**UNIVERSITY OF HERTFORDSHIRE**

**Faculty of Engineering and Information Sciences**

**Modular MSc Honours in Computer Science (Software Engineering)**

**7WCM0031 Software Engineering MSc Project (Online)**

**Final Project Report**

**May 2016**

***Determine if the development and implementation of software engineering technologies, will increase business productivity and performance of ‘MSc Properties’***

**Mr D L Edwards**

**SRN: 07156987**

**Supervised by: Thiago Matos**

Abstract

With the increasing number of people having to move out of their local area, due to the current unemployment rates and recent legislation changes, resulting in local councils being able to house homeless families outside of the local borough [2] in private rented accommodation, [3] and benefit caps [2] meaning households are unable to afford their local area rental rates.

This means that estate agents such as the fictional estate agent ‘MSc Properties’, require a secure and fast means to store and share data and documents with all of their different stores that could be 100’s of miles apart, and therefore require a system that will enable each store to access data and documents for ‘MSc Properties’ from a number of different locations.

For me to alleviate the issue of data and document storing, and shared data and document access for ‘MSc Properties’, I decided to incorporate a number of design, implementation and testing techniques, such as UML diagrams, Remote Method Invocation, Object Relational Mapping, Unit Testing, System Testing and more, which is described in more detail during this report, to produce a distributed system that will achieve the requirements of ‘MSc Properties’.

From the testing results outlined in the report, the project successfully managed to implement a distributed system that manages ‘MSc Properties’ data and documents, allowing for any user at any store location, who is logged in to the system to retrieve any data or documents, depending on whether or not they have the right security levels.

Also ‘MSc Properties’ data and documents is successfully stored to a MySQL database, to enable the system to be robust and protect data and documents against system crashing or downtime.

Furthermore, I have successfully implemented an observer pattern which enables the home screen of each user to be updated, whenever certain data within the system is created, updated or deleted, providing an up-to-date home screen feed at all times.

Moreover, I have successfully implemented task scheduling functionality that enables tasks to be automated, reliving ‘MSc Properties’ employees of tasks they would have had to manually complete.

Lastly, I have successfully implemented a website that widens the advertising of ‘MSc Properties’ services and allows for potential customers to submit a service request notifying the required office of customer interest, allowing for the request to be processed.

Finally, using the software engineering technologies outlined during the literature review, I was able to develop and implement a distributed system that achieved the main core objectives and all of the advanced objectives outlined in this report, and in doing so I was able to assess the benefits gained by the business implementing the system, over their current business processes, which produced more streamlined business processes and improved the business efficiency and effectiveness.

I believe the project was a success as I successfully achieved the main objectives, resulting in me being able to draw conclusions on the benefits gained through the implementation of software engineering technologies, and although there were a few changes I would have made during the project such as implementing a document management and task scheduling framework from an existing tested framework or writing a more accurate project plan at the beginning of the project, the project was overall a success.

Acknowledgements

I would like to thank:

Firstly, my project supervisor Thiago Matos Pinto who has been there to guide and assist me from the start of the project, through to the final submission date of the project.

Secondly, all the lecturers that have taught me over the course of both my BSc and MSc Software Engineering courses, each providing me with knowledge and skills that has enabled me to produce this dissertation.

Finally, my family and close friends who have been with me through my ups and downs at University, providing me with love, support and encouragement.

