Queen Mary University of London

School of Electronic Engineering and Computer Science

EBU5304 – Software Engineering Group Project

30% coursework. [*Student groups are allocated by the module organiser*.]

A smart energy management and monitoring system

-developing the software using Agile Methods

1. General information

In the next few weeks, your team will be required to develop the software of a smart energy management and monitoring system using Agile methods. Iterations should be planned and Agile methods should be used in all activities, from requirements, through to analysis/design, implementation and testing.

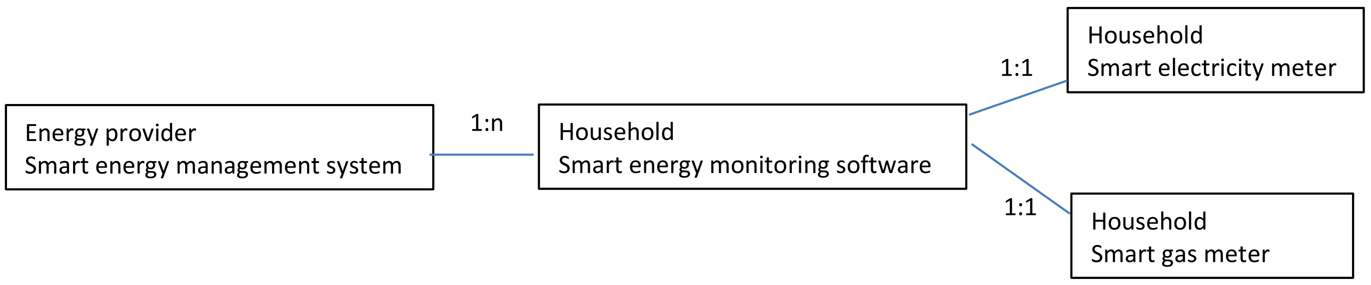
It should be noted that determining the requirements of a software is one of the most important and complex phases in any development project. The given specification contains a lot of noises—requirements are described in an abstract and ambiguous way. You should apply requirement finding techniques and Agile methods to extract the relevant information at appropriate level. Most importantly, you need to prioritise the features that are implemented in accordance with both ease of implementation and meeting customer requirements. As Agile is designed to adapt to change, do expect that new requirements or change of requirements will be available during the development stage. As in real software though, there may be more details you want to know that are missing from the given specification. You can make your own assumptions but **do NOT over design**. Keep your design **SIMPLE**. Bear in mind that there is no absolute right answer – your solution may be perfectly appropriate.

Handout release date: **Friday, 15th March 2018**
First QM+ submission (Product backlog): **Wednesday, 18th April 2018**
Final QM+ submission (Report and Software): **Friday, 1st June 2018**
Demonstration/feedback 1: during the split class session in teaching week 2 (23-27 April)
Demonstration/feedback 2: during the split class session in teaching week 3 (14-18 May)
Demonstration/feedback 3: during the split class session in teaching week 4 (4-8 June)

Marks returned: Approximately 2-3 weeks after the final demonstration.

2. Specification of project

You are a software development team that is responsible for developing a **smart energy management and monitoring system** as part of the big smart home project. The smart energy monitoring system should make it easy for the consumer to view their electricity and gas usage and help them to keep the cost down. The energy provider should be able to use the smart energy management system to set up tariff, get the meter readings from each household and generate bills.



As shown in the diagram above, each household can have only one smart energy monitoring software, one smart electricity meter and one smart gas meter. The smart energy monitoring software can communicate with the smart electricity meter and smart gas meter. The energy provider uses the smart energy management system to communicate with the smart energy monitoring software at each household.

The high-level specifications and requirements are described as follows:

2.1 Smart energy monitoring software

View electricity consumption and cost

The system should display the live electricity usage of the day, in kWh. The usage should be updated frequently, e.g. approximately every 30 seconds. It should also show the current electricity cost of the day, in pounds and pence £.

View gas consumption and cost

The system should display the live gas usage of the day, in kWh. The usage does not need to be updated frequently, e.g. approximately every 30 minutes would be good enough. It should also show the current gas cost of the day, in pounds and pence £.

Budget and alert

Consumer should be able to set a budget to keep track of the spending. The system should be able to show the consumption level, compare with the budget and give alert. For example, if the electricity or gas usage is lower than the budget, display a green signal; or if it is over the budget, display a red signal. The consumer should be able to change the budget at any time. The budget can be set in either usage (kWh) or cost (£).

View historic consumption and costs

Consumer should be able to view the historic consumption and costs, for example daily, weekly and monthly electricity/gas usage and cost.

Check tariff

The tariff is set by the energy provider and can only be changed by the energy provider. The consumer can only view the current tariff (price per kWh).

