Final Report

Name: Harman Sandhu
Team: Group B
Student number: 215629967

Table of Contents

Project Summary

Lessons Learned from the Project

What have I learned from the group project:

Through this group project on smart home systems, I gain a tremendous understanding of the architecture and functionalities of modern-day smart home systems (SHS). As I got to explore the ecosystem that most SHS are built off, including components like thermostats, cameras, lights, smart locks, and security systems that are integral to the automation of the smart home application. The analysis of the integration of these various components helped me develop a solid understanding of the complexities and nuances of how these devices interact with the SHS.

Starting from the system proposal I was able to gather information about the problems that needed to be addressed and the objective that my system would aim to cover. The only thing I did not account for was the complexity of the scope I was trying to cover. Soon I realized my initial scope was too large for the requirements outlined leading me to narrowing the scope that I was to focus on. The scope of the system was adjusted in the Requirements Analysis Document (RAD) where I was able to identify key functional requirements (FR) and non-functional (NFR) requirements that were essential for the smart home systems usability. By creating these FRs, I was able to create various system models including Use Case Diagram, Scenario's to activity diagrams, sequence diagrams and state machine diagrams. The analysis from these diagrams allowed me to gain valuable feedback in terms of user interactions through the different scenarios that can occur and in the user flow within the system.

When developing my System Sequence Diagram's (SSD) I was able to use the analysis from the RAD document in order to define design goals from the NFR's. In addition, I was able to examine current software architectures in the market and select the architecture that would work best for the users of my system in terms of adaptability and flexibility. This would allow me to accommodate different user preferences and emerging technologies, to ensure service for long-term viability. The SSD allowed me to consider aspects such as hardware/software mapping, subsystem decomposition, global control flow, boundary conditions, access control and security, which are essential. This allowed me to visualize the interactions between the hardware components and software functionalities, allowing me to break down the system into manageable parts. From here I was able to understand the flow of control between components, identify the constraints of the system and make sure essential security measures were in place to prevent unauthorized access.

Things that went well and things that went wrong.

Overall, I believe we performed quite well in defining each category throughout the majority of the project. Making sure we got as much detail in to allow the readers of the report to understand the message we are conveying. Several challenges arose during the project, with one of the most significant being time management. Some work was submitted very close to the deadline, leaving little time for revisions. This made it hard to review and be able to change errors seen in the report because there would not be enough time to completely change the design or cases. Secondly, when approaching the decomposition of the Smart Home System, because of its complexity, it was very difficult to decompose the way we wanted it. This was due to limited resources available online regarding the topic. This took multiple sessions and conversation for clarification with the professor in order to better understand how to deal with the complexities of our system. There were obviously multiple designs that turned out to be incorrect for us with each iteration being really different from the other. But finally, we were able to agree on one that looked like it would work the best for us using the layered application design.

What are the difficulties in the project?

I feel like the general understanding of the concepts were very solid, but I do feel I didn't perform the best when it came to mapping persistent data. Although I was able to explain through words how I envisioned it to be I struggled with piecing the diagram together because I did not feel as confident as I did with all the other sections. I also felt I made a lot of small error's when it came to setting in protocol communication, for example when I listed as JDBC when I had it hosted using HTTP in the earlier section. I felt the difficulties in restricting the scope was hard in this project as well because once you block a part of the smart home system out, in later iterations you might need to open the scope up or revisit the initial scope to adjust accordingly. These adjustments can cause the group to complete sections with the previous scope ideals and not the restructured as they might view it from a different scope and can cause inconsistencies in the reports like use cases being missing.