**Table of Contents**

**Abstract1**

**Acknowledgements3**

**Table of Contents4**

1. **Introduction7**
   1. Introduction to the project7
      1. Report Structure7
   2. Project Aim8
   3. Project Objectives8
      1. Core Objectives8
      2. Advance Objectives9
   4. Project Background9
   5. Literature Review10
      1. Design Methodologies10
         1. Software Development Approach10
         2. Modelling Software Behaviour12
         3. Design Patterns13
      2. Development Methodologies15
         1. Networking15
         2. Document Management17
         3. Task Scheduling18
         4. Database Management Systems19
         5. Web Server20
         6. Graphical User Interface21
      3. Testing Methodologies22
         1. Unit Testing22
         2. System Testing23
   6. Project Plan24
   7. Relevance to target award25
   8. Required Resources & Skills26
      1. Hardware26
      2. Software26
      3. Access26
      4. Skills26
   9. Ethics Approval27
2. **Design28**
   1. Introduction28
   2. Software Lifecycle28
   3. Modelling System Behaviour29
      1. Use Case Diagrams29
      2. Class Diagram30
      3. Enhanced Entity Relationship Diagram31
      4. Sequence Diagrams32
      5. Storyboard34
3. **Implementation35**
   1. Introduction35
   2. Iteration Cycle 135
      1. Design Patterns35
         1. Iterator Pattern35
         2. Inheritance36
         3. Object Composition36
         4. Singleton Pattern36
      2. Graphical User Interface37
   3. Iteration Cycle 238
   4. Iteration Cycle 340
      1. Push vs Pull40
      2. Local Reference vs Remote Reference41
   5. Iteration Cycle 441
   6. Iteration Cycle 542
   7. Iteration Cycle 643
   8. Iteration Cycle 743
   9. Iteration Cycle 844
   10. Iteration Cycle 945
4. Testing46
   1. Unit Testing46
   2. Integration Testing48
   3. System Testing49
5. **Conclusion54**
   1. Objectives Achieved54
   2. Choice of Methodology56
   3. Project Plan and Management57
6. **Evaluation58**
   1. Quality of Work58
   2. Changes to Project59

**Bibliography61**

1. Introduction
   1. Introduction to the project

For my MSc Computer Science Dissertation (Software Engineering), I decided to solve the problem of data and document management through the implementation of a distributed system for a fictional estate agent called ‘MSc Properties’.

The reason for me undertaking this project is that there is an ever increasing demand for distributed systems which manage businesses data and documents, enabling the business to become more efficient, and effective and produce more streamlined services.

Additionally, by adopting specific functionality when developing a distributed system to manage a business’s data and documents, it can reduce the need to relay changes to the business data and documents to all within the business as these are located in a shared area, accessible to all who have the right access privileges.

Finally, it can improve business security as documents are not stored in a physical location, nor are employees able to perform jobs without having the right access privileges, as security is placed on the data and documents, and each system user will be provided a user account which will manage their security privileges preventing them from performing an action which they do not have the correct security privileges to perform.

* + 1. **Report Structure**

During this report I will document the aim and objectives of this project and explain the problem background in more detail, looking at the different factors that contribute to the problem identified.

I will then document the research and literature review I carried out in order for me to identify the different software engineering techniques and technologies available to solve the problem.

I will then document the design of the system, outlining the different software engineering techniques, and why these were chosen. I will then explain how these were used to enable me to structure and manage the project, and just as importantly allow me to produce diagrams that virtualized the structure, behaviour and interaction of the distributed system.

Furthermore, I will document the implementation and testing of the distributed system, explaining which techniques and technologies I chose, and why they were the best solution for this project, given the available resources. I will then go on to provide an analysis of the test results to draw some conclusions on the validity of the software produced.

Lastly I will document my conclusion and evaluation of the project as a whole, stating whether or not I have successfully achieved the project aim and objectives. I will then outline what went well, what didn’t go so well, and what I would do different if I was to do the project again.

* 1. Project Aim

The aim of this project is to tackle the issues of data and document sharing across the Internet by implementing software engineering technologies for a fictional estate agent called ‘MSc Properties’, and then assess whether the implementation of these technologies would result in increased productivity and performance for the business.

The distributed system should allow ‘MSc Properties’ to share business data and documents across the Internet, whilst providing data security and integrity. ‘MSc Properties’ requires the distributed system to be maintainable, dependable and usable, which means I will explore the different techniques that support program specification, design, validation and evolution of software to implement this distributed system.

