Umbrella concept used for all the different types of notifications used by the system. Notifications are sent out on several events. The precise information depends on the type of event.
- **O_2 gas**
- **Online Service**
- **Online Service**
- **Passive IR**
- **Peripheral Data**
- **Peripheral History**
- **Peripheral type**
- **Peripheral**
- **Power socket**
- **Rejection Notification**
- **SIoTIP Corporation**
- **Sandbox environement**
- **Sensor**
- **Server provider**
- **Server**
- **Service SLA**
- **Service providers**
- **Synchronisation protocol**
- **System Administrator**
- **System Notification**

- **Telecom SLA**
- **Telecom operators**
- **Topology**
- **UI**
- **Web interface**

2. Functional requirements

Use case model

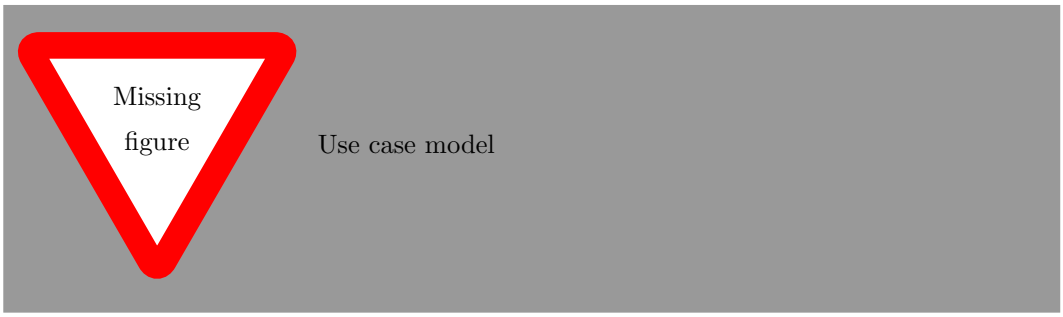

Figure 2.1: Use case diagram for the system.

2.1 Use case overview

UC1: Log in The user wishing to use the system provides his credentials. The system verifies the credentials and authenticates the user. If the provides credentials were not correct, the system does not authenticate the user.

UC2: Log off The user indicates he wants to log off from the system. The system logs him of.

UC3: Enroll as application provider The organization wishing to becoma an application provider contacts SIoTIP and they negotiate the contract. The organization reveices API documentation and rules of conduct from SIoTIP. When the negotiations are conducted, the organization is provided dashboard accounts for the individual developers employed by the organization.

UC4: Add application The user logs into the application provider dashboard and uploads the new application. SIoTIP initiates a number of automated tests and shows the progress on the user's application provider dashboard. If the application passes all tests, the user receives a notification and the application is made available to customer organizations for subscription. If the application does not pass al tests, a SIoTIP system administrator performs a secondary review and decides whether to accept or reject the application. In the latter case, the user is notified of the reason of rejection.

UC5: Update existing application The user logs into the application provider dashboard and uploads the updated application. The user also indicates whether to automatically update existing instances of the application or to require customer organizations to subscribe to the application. SIoTIP initiates a number of automated tests and shows the progress on the user's application provider dashboard. If the application passes all tests, the user receives a notification and the application is made available to customer organizations for subscription. If the application does not pass all tests, a SIoTIP system administrator performs a secondary review and decides whether to accept or reject the application. In the latter case, the user is notified of the reason of rejection.

UC6: Register as new infrastructure owner The infrastructure owner contacts SIoTIP and negotiations are started. An infrastructure owner dashboard account is set up for the new user. The infrastructure owner provides the names of the currently renting companies, and SIoTIP contacts them for registration (See UC: Register new customer organization).

UC7: Register as new customer organization The new customer organization contacts or is contacted by SIoTIP and provides it's billing and contact information.

UC8: Subscribe to application The user logs into his dashboard, subscribes to the application and provides the needed information. The user is informed that the application will be activated once the required peripherals are installed. SIoTIP checks whether or not the customer organization has access to all the peripherals needed for the application. If not, the infrastructure owner is automatically notified of the subscription and the needed peripherals (see UC9: process peripheral request). Once all required hardware is installed, the application is activated and the user is notified.

UC9: Process peripheral request The infrastructure owner is notified of a new request for peripherals. SIoTIP automatically adds sufficient gateways needed to support these peripherals to the request, if any. The infrastructure owner approves or rejects the purchase of the hardware and a notification of the decision is sent to the user which requested the peripherals. If the infrastructure owner approved the request, the hardware is ordered from SIoTIP.