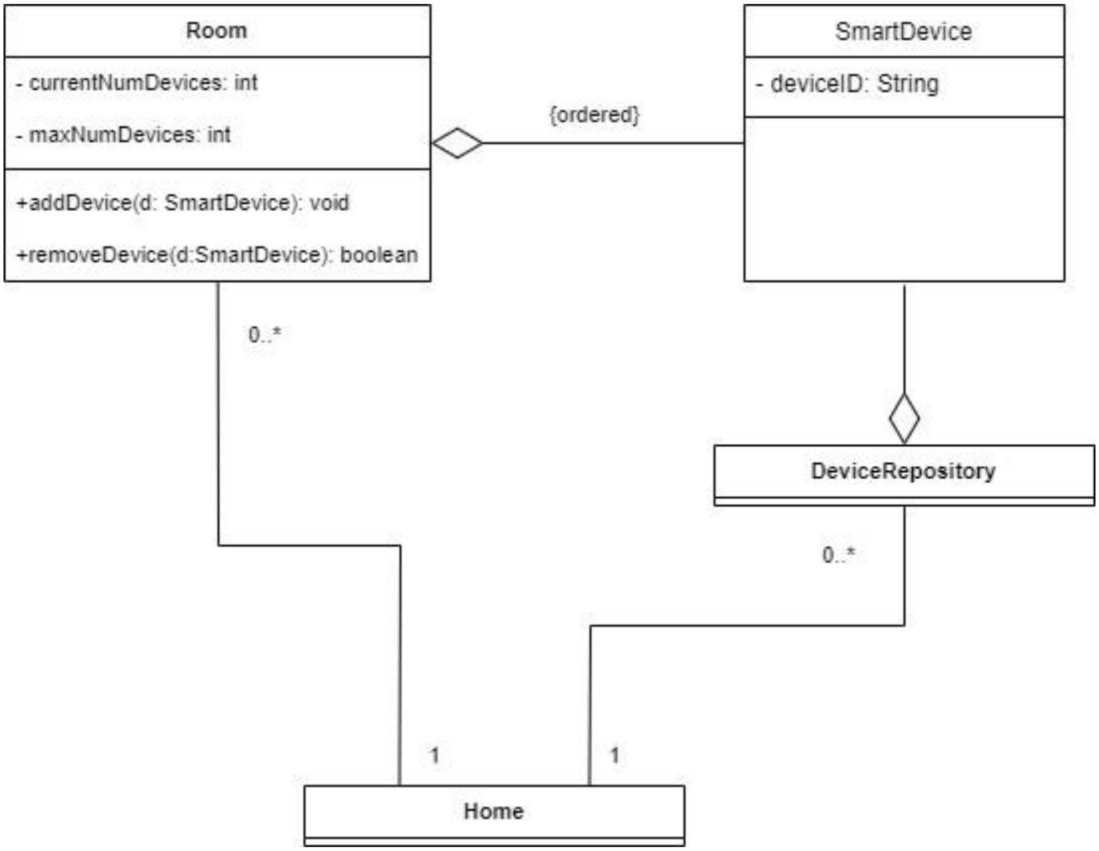
How can you improve the analysis and design procedure?

Based on the feedback received, there are several areas we can improve on in regards to the analysis and design procedure. The scope would be the first improvement that would require a more comprehensive and focused understanding, by making sure it aligns with the objects from start to finish. By limiting the scope, it would improve the design procedures by helping eliminate additional objects that could be added to the diagrams and ensuring no actors or use cases are missing. Another area that would have to be improved is the mapping of functional requirements to use cases, as there was a functional requirement that we did not map therefore weakening our overall analysis. In addition, taking the extra time to checking if all diagram components follow the UML diagraming rules. One key area would be the persistent data model where there are missing objects and relationships that are discussed in data management section but were not covered such as rooms and rules. In addition, elaborating on how notification component was used by the security management and device management but not by any other subsystem. By implementing these improvements, the overall analysis and design procedure will be enhanced to produce a professional document that is accurate, comprehensive, and professionally presented.

Object Design

Class interfaces

The component that was selected was Room, focusing on the adding and removing of devices.



Void addDevice(d:SmartDevice)
Pre-Conditions:

1. The current device count is less than or equal to the maximum devices after adding the device.
2. The device is from the deviceRepository of the Home.
3. The given device does not exist in the Room before.

```
context Room:: addDevice(d:SmartDevice) pre:
   maxNumDevices <= currentNumDevices + d.deviceID and
   home.deviceRepository.smartdevice - > includes (d) and
   smartdevice -> forAll(i:SmartDevice | i <> d)
```

Post-conditions:

1. The size of the devices increases by 1
2. The max number of devices remains unchanged.
3. The current number of devices is increased by 1
4. The device is added to the smartdevice collection.
5. The device is appended to the "device" list

```
context Room:: addDevice(d:SmartDevice) post:
   smartdevices -> size() = @pre.smartdevices->size() + 1 and
   maxNumDevices = @pre.maxNumDevices and
   currentNumDevices = @pre.currentNumberDevices + 1 and
   smartdevices->includes(d) and
   smartdevices->at(smartdevices->size()) = d
```

Boolean removeDevice(d:SmartDevice)

Post-Conditions:

1. If the 'smartdevies' contain the given smartdevice, the size of "smartdevices" should decrease by 1, and the currentNumDevices should decrease by 1.
2. If the "smartdevices" does not contain the given question, the size of "smartdevices" should remain the same, and the currentNumDevices should remain unchanged.
3. The result should be True if "smartdevices" contains the given question, and False otherwise.
4. The maxNumDevices should remain unchanged.

```
context Room:: removeDevice(d:SmartDevice) post:
  (@pre.smartdevices -> includes(d) implies
        (smartdevices->size() = @pre.smartdevices->size()-1) and
        (currentNumDevices = @pre.currentNumDevices – 1)
   ) and
   (not @pre.smartdevices->includes(d) implies
          (smartdevices->size() = @pre.smartdevices->size())
                   and (currentNumDevices = @pre.currentNumDevices)
   ) and
   (result = @pre.smartdevices->includes(q)) and
   (maxNumDevices = @pre.maxNumberDevices)
```

Invariants

1. All smartdevices are from deviceRepository.
2. There are no duplicated smartdevices in the Room

3. The maxNumDevices is greater or equal to the currentNumDevices

```
context Room inv:
   smartdevices->forAll(d:smartdevices | home.deviceRepository. smartdevices->exists(
   dr: DeviceRepository | dr.smartdevices->include(d)
   )) and
   smartdevices-> size() = smartdevices-> asSet()->size() and
   maxNumDevices >= currentNumDevices
```