Katholieke
Universiteit
Leuven

**Department of
Computer Science**

# Shared Internet Of Things Infrastructure Platform:

## Domain Analysis

Software Architecture (H09B5a and H07Z9a) – Part 1

Filipcikova-Halilovic

**Monika Filipcikova (r)**
**Armin Halilovic(r)**
Academic year 2016–2017

# Contents

# 1. Domain analysis

## 1.1 Domain models

This section shows the domain model(s).

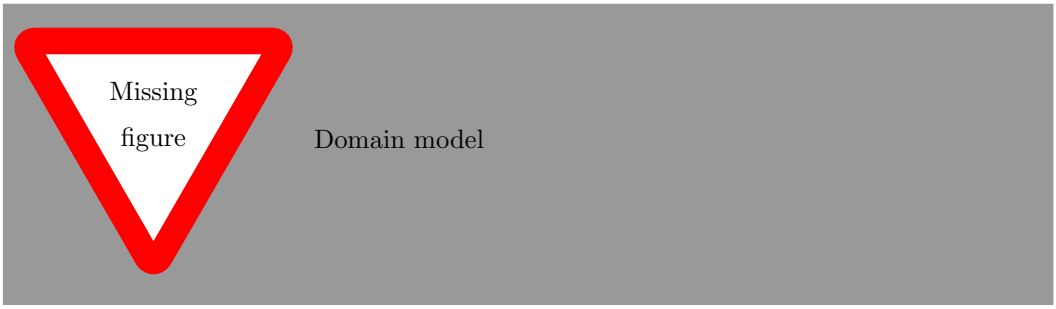


Figure 1.1: The domain model for the system.

## 1.2 Domain constraints

In this section we provide additional domain constraints.

- This is a first constraint.
- This is a second constraint.

## 1.3 Glossary

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

- **Notification**: definition
- **Replica**: definition

# 2. Functional requirements

## Use case model

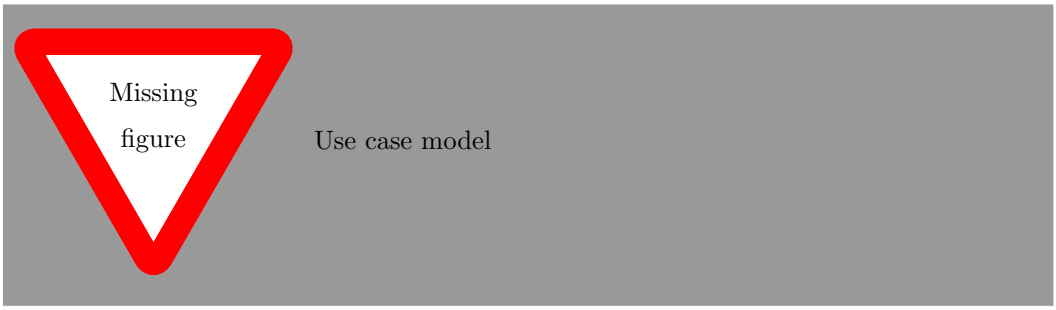


Figure 2.1: Use case diagram for the system.

## 2.1 Use case overview

**UCXXX: send data to gateway** A mote sends data to a gateway.

**UCXXX: upload application** An appplication developers uploads an application to the Online Service.

**UCXXX: get data from sensor** An application requests to get data from a specific sensor.

**UCXXX: search for applications** A customer organisation representative searches for available applications they could potentially subscribe to.