* 1. Project Objectives
     1. Core Objectives

Analyse ‘MSc Properties’ current business processes by week 5.

Complete literature searches and review of existing data management systems, identifying the software engineering models, methodologies, tools and metrics used in the development process by week 12.

Set out functional and non-functional requirements for the development within the requirements document by week 9.

Ensure required resources are available for the entire project by week 9.

Carry out risk assessment by week 10.

Set out the distribution mechanism I am going to employ for the distributed system by week 13.

Develop a suitable data management system model that meets the requirements defined by week 15.

Write test scripts to test the implementation of the system outlined in the development model by week 15.

Develop a suitable database to handle the business data and import dummy data into the database by week 16.

Develop a suitable application to handle the business processes and connect to the database to store the business data by week 24.

Develop a suitable search facility so users can search for information stored in the database, and should be implemented by week 24.

Develop reporting functionality so certain users can report on business performance indicators by week 24.

Develop a log in facility for users, allowing for restricted access, and to prevent unauthorised access and should be implemented by week 26.

Test the system using the test scripts created, ensuring the test results are above the acceptable failure rate defined in the requirements by week 33.

Develop and test a user manual by week 31.

Evaluate the project in a report to detail the entire development and outline what went well and what could have been done better by week 34.

* + 1. Advanced Objectives

Develop a website to advertise services offered to potential customers/suppliers. Customers/Suppliers will be able to register and submit a service request through the website and should be implemented by week 26.

Develop document management facility that allows for documents to be stored electronically, and should be implemented by week 26.

Develop a home screen which provides a live feed of the tenancies and leases due to expire by week 26.

Develop a reset password facility, so users are able to reset their password if they have forgotten it allowing users to establish access to the system. This should be implemented by week 26.

* 1. Project Background

‘MSc Properties’ is a fictional estate agent with a number of sites nationwide across England. Due to the current unemployment rates, and recent legislation changes resulting in local councils being able to house homeless families outside of the local borough [3] and benefit caps [2] meaning families have to move out of their local borough due to not being able to afford local rents [3]. ‘MSc Properties’ require the need to be able to transfer customers between sites, meaning the transfer of data and documents across sites that could be 100’s of miles apart.

‘MSc Properties’ require me to develop a distributed system to create and manage their property portfolio, as well as creating and managing customer/supplier/employee accounts. They require this so that data can be stored on a server or locally and all the officers of different ‘MSC Properties’ sites will be able to access this data. The system will have a login facility to provide restricted access for users, and will also allow managers of ‘MSc Properties’ stores to manage their employee accounts. The system will also allow ‘MSc Properties’ managers to report on business data.

‘MSc Properties’ have allocated an office manager to the project to assist with analysis of current business processes, map system requirements, and communicate back to ‘MSc Properties’ with the work that is occurring in project meetings, and relaying any project document back to the business for sign off. From the project meetings a business analysis and system requirements document was produced which is documented in Appendices A.

I have presented this project background as this outlines the fictional business ‘MSc Properties’ current processes and the analysis I conducted of their current processes to produce the system functionality that has driven the literature reviews, design stages, implementation and testing for this project.

Additionally, the background information has provided me with a basis of a business and their processes for me to compare my project deliverables against, as I will be aiming to determine if the distributed system developed during this project has resulted in more streamlined business processes thus, increasing productivity and performance for the business.

* 1. Literature Review

As previously outlined, I will now document the literature reviews I undertook to allow me to understand the different options available to me to enable me to successfully achieve the project aim and objectives.

* + 1. **Design Methodologies**

The problem I am trying to solve by undertaking this project, requires a piece of software to be developed for me to successfully achieve the project objectives surrounding data and document sharing.

But as well as trying to solve the business problem surrounding data and document sharing, there is also the problem of producing software that is maintainable, dependable and usable, so it will require me to produce high quality software. This means I am required to undertake literature reviews into different design decisions I will need to make when designing the system, and also require the project to go through a structured development process to give the project the highest possibility of successfully achieving the project aim and objectives.