Send meter readings to energy provider

The system should send gas and electricity meter readings to the energy provider every month, on a fixed day of the month (e.g. every 1st of the month). The meter reading is a 4-digit number, (0000-9999) with each unit is 1 kWh.

2.2 Smart energy management system

Add/view/remove consumer and meter information of each household

The energy provider should be able to add new consumer to the system and remove existing consumer. When adding the consumer to the system, the household’s smart energy monitoring software and smart meter are registered to the system. The system should be able to remove an existing consumer. The energy provider can view the readings and bills history of the consumer.

Set tariff

The tariff is set by the energy provider and can only be changed by the energy provider. The standard tariff is as below:

- Gas Unit rate 3.88p per kWh
- Electricity Unit rate 14.60p per kWh

Receive meter readings and generate bills

The system should receive monthly gas and electricity meter readings from households and generate a monthly bill and send it to the consumer by email and/or post.

2.3 Other requirements

- Basic restrictions and error checking must be considered: for example, the meter reading should be a positive integer; meter reading can only increase. etc.
- The software should be easy to use: that is, the user should be able to operate the software with common sense or with simple instructions.
- The software should be user friendly: for example, the user should be able to cancel the operation at any time; it should display messages promptly to user during the operation; etc.
- The software must be developed using Java as a stand-alone application. Simple graphic user interface (GUI) should be used. Java SE 8 or above should be used.
- All input and output files should be in simple text file format. You may use plain Text (txt), CSV or XML. Do NOT use database.
- Your design must be flexible and extensible, so that it can adapt to continuously changing requirements and can be used in a general market in the future. E.g. connect to other smart home devices such as smart light, smart radiator, smart curtain and smart speakers etc.
- Your design of the software must be capable of adapting to such future changes. That is, when developing a new software, you should be able to reuse the existing components. When adding new features to the existing software, you should make the least impact on the existing code.

Your tasks are to define detailed requirements, design, develop and test the above described software using Agile methods. For any details of the software or operation which is not clearly stated, ***you may make your own assumptions***. In your report, make it clear where you have made assumptions. Feel free to design the software as long as it satisfies the basic requirements, but **do NOT over design it**.

3. Agile project management

Each coursework group has 6 students[1]. You are the Agile team working together to complete the coursework. All students in a group must work on all aspects of the project, in order to obtain full software engineering skills.

You should use the techniques you have learnt in the lectures to manage the project, e.g. Scrum, daily stand up meetings, working around a table, scrum master and one hand decision making etc.

[1] Due to the size of the cohort, some groups may occasionally have 7 students instead.

4. First QM+ submission: 18th April

The first QM+ submission is **Product Backlog**.

- Use the template provided on QM+, feel free to modify it to meet your needs.
- The submission must be an EXCEL file.
- Only the group leader should submit the excel file.
- The file must be named **ProductBacklog_groupXXX. xlsx**, where **XXX** is your group number.

5. Final QM+ submission: 1st June

The final submission includes a short report and software.
The short report should contain the following parts:

i. Project management

- The project management in your team working. E.g. using project management techniques, planning, estimating, decision making and adapting to changes.

ii. Requirements

- Apply the requirements finding techniques.
- Describe any changes of the product backlog since the first submission.
- Iteration and estimation of the stories.

iii. Analysis and Design

- A design class diagram describing the design of the software classes in the software, show the class relationships. Note that your design should *address the issue of re-usability of software components*. You should provide clear justification for your proposed approach and show that your design is adaptable to change where necessary.
- Discuss the design of the software.
- Discuss the extent to which your design and the code that implements it meets the main design principles of programming.

iv. Implementation and Testing

- Discuss the implementation strategy and iteration/built plan.
- Discuss the test strategy and test techniques you have used in your testing.
- Discuss the using of TDD. Note: TDD is not required for developing the whole software, however, you should try to use TDD to develop a few programs.

v. All reports should include a list of references in the appendix.

vi. Main screenshots of the system should be included in the appendix (**Note**: Convert them to JPEGs).

Final report must be in PDF format (maximum 15 pages, excluding the Appendix). This must be named **FinalReport_groupXXX.pdf**, where **XXX** is your group number. Only the group leader should submit the file.

The software should contain the following parts:

i. A working software application written in Java. All main functionality should be implemented. Code should be well documented.

ii. A set of test programs using Junit as an example of using TDD.

iii. JavaDocs.

iv. User manual.

v. Note: You only need to consider the logical information flow for implementing some functions. For example, you do NOT have to consider the actual communication between energy provider, monitor software and devices, you may assume they are all running in the same computer; you do NOT have to actually implement the email function, you may display a message show "email sent".

