Title of Dissertation

A dissertation submitted in partial fulfilment of

the requirements for the degree of

BACHELOR OF ENGINEERING in Computer Science

in

The Queen's University of Belfast

by

Saoirse McCann

9TH May 2017

**SCHOOL OF ELECTRONICS, ELECTRICAL ENGINEERING and COMPUTER SCIENCE**

**CSC3002 – COMPUTER SCIENCE PROJECT**

**Dissertation Cover Sheet**

A signed and completed cover sheet must accompany the submission of the Software Engineering dissertation submitted for assessment.

Work submitted without a cover sheet will **NOT** be marked.

Student Name: Student Number:

Project Title:

Supervisor:

**Declaration of Academic Integrity**

Before signing the declaration below please check that the submission:

1. Has a full bibliography attached laid out according to the guidelines specified in the Student Project Handbook
2. Contains full acknowledgement of all secondary sources used (paper-based and electronic)
3. Does not exceed the specified page limit
4. Is clearly presented and proof-read
5. Is submitted on, or before, the specified or agreed due date. Late submissions will only be accepted in exceptional circumstances or where a deferment has been granted in advance.

**I declare that I have read both the University and the School of Electronics, Electrical Engineering and Computer Science guidelines on plagiarism - http://www.qub.ac.uk/schools/eeecs/Education/StudentStudyInformation/Plagiarism/ - and that the attached submission is my own original work. No part of it has been submitted for any other assignment and I have acknowledged in my notes and bibliography all written and electronic sources used.**

*Student’s signature* *Date of submission*

**Acknowledgements**

To those who have helped the author during the project and the preparation of the dissertation and to anybody who has given financial support.

**Abstract**

A summary (100 words) which provides an outline of the subject matter and the results, findings and/or conclusions of the dissertation.

This thesis examines the methods that could be implemented by a company to maximise the utilisation of resources such as desks and meeting rooms.

This master thesis examines what type of measurement should be conducted in agile software development. The paradigm shift that agile methodology offers is in conflict with traditional measurement approaches and there is a need to clarify if and how measurement can benefit the agile practice. In this paper we develop a model that addresses both performance and process optimization measurement in agile processes. The model is evaluated at bwin Games AB, a web game software company based in Sweden. As a part of the evaluation, an agile inspired measurement framework for implementation of costefficient and flexible metrics is successfully tested. The study shows that measurement can be beneficial for the agile practice, but that careful consideration to dysfunctional behavior is necessary.

**Contents**

A complete list of chapters, sections, appendices etc. with page numbers.

**1.0 Introduction and Problem Specification**

Background material should be given which introduces the problem area, its context and background. You should identify the particular problem under consideration along with information about the problem area that enables the reader to understand the problem scope and nature. If your project involves a particular domain, algorithm, method theory etc., you may describe it in the introduction (alternatively or additionally, it may be described later, if appropriate). For best marks the student should show that they have systematically researched and fully analysed the problem, synthesising the relevant information.

Nowadays, companies face major overheads of office space and resources such as desks, computers, rent, etc. With people working from home, off sick, on annual leave or on customer sites, a number of resources can be left vacant. These resources cannot be utilised by others as they are still technical “occupied” by said employee.

It’s becoming more normal for companies to move away from the normal resource allocation process of providing each employee with a desk. The more efficient process of allocating resources as required allows companies to ensure that no resource is not utilised when it could be, and ensures each employee only has the resources they require.

The goal is to develop an end to end software solution which will aid in the planning and management of resources within a modern open plan office.

The system must have an intuitive UX design allowing the user to add all of the required information regarding their company. Once all of the information has been entered the system will provide an Executive Officer Dashboard that will display any relevant Key Performance Indicators (KPIs) that the company requires to effectively run their business. The dashboard will provide charts and graphs which will aid the user in making decisions on how best to manage and allocate their resources to make them more cost efficient.

When a user is selecting a resource the system will provide functionality for them to auto assign their desk space, selecting the desk that is located closest to their line manager.

The system shall also have an end point to allow users to log on and book a resource from their mobile device. This will involve touch screen development.

When booking a resource, consideration will have to be taken into the schedule of each resource to ensure it is not double booked or booked beyond its capacity.

The system was designed with two end users in mind:

1. Administrators – all end users who are added to the system as administrators will have access to the Executive Officer Dashboard and the resource booking functionality.
2. Non-Adminstrators – all end users who t=do not have administrative rights will only have access to the resource booking functionality, they should not be able to access the Executive Officer Dashboard.

1.1 Pareto’s 80-20 rule

roughly 80 percent of the effects of anything you might be doing come from 20 percent of the causes

80% of available space is underutilised.