* + - 1. **Software Development Approach**

The first design principle I am going to discuss is the software development approach and the piece of literature I am going to review is the article called *“Software Quality & Agile Methods”* written by *M. Huo, J. Verner, L Zhu and M.A. Babar* [5]*.* This article looks at the quality of software produced when comparing the Waterfall Model and Agile methods, and specifically how agile methods can achieve high quality software even if the process is not linear and a complete requirements specification has not been developed prior to the design and implementation stage of the development.

The article then goes on to conclude that “agile methods do have practices that have Quality Assurance abilities, some of them are inside the development phase and some others can be separated out as supporting practices. The frequency with which these agile Quality Assurance practices occur is higher than in a waterfall development and lastly, agile Quality Assurance practices are available in very early process stages due to the agile process characteristics” [5]. The below diagrams show the different methods and Quality Assurance techniques undertaken within the Waterfall Model and Agile Methods.

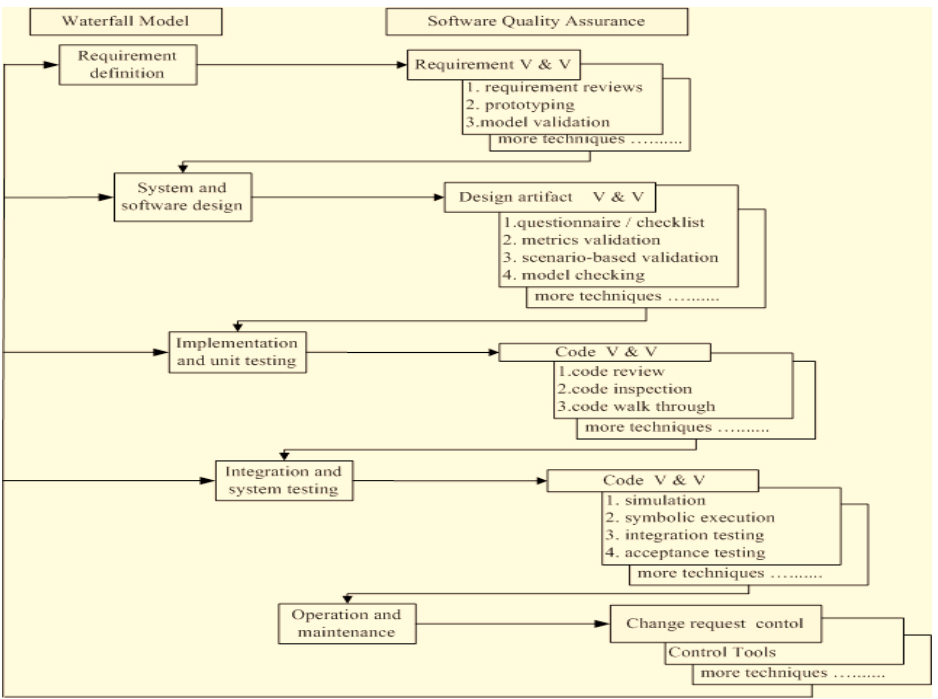


Fig. 1. Waterfall Process Model with Quality Assurance Techniques [5].