You must submit a ZIP format file containing all the .java files of product programs and test programs, Javadocs, user manual and a Readme file to instruct how to set up or configure and run your software. Do not include .class files, as your programs will be re-compiled. This must be named S**oftware_groupXXX.zip**, where **XXX** is your group number. Only the group leader should submit the file.

6. Demonstration

Demonstration 1: during the split class session in teaching week 2 (23-27 April)

- Product backlog
- Paper prototype
- Up-to-date iteration of WORKING software

Demonstration 2: during the split class session in teaching week 3 (14-18 May)

- Up-to-date iteration of WORKING software
- Unit Testing

Demonstration 3: during the split class session in teaching week 4 (4-8 June)

- Final version of WORKING software
- Integration Testing

ALL group members MUST attend all the demonstration sessions. For each demonstration, it is OK if only some features are implemented, your code is incomplete or has bugs – just show your latest built of the working system, you still could receive full marks of the demonstration. Remember we look at setting a priority for the features that are implemented in accordance with both ease of implementation and meeting customer requirements. Detailed instructions of each demonstration will be sent out in due course.

7. Important notes

Only coursework materials submitted via QM+ will be accepted. It is the responsibility of each group to check that their submissions are correct.

Although real software would require more advanced features (such as consideration of security issues, database access, data synchronisation etc) this would distract from the core software engineering skills that must be developed on this module. The following guidelines MUST be followed.

- **Standard-alone application:** You must develop a stand-alone application and ignore any networking concerns. Students must NOT write HTML, JavaScript, servlets, JSPs, sockets, ASPs, PHPs, etc … This is NOT a network programming module.
- **NO database implementation**. Students should develop this application without using a database, i.e. do no not use JDBC, Access/Postgres/MySQL database etc. Students should concentrate on their software engineering skills without being concerned by more precise deployment issues. **Hint**: Students should think about the use of Java interfaces for the database (or proxy database) access.
- **Code development.** Code should be written for maintainability and extensibility – it should both contain Javadocs and be clearly commented. Responsibilities should be separated – one class should be responsible for one thing only.
- **Code delivery.** A Readme file should explain clearly how to install, compile (i.e. what to type at the command line) and run the code. All code should run from the command line and MUST NOT require users to install any extra software (e.g. database, Eclipse or any other IDE) or extra Java libraries.
- **Key Points of report.** Markers will be impressed by quality, not quantity – please pay attention to the page limit. Markers will be impressed by groups who can criticise their solution and indicate how this can be improved in future iterations. Students should take care over the presentation of the report (and check for spelling and grammar mistakes) – they should imagine that this report will be presented to the client. Students should not spend hours and hours concentrating on making the most beautiful GUI! For example, no marks will be gained for designing a lovely logo. The focus of this work is software engineering – correct functionality and elegance of code (classes that do only one thing, methods that do only one thing, code that is not duplicated, delegation, i.e. following the principles outlined in the course) are much more important.
- **Key points of Participation and Achievement.** If students are not turning up to meetings or doing any work, then the module organiser need to be informed **immediately**. The coursework project is marked out of 100. Individual participation/achievement will be evaluated through the demonstration sessions. If a student has not participated at all in the group, they will get 0% of group marks.

8. Marks breakdown (approximate)

<u>Group mark</u> (maximum 100 marks)

Demonstration 1: 30%

- Ability to extract and define the software requirements using Agile techniques.
    - Use of appropriate fact finding techniques.
    - Correctness of writing user stories.
    - Correctness and completeness of product backlog.
    - Quality of prototype.
- Progress to date and project management

Demonstration 2: 20%

- Ability to refine the requirements through analysis and Ability to design high quality software
    - Quality of design
- Unit testing
- Progress to date and project management

Demonstration 3: 20%

- Quality of the final version software
- Integration testing
- Project management

Final submission: 30%

- Correctness of Java code – the code must match the design. *If the code does not match the design* (or if there is no correspondence between the design/code), then ALL these marks will be lost.
- Quality of Java code
- Testing: - appropriate test strategy
- Quality of report

<u>Individual mark</u>
Individual marks will be given according to the participation in the group: Quality of work performed and Understanding of the performed work. At each demonstration, each student will be evaluated through explaining the work and answering the questions. Grade will be awarded as below:

A	Satisfactory	Receive 100% group marks
B	Unsatisfactory	Receive 50% of group marks
C	No contribution or absent	Receive 0% of group marks

You, <u>AS A GROUP</u>, are responsible for managing any issues and for completing all of the tasks.

Please use the messageboard on QMPlus for enquires and discussions; do not email the lecturers unless it is a personal related issue.