You could have 2 big meeting rooms that hold 100 people each, while all the desk space for the office is cramped together. By looking a the frequency at which each of the rooms is used, how many occupants they have when they are ulisaised a company would be able to make decisions based on these caclations which will allow them to make changes to better suite their company needs. For example, if the first room has an frequency rate of 90%, and an occupancy rate of 50%. And the second room has a frequency rate of 100%, and an occupany rate of 50%. This means the company could get rid of one of the rooms and spilt the toher room into two. Leaving space for more desks for the company to expand.

Example 2. Both of the rooms have a high frequency and occupancy rate. If the occupants in the room are company employess this means that their current desks are left vacant. Pareto’s rule is relevant here as it gives an insight into the fact the you do not need to provide resources for all employees. The company could still expand, keep the same location with the same rent. This is when a booking systems for resources would come in useful. And were a dashboard containing information on the frequency, occupany and the utilisation of each resource, on a certain day at a particular time may become very cost eefective.

**2.0 System Requirements Specification**

You should provide a precise description of the system developed. These may have been written much earlier but for the dissertation they should be updated to match the final system delivered. You should list any assumptions made about the problem and any system constraints. Overall your requirements, functional AND non-functional should be complete, clear, accurate, feasible and objectively verifiable. Content depends on your project but could include:

* A complete set of function definitions (as use cases if preferred), as far as possible written so as to be testable
* Measurable and testable non-functional requirements
* Description of interfaces required such as with other software or systems
* Any specific user interface requirement
* User characteristics

The target to aim for here is that your requirements could be the basis for a contract or handing to external developers to complete.

2.1 Assumptions

Due to the time constraints that have been set for developing the solution the following assumptions have been made:

* A prototype solution will be developed for one office consisting of a single one level building
* When an employee is off work on annual leave or working from home they do not need to book any resources for said days
* It is the responsibility of each user to book their resources for the upcoming week
* Archiving of the “Timetable” table in the database will not be implemented for the prototype developed. Although possible approaches on how this would be carried out will be investigated.
* Users only have to book resources for Monday – Friday between 8am and 6pm.
* Users will only be able to book resources for the current week and the upcoming week.
* Resources can only be booked against existing users of the system

2.2 Functional Requirements

1. Administrators shall have the functionality to add information regarding the office’s measurements
2. Administrators shall have functionality to add new users to the system
3. Administrators shall have functionality to add new resources on the system
4. Information regarding existing resources should be available to the administrator to update
5. Information regarding existing users should be available to the administrator to update
6. Accounts shall be automatically created for each employee when they are added to the system
7. An email shall be sent to users with their log in credentials when their account has been created
8. Administrators shall have access to an executive officer dashboard which will provide information of measurements of utilisation
9. Analytics for the key performance indicators on the executive officer dashboard shall be displayed in the form of charts and graphs:
10. Utilisation rate: frequency rate \* occupancy rate
11. Frequency rate: percentage of time space is used compared to its availability
12. Occupancy rate: how full the space is compared to its capacity
13. A user shall be able to send a change request for an occupied resource
14. A user shall have access to functionality to delete a resource booking for the current week, the week prior and the upcoming week
15. The user shall have functionality to book resources for themselves for the current week and the upcoming week
16. The system shall have functionality to allow a user to auto assign their desk for a day which will select the closest desk available to their line manager.
17. The system shall prevent users from booking multiple resources for the same time period
18. The system shall keep a timetable of allocation for each resource
19. The system shall keep a timetable of resource allocation for each user showing their bookings for the current week, the week prior and the upcoming week
20. The system shall prevent a resource being double booked
21. The system shall prevent a resource from being over booked

Non-Functional requirements

1. The system should be mobile compatible to allow a user to easily:
   1. book a meeting room for themselves and other attendees
   2. delete a resource booking from their schedule
   3. book resources
2. Modified data on the system should be updated in the database within 2 seconds for all other users accessing it.
3. The system should allow the user to easily switch between booking a desk, room or other resource.
4. The UI should be intuitive and user friendly, requiring minimal trianing for the end user.

**3. Design**

This section should describe the design of your proposed system. Normally this several parts, depending on your project:

1. Architectural Description of the system – textual and/or diagrammatic. This could be a simple diagram showing the components and how they relate or it could describe the choice of architectural style or pattern used.
2. User Interface Design (if applicable). Show sketches of the design or screenshots with explanations of choices made, if necessary.
3. Software System Design.

The role of each component and the interfaces between components should be described. There should be a clear correlation between your design and your specification.

The design should be linked to requirements and, where applicable give a critical discussion of key design decisions/styles/patterns used. There might be a data model, a UI design, details of external interfaces, and of other important issues e.g. concurrency, event handling, error and exception handling, security, data persistence. No particular notation or tool is mandated.