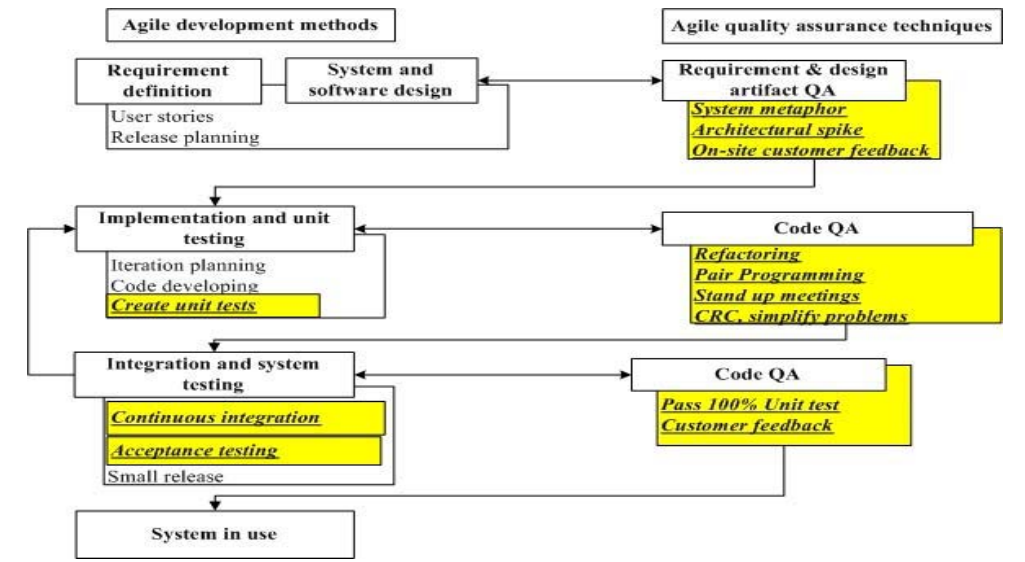


Fig. 2. Agile Development Methods with Quality Assurance Techniques [5].

By adopting a software development approach for this project, it will enable the project to have structure and whether or not I decide to go with a more waterfall based approach, or a more agile based approach, by adopting the software approach method it gives the project a higher rate of success as the project is following some sort of pre-defined process, set out by the software approach model.

Furthermore, from the literature review I have come to understand that even though the software process may not be linear (follow a strict structure, of one section leading on to another, and so on), the more agile process is still able to develop software that meets high quality standards, but just does it in a less formal way than the waterfall model.

* + - 1. **Modelling System Behaviour**

The piece of literature I am going to review for the modelling of system behaviour is the article called *“Designing Concurrent, Distributed, and Real-Time Applications with UML”* written by *H. Gomaa* [6]*.* This article looks at two areas, the software design method called Concurrent Object Modelling and Architectural Design Method (COMET), which is an example of a Unified Modelling Language (UML) based method, and the different modelling required for concurrent, distributed and real time applications using UML.

The article explains that “In the requirements model, the system is considered as a black box and the use case is developed… In the analysis model, the emphasis is on understanding the problem, hence the emphasis is on identifying the problem domain objects and the information passed between them… In the design model, the solution domain is considered, so the analysis model is mapped to a concurrent model” [6]. These different models are what I will have to consider when going through the software development process for this project.

Below is a list of different techniques to model system behaviour which I have come across during the research for this project:

* Use Case Diagrams – A representation of a user’s interaction with the system, showing the relationship between the user and the use cases they are involved in [16].
* Data Flow Diagrams – A graphical representation of the “flow” of data through an information system [17].
* Class Diagram - A static structure diagram that describes the structure of a system showing the system classes, their attributes, methods, and the relationships amongst objects [18].
* Entity Relationship Diagram – A data model for describing the data or information aspects of business domain or its process requirements, in an abstract way that lends itself to ultimately being implemented in a database such as relational database [19].
* Class Responsibility Collaborator (CRC) Models – A brainstorming tool used in the design of object-oriented software, documenting the dynamics of object interaction and collaboration [20].
* Sequence Diagrams – An interaction diagram that shows how processes operate with one another and in what order, showing object interaction arranged in time sequence [21].
* Storyboards – A graphic organizer in the form of illustrations or images displayed in sequence for the purpose of pre-visualising a motion picture, animation, motion graphic or interactive media sequence [22].

By looking at the system behaviour through the requirements, analysis, and design models, and considering what design techniques I will implement, it will enable me to forward plan, how each component to my system is going to be implemented and how they will interact with each other, and allow me to then document this behaviour through design documents.