UC10: Install new hardware The infrastructure owner receives the hardware from SIoTIP and installs it. The infrastructure owner configures any new gateways to connect to the (WiFi?) local network. Once online, the gateways immediately connect to the Online Service to register themselves. The infrastructure owner then logs in to the infrastructure owner dashboard and provides the necessary topology information. Lastly, he assigns access rights to the customer organizations in his building to use the new peripherals.

UC11: Resolve hardware failure The system detects that a hardware component is no longer sending data and sends a notification to the infrastructure owner. The system also notifies all applications currently using the failing hardware component so that the applications can search for equivalent sensors in the topology.

UC12: Configure end-users The user logs in to the customer organization dashboard and assigns other users to an application. Nodig?

UC13: Transmit data The sensor sends data to the corresponding gateway.

2.2 Detailed use cases

2.2.1 *UC1*: Log in

- **Name:** log in
- **Primary actor:** the User
- **Secondary actor(s)**: secondary actor(s)
- **Interested parties:**
  - *System:* wants to authenticate its users.
- **Preconditions:**
  - The User is registered into the system and has credentials to prove his identity.
  - Second precondition.
- **Postconditions:**
  - First postcondition.
  - Second postcondition.

- **Main scenario:**
    1. Step 1
    2. Step 2
    3. Step 3
    4. ...
- **Alternative scenarios:**

  3b. Alternative at step 3
- **Remarks:**
    - First remark

UC1: Name Short summary of this use case scenario

2.3 Detailed use cases

2.3.1 *UC1*: Log in

- **Name:** log in
- **Primary actor:** the User
- **Secondary actor(s)**: secondary actor(s)
- **Interested parties:**
    - *System:* wants to authenticate its users.
- **Preconditions:**
    - First precondition.
    - Second precondition.
- **Postconditions:**
    - First postcondition.
    - Second postcondition.
- **Main scenario:**
    1. Step 1
    2. Step 2
    3. Step 3
    4. ...
- **Alternative scenarios:**

  3b. Alternative at step 3
- **Remarks:**
    - First remark

UC1: Name Short summary of this use case scenario

2.4 Detailed use cases

2.4.1 *UC1*: Log in

- **Name:** log in
- **Primary actor:** the User
- **Secondary actor(s)**: secondary actor(s)
- **Interested parties:**
  - *System:* wants to authenticate its users.
- **Preconditions:**
  - First precondition.
  - Second precondition.
- **Postconditions:**
  - First postcondition.
  - Second postcondition.
- **Main scenario:**
  1. Step 1
  2. Step 2
  3. Step 3
  4. ...
- **Alternative scenarios:**

  3b. Alternative at step 3
- **Remarks:**
  - First remark

3. Non-functional requirements

In this section, we model the non-functional requirements for the system in the form of *quality attribute scenarios*. We provide for each type (availability, performance and modifiability) one requirement.

3.1 Availability

3.1.1 *Av1*: Name of the quality attribute scenario

Shortly describe the context of the scenario.

- **Source:** source
- **Stimulus:**
  - Description of a first stimulus.
  - Description of a second stimulus.
- **Artifact:** the stimulated artifact
- **Environment:** the condition under which the stimulus occurs
- **Response:**
  - Describe how the system should respond to the stimulus.
- **Response measure:**
  - Describe how the satisfaction of a response is measured.

UC2: Name Short summary of this use case scenario

3.2 Performance

3.2.1 *P1*: Name of the quality attribute scenario

Shortly describe the context of the scenario.

- **Source:** source
- **Stimulus:**
  - Description of a first stimulus.
  - Description of a second stimulus.
- **Artifact:** the stimulated artifact
- **Environment:** the condition under which the stimulus occurs
- **Response:**
  - Describe how the system should respond to the stimulus.
- **Response measure:**
  - Describe how the satisfaction of a response is measured.

3.3 Modifiability

3.3.1 *M1*: Name of the quality attribute scenario

Shortly describe the context of the scenario.

- **Source:** source
- **Stimulus:**
  - Description of a first stimulus.
  - Description of a second stimulus.
- **Artifact:** the stimulated artifact
- **Environment:** the condition under which the stimulus occurs
- **Response:**
  - Describe how the system should respond to the stimulus.
- **Response measure:**
  - Describe how the satisfaction of a response is measured.

3.4 Usability

3.4.1 *U1*: Name of the quality attribute scenario

Shortly describe the context of the scenario.

- **Source:** source
- **Stimulus:**
  - Description of a first stimulus.
  - Description of a second stimulus.
- **Artifact:** the stimulated artifact
- **Environment:** the condition under which the stimulus occurs
- **Response:**
  - Describe how the system should respond to the stimulus.
- **Response measure:**
  - Describe how the satisfaction of a response is measured.