**UCXXX: review application** A SIoTIP system administrator reviews and approves or declines it.

## 2.2 Detailed use cases

### 2.2.1 *UCXXX*: Send sensor data

- **Name:** Send sensor data
- **Primary actor:** Mote
- **Secondary actors:** Sensor, Gateway
- **Interested parties:**
  - *Customer organisation:* pays for an application that uses this sensor.
  - *Infrastructure owner:* needs the sensor data for an application they set up.
  - *End-users:* use the sensor data in an application.
- **Preconditions:**
  - The mote has received data from a sensor connected to it.
  - The mote is connected to a gateway.
- **Postconditions:**
  - The gateway has received and processed the sensor data.
  - The Online Service has received the data and can process it.
- **Main scenario:**
  1. The mote sends the sensor data to the connected gateway.
  2. The gateway receives the data and if applicable, runs some application logic.
  3. The gateway collects data until a synchronisation point is reached. At that point, the gateway sends the data to the Online Service.
- **Alternative scenarios:**

  3b. The gateway determined that the data was important (e.g. cause for alarm, notification, etc.) and sent the data to the Online Service immediately instead of waiting for the sycnhronisation point.
- **Remarks:**
  - It is essential that the synchronisation protocol works correctly in the presence of non-reliable network communication so that there is no loss of data.

### 2.2.2 *UCXXX*: Upload application

- **Name:** Upload application
- **Primary actor:** Application Developer
- **Secondary actor(s)**: SIotIP system
- **Interested parties:**
  - *Customers organisations:* want to subscribe the applications.
- **Preconditions:**
  - The application developer has access to his dashboard.
- **Postconditions:**
  - The application is uploaded into the Online Service.
  - The application is available to customer organisations for subscription.
- **Main scenario:**
  1. The application developer logs in and opens his dashboard.
  2. The system provides the ability to upload new application.
  3. The application developer uploads application.
  4. The system check application and initiates a number of automated tests.
  5. The application developer follows the progress and results of these tests via application provider dashboard.
  6. The application successfully passes all tests.
  7. The system makes the application available for the customers organisations.
  8. The system send a notification to the application developer.
- **Alternative scenarios:**

  4b. The system can not load application and send error to the application developer.

  7b. The system interrupt loading of the application, because of potential memory leak.

  8b. The SIoTIP administrator performs a secondary review and decides whether to accept or reject the application.
- **Remarks:**
  - First remark

### 2.2.3 *UCXXX*: Get data from sensor

- **Name:** Get data from sensor
- **Primary actor:** Application
- **Secondary actor(s)**: Online Service, Gateway, Mote
- **Interested parties:**
  - *End-user:* wants to get information from sensors
- **Preconditions:**

- The application is uploaded in Online Service.
- A connection between Online Service and Getway is established.

- **Postconditions:**
  - The application received data from sensor.

- **Main scenario:**
  1. The application send request to Online Service.
  2. The Online service communicate with the Gateway.
  3. The Gateway relay information to the mote by sending request.
  4. The mote send data to Online Service (include:send sensor data (UCXX))
  5. The Online Service send data to the application.

- **Alternative scenarios:**

  5b. 5b. The application does not receive any data within 2s. It will retry the request 2 more times.

- **Remarks:**
  - First remark

### 2.2.4 *UCXXX*: Search for applications

- **Name:** Search for applications
- **Primary actor:** Customer orgranisation
- **Secondary actor(s)**: secondary actor(s)
- **Interested parties:**
  - *Name of interested party:* reason why party is interested
- **Preconditions:**
  - The primary actor is authenticated.
  - 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

### 2.2.5 *UCXXX*: Review application

- **Name:** Review application
- **Primary actor:** SIoTIP System Administrator
- **Secondary actor(s)**:
- **Interested parties:**
  - *Application developers:* want to add their application to the system
- **Preconditions:**
  - The system administrator is authenticated.
  - The application needs to be reviewed by a system administrator.
- **Postconditions:**
  - The system adminstrator accepts or declines the application.
  - The application developers have been notified of this event.
- **Main scenario:**
  1. The system administrator navigates to the applications component on their dashboard and selects the application that needs to be reviewed.
  2. The system administrator reviews the application's functionality and log history, and the message history with the developers.
  3. The system administrator indicates he wants to accept the application.
  4. The system logs this event and makes the application available to customer ogranisations.
  5. The system notifies the application developers.
- **Alternative scenarios:**

  3b. The system administrator indicates he wants decline to the application and is prompted by the system to fill in the reason for this. The application will thus not become available to customer organisations.