And by documenting the system behaviour in design documents it will improve the chance of developing high standard software that meets the requirements model.

* + - 1. **Design Patterns**

The last design principle I am going to discuss is design patterns, and most importantly the design patterns I can employ within the software I am going to produce. Again as with modelling system behaviour previously, this project centres on software development, and part of the aim is to produce software that is maintainable, dependable and usable, so it will require me to produce high quality software, and by implementing design patterns, it will allow me to produce software with high cohesion, low coupling, encapsulation, and other metrics of software development which indicate high quality software.

Ian Sommerville explains design patterns as a description of accumulated wisdom and experience, a well-tried solution to a common problem, and the Hillside Group puts it as “Patterns and Pattern Languages are ways to describe best practices, good designs, and capture experience in a way that it is possible for others to reuse this experience” [8].

Below are some design patterns I have come across during my research for this project:

* Observer pattern – A software design pattern in which an object, called the subject, maintains a list of dependents, called observers, and notifies them automatically of any state changes, usually by calling one of their methods [23].
* Singleton pattern – A software design pattern that restricts the instantiation of a class to one object [24].
* Strategy pattern – A software design pattern that enables an algorithm’s behaviour to be selected at runtime [25].
* Creational pattern – A software design pattern that deals with object creation mechanisms, trying to create objects in a manner suitable to the situation [26].
* Iterator pattern – A software design pattern in which an iterator is used to traverse a container and access the container’s elements [27].
* Composite pattern – A partitioning software design pattern, which describes that a group of objects is to be treated in the same way as a single instance of an object, allowing clients to treat individual objects and compositions uniformly [28].
* Inheritance – Is when an object or class is based on another object or class, using the implementation to maintain the same behaviour, and is a mechanism for code reuse and in programming languages that support inheritance, produce an “is a” relationship between sub classes and its parent class. [29].
* Object Composition – Is a way to combine simple objects or data types into more complex ones, and are a critical building block of many data structures. Composition can be regarded as a relationship between types: an object of a composite type “has an” object of a simpler type [30].
* Object Relational Mapping – Is a programming technique for converting data between incompatible type systems in object-oriented programming languages, and in effect creates a “virtual object database” that can be used within the programming language [31].

The last design principle I have looked into is another very important design principle as this offers to me a wide range of techniques that are tried and tested through experienced programmers, and also outline common pitfalls developers are likely to fall into when implementing certain technologies, enabling me to forward plan how I am to implement the functionality, ensuring that I do not fall into the common mistakes made by previous programmers.

Although none of the design principles outlined directly assist me to achieve the project aim of determining if the development and implementation of software engineering technologies, will increase business productivity and performance of ‘MSc Properties’, this will assist me to develop high quality software giving me the best results to perform my evaluation on whether or not business productivity and performance has increased.

* + 1. **Development Methodologies**

I am now going to discuss the different development methodologies I uncovered during the course of the project, whilst undertaking the literature review.

* + - 1. **Networking**

One of the major problems I am trying to solve by undertaking this project, is to be able to produce a system that allows ‘MSc Properties’ to share data and documents across the Internet, whilst ensuring that the validity of the data and documents being shared are maintained. For me to do this, it has meant that I have had to undertake a literature review into the different networking technologies that will enable me to successfully achieve the project aim and objectives related to networking.

The first piece of literature I am going to review for Networking is the article called *“Implementing Remote Procedure Calls”* written by A.D. Birrell and B.J Nelson [8]. This article talks about the options that face the designer implementing remote procedural call (RPC) functionality and the considerations that need to be made when making decisions on this type of system.