**4. Implementation and Testing**

You should describe any languages, packages, and libraries etc. that are used in the development of your system. There is no need to describe your code in detail. You may highlight data types and implementation techniques that are of special interest. If appropriate, you may provide:

1. Choice of implementation language(s)/ development environment(s)
2. Use of software libraries;
3. Key implementation decisions
4. A description of how some important functions and algorithms were implemented.
5. A description of how each component is implemented.
6. Discussion of Test Approach e.g. unit testing, system testing, regression testing etc; Test cases described; Testing tools used. Evidence that testing coverage was complete.

Program code can be accessed by the assessors via the git repository **so there is no need to print code listings**. It is recommended that you comment code appropriately. Programs should be written in a clear style with good program structure and well-defined data structures. The program code should reflect its design.

**5. System Evaluation and Experimental Results**

Different projects will have a different emphasis. In all cases you are expected to provide empirical results and to draw conclusions from those results. You may use your software to generate experimental results. Be sure to describe the methodology of your evaluation or experimentation. An experiment is typically described in terms of its goals, the hypotheses being tested, the subject of the experiment, what is being measured and what is controlled, the results obtained and the analysis and interpretation of those results. Alternatively, you can assess the product in terms of how it compares with other similar products and/or in terms of user feedback (e.g. via a survey) or some measurable quality aspect such performance efficiency or reliability. Your supervisor can guide you on what is appropriate, but typically the very best projects have shown results that could be publishable with little or no work or show an exemplary empirically based evaluation of a software product

**6. Conclusion**

A general summary evaluation of the success of the project should be given with respect to criteria identified in the introduction. A discussion of the significance of your experimental results may be appropriate. Do they agree with other previous work or ideas? How does your system compare with similar ones? An evaluation of the hardware/software environment and language used may be presented, if appropriate. Draw conclusions on the process used in the project as well. What went well? What did not go well? What are the strengths of your solution or conclusions? What are the weaknesses? Suggestions for further work should also be discussed. You can be critical and draw a negative conclusion. Not all projects will be successful. A well-explained failure is as an acceptable an outcome as a spectacular success. Assessors are looking for excellence in a critical appraisal of the work and a convincing argument for the significance of contribution in the context of wider work.

## Background Reading

Supervisors may provide references to suitable background material for the project. However, it is the student’s responsibility to read around the area using books, articles and web based material. There are various search tools available via the library site (Use the Article search to obtain research papers). Google Scholar is also very useful.

## Use of other resources

* You must not make any use of any projects which are available online unless approved by your supervisor (any such projects should be referenced clearly). Project submissions will be checked for plagiarism (via the Turnitin).
* DO NOT copy text from other sources unless placed in quotes and cited.
* Even if you write something in your own words but it is based on an existing source, please place a citation in the document to the source.

**References**

A list of references to documents (books, papers, web pages etc.) which are referred to in the main body of the text. Use the IEEE citation style as detailed here <https://www.ieee.org/documents/ieeecitationref.pdf>

References

<http://businessanalystlearnings.com/blog/2013/7/26/15-tips-for-writing-better-requirements>

<http://www.processimpact.com/articles/qualreqs.html>

<http://searchcio.techtarget.com/definition/executive-dashboard>

<http://www.freshbusinessthinking.com/utilisation-analysis/>

<http://www.smg.ac.uk/documents/utilisation.pdf>

<http://reqtest.com/requirements-blog/functional-vs-non-functional-requirements/>

<http://www.requirements.com/Glossary/NonFunctionalRequirements/tabid/91/Default.aspx>

<http://softwareengineering.stackexchange.com/questions/162399/how-essential-is-it-to-make-a-service-layer>

<https://msdn.microsoft.com/en-us/library/ff648105.aspx>

<https://www.lucidchart.com/pages/er-diagrams>

<http://www.conceptdraw.com/examples/er-notation>

<http://survey.datassentialsurveys.com/help/Inquisite_ER_Diagram.htm>

<http://kc.litiumstudio.se/documentation/previous-versions/litium-studio-4-6/litium-studio-4-6-2/litium_studio_ecommerce/architecture-design/er-diagrams>

<http://www.tableau.com/sites/default/files/media/which_chart_v6_final_0.pdf>

<http://www.smg.ac.uk/documents/utilisation.pdf>

<http://www.investopedia.com/terms/c/capacityutilizationrate.asp>

<http://www.businessballs.com/pareto-principle-80-20-rule.htm>

<https://en.wikipedia.org/wiki/Pareto_principle>

<http://ui-patterns.com/blog/The-8020-rule--the-Pareto-principle>

**The first citation should be the URL to the software code repository which should contain the code and any other resource required to run the software.**

**Appendices**

These should include as appropriate:

(a) A User manual giving details on how to use the software, including details of input data, output formats and error messages.

(b) Test results, if appropriate.

(c) Other information which is not convenient or appropriate to include in the main body of the dissertation.

(d) Minutes of the Project meetings.

Delete functionality -> it was implement so that it would not affect the calculations Resources, Users and Slots

Add interview with workers -> reference in conclusion that perphaps and electronic tracker would be more useful