  3c. The system administrator indicates he wants communicate with the developers before making a decision. Afterwards, return to step 2.
- **Remarks:**
  - When an application first gets uploaded to the system, it needs to be reviewed and accepted by a system administrator. This will prevent the available apps to be flooded with apps of very poor quality or apps that are copies of other apps.
  - If at some point in the future, automated testing for an app fails, the app will need to be reviewed again before it can be made available again.

# 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*: Unusable database

A database in the system cannot be used anymore. This could happen because of corrupted disks, power outages, too many requests coming into the database server, network failures, natural physical disasters, etc. A different database will be used while this one is unusable.

- **Source:** Database server
- **Stimulus:**
  - The database crashed.
  - The database does not respond to requests.
  - The database does not respond to requests within reasonable time.
  - The database returns invalid data or responses.
- **Artifact:** Persistent storage
- **Environment:** Normal operation
- **Response:**
  - Use a working replica until the database can be used again.
  - Log the fault.
  - If the problem with the database cannot be fixed automatically (e.g. by an automatic restart and resynchronisation), send a technician if it is possible for them to fix the problem (e.g. replace corrupted disks). Otherwise (e.g. power outages, natural physical disasters), send a notification of high priority to SIoTIP system administrators.
- **Response measure:**
  - If a database becomes unusable, this should be detected within 3s of the system trying to use the database.
  - When the system detects that a database is unusable, a working replica should be used within 5s.
  - If a database is unusable because of a system crash or user error, the database should restart and start its boot procedure within 5s.
  - If a database is unusable because of corrupted disks, a technician should replace the disk within 30 min.

### 3.1.2 *Av2*: Broken sensor

A sensor breaks. The infrastructure owner is notified and if possible, another sensor is used for the responsibility that the broken one was fulfilling.

- **Source:** Sensor
- **Stimulus:**

  - No data can be received anymore from the sensor.
  - The sensor stopped sending regular updates.
  - The sensor sends invalid/corrupted data.
  - The sensor disappeared from the heartbeats of the mote it is connected to.
- **Artifact:** Communication channel between sensor and gateway
- **Environment:** Normal operation
- **Response:**
  - The gateway uses another sensor to be used for the same responsibility as the broken one.
  - Log the fault and notify the infrastructure owner.
- **Response measure:**
  - When a sensor breaks, this is detected within 2s.
  - If there is another sensor that can fulfill the same responsibility as the broken one, it should be chosen within 2s.

## 3.2 Performance

### 3.2.1 *P1*: Application requests under peak load

An application sends requests to the Online Service while the system is under peak load. These requests could be for getting data from a specific sensor, for configuring application settings, etc. These request should be processed in a timely manner.

- **Source:** Application
- **Stimulus:**
  - An application sends a request to the Online Service.
- **Artifact:** The whole system
- **Environment:** Under peak load
- **Response:**
  - When a request comes into the system, core work necessary to generate a reply is done first. Other work of less importance (e.g. sending emails, generating low-priority notifications, etc.) is scheduled to be done later.
  - Load balancing is used to divide the requests over available servers.
  - Servers that are closest to source of the request are chosen over servers that are farther away to minimize network delay.
- **Response measure:**
  - Applications get a response within 3s.

### 3.2.2 *P2*: Mote data to application delay

When a mote sends data to an application, that data reaches its destination in a bounded time. This bound is determined by the priority of the data. The possible priorities are low, medium, and high. These priorities can be set by an infrastructure owner.

- **Source:** Mote
- **Stimulus:**
  - A mote sends data to an application.
- **Artifact:** The whole system
- **Environment:** Normal operation
- **Response:**
  - The data is always sent to the next node before other extra work is done. For example, other work could be logging or sending notifications.
- **Response measure:**
  - If the data priority is low, the data reaches the application within 60s.
  - If the data priority is medium, the data reaches the application within 10s.
  - If the data priority is high, the data reaches the application within 1s.

## 3.3 Modifiability

### 3.3.1 *M1*: Add a new type of sensor

The SIoTIP corporation wishes to provide a new type of sensor to customers.

- **Source:** SIoTIP developer
- **Stimulus:**
  - Developers add a new type of sensor to the system
- **Artifact:** System codebase and database.
- **Environment:** Normal operation, during a development iteration
- **Response:**
  - The software on motes and gateways must be updated so that the new sensor can be read, calibrated and configured.
  - Components in the system responsible for storage of sensor readings must be updated to save the readings of the new sensor correctly.
  - UI components must be updated so that the readings of the new sensor can be displayed correctly.
  - This modification does not affect the way data is communicated whithin components of the system.
  - Infrastructure owners can choose when the update will happen. The motes and gateways must be updated before the new typ of sensor can be ordered.
  - The updates to the UI components and storage components can be deployed at run time. There is no need to destroy any login sessions for this, since the sensor will not be in use at that time yet.
- **Response measure:**
  - This modification is finished within 5 man months of development time.
  - The deployment of the modifications to the UI components and storage components is finished within 10 minutes.

### 3.3.2 *M2*: Changes to UI components or new UI components

Developers want to add or edit components that make up the UI. For example, they want to improve components that are reused in dashboards.

- **Source:** SIoTIP developers
- **Stimulus:**
  - Developers add a UI component.
  - Developers edit a UI component.
- **Artifact:** User interface and platform.
- **Environment:** Normal operation, during a development iteration
- **Response:**
  - Users are not able to use the system during the deployment of the changes. All users that are logged in at the time the deployment starts will be logged out first.
  - All users are notified in advance of incoming downtime.
  - The update does not affect currently ongoing activity in the system.
  - The update only affects the subsystem responsible for the UI.
- **Response measure:**
  - The deployment of the new UI components is finished within 30 minutes. Users can log in again immediately after the deployment is finished.

## 3.4 Usability

### 3.4.1 *U1*: System administrator reviews application

Before an application can be published or when its automated tests fail, it needs to be reviewed by a SIoTIP system administrator.

- **Source:** SIoTIP system administrator
- **Stimulus:**
  - The system administrator wants to review the application and/or test logs quickly.
  - The system administrator wants to contact the application developers.
- **Artifact:** System administrator dashboard
- **Environment:** At normal operation
- **Response:**
  - The system administrator dashboard is built up using clear components which contain informations about different elements of the system. One such component displays applications that are waiting for a review.
  - The application review page contains a component that describes the application's functionality. There are clear buttons to accept or decline the application. There is a button to open the application.
  - Optionally, the application review contains a component which displays images of the application from an infrastructure owner's point of view.
  - There is a component which displays the log history of the application.

  - There is a component for communication with the application developers.

- **Response measure:**
  - All actions for reviewing the application, accepting or declining the application, reading the application's log history, and contacting the developers are possible within 5 clicks.

### 3.4.2 *U2*: Infrastructure owner manages application

An infrastructure owner wants to manage an application. For example, they might want to edit the topology of some hardware, subscribe customer organisations to applications, order new hardware, etc.

- **Source:** Infrastructure owner
- **Stimulus:**
  - The infrastructure owner wants to use the infrastructure owner dashboard efficiently
  - The infrastructure owner wants to feel comfortable with the infrastructure owner dashboard
- **Artifact:** Infrastructure owner dashboard
- **Environment:** At normal operation
- **Response:**
  - The infrastructure owner dashboard consists of clear UI components that provide information about different elements of the system.
  - Topologies of hardware are displayed clearly in diagrams.
  - There is a help system the user can use to learn about the functionality of their dashboard. When this help system is used, it displays key information about the page the user is currently on. It also contains a search form so the user can learn about anything they want that is relevant to their dashboard.
- **Response measure:**
  - All actions an infrastructure owner can take in the system can be done within 5 clicks.