The article states “when making a remote call, five pieces of program are involved: the *user*, the *user-stub*, the RPC communications package (the RPCRuntime), the *server-stub*, and the *server*… When the user wishes to make a remote call, it actually makes a perfectly normal local call, which invokes a corresponding procedure in the user-stub. The user-stub is responsible for placing a specification of the target procedure and the arguments into one or more packets and asking the RPCRuntime to transmit these reliably to the callee machine. On receipt of these packets, the RPCRuntime in the callee machine passes them to the server-stub. The server-stub unpacks them and again makes a perfectly normal local call, which invokes the appropriate procedure in the server. Meanwhile, the calling process in the caller machine issues pended awaiting a result packet. When calling the server completes, it returns to the server stub and the results are passed back to the suspended process in the caller machine. There they are unpacked and the user-stub returns them to the user” [8]. This process is represented in the below figure.

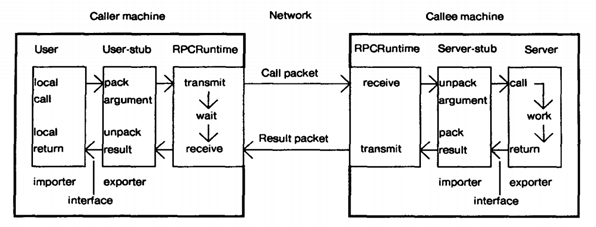


Fig. 3. The components of the system, and their interactions for a simple remote call [8].

The Remote Procedural Call functionality highlights the type of technology I could implement to successfully achieve the project aim and objectives of data sharing amongst ‘MSc Properties’ hosts that are at different locations.

The second piece of literature I am going to review for Networking is the article called “*Push vs. Pull in Web-based Network Management”* written by *J.P. Martin-Flatin* [9]. This article talks about two models of network management, which are “The Pull Model” and “The Push Model”, which represent two well-known approaches to exchanging data between two hosts with a distance between them, which is one of the major problems I am trying to solve during this project.

The article states “The pull model is based on the request/response paradigm, the client sends a request to the server, then the server answers, either synchronously or asynchronously. This is functionally equivalent to the client “pulling” the data off the server. In this approach, the data transfer is always initiated by the client, i.e. the manager. The push model, conversely, is based on the publish/subscribe/distribute paradigm. In this model, agents first advertise what Management Information Bases they support, and what Simple Network Management Protocol notifications they can generate; the administrator then subscribes the manager (the Network Management Station) to the data he/she is interested in, specifies how often the manager should receive this data, and disconnects. Later on, each agent individually takes the initiative to “push” data to the manager, either on a regular basis via a scheduler (e.g., for network monitoring) or asynchronously” [9]. The article then goes on to state that “the pull model, well suited to ad hoc management, and the push model, well adapted to regular management” [9].

By implementing networking functionality, it will enable users of the system to have access to data and documents stored on a server, where the server is not necessarily local to them, meaning that employees from MSc Properties, that could be miles apart from other offices, and the server, can have access to MSc Properties data and documents providing they have the right security privileges to access that information.

By doing this, it provides MSc Properties with better data and document sharing capabilities, instead of sharing data through emails, or phone calls, or faxing a document across to another employee/office which is currently done. Data and documents are stored within the system, and depending on the networking set up, this data can be stored locally or remotely, but any changes to system data will be propagated to a server, which can then either be stored there, or propagated out to all users, ensuring that all users have the most up to date information at all times, or at least upon user request.

Therefore, implementing networking functionality can assist me to achieve my project aim of determining if the development and implementation of software engineering technologies, will increase business productivity and performance of ‘MSc Properties’.

* + - 1. **Document Management**

The next development area I am going to discuss is document management, as the business deals with numerous documents that need to be stored, and made available for access by any host at different locations.

The piece of literature I am going to review for Document Management is the article called *“Electronic Document Management: Challenges and Opportunities for Information Systems”* written by *R.H. Sprague, Jr* [10]*.* This article talks about a number of benefits gained from implementing a document management system, such as improving the publication process, supporting organisational processes and communication amongst people and groups, improving access to external information, creating and maintaining documents, maintaining corporate records and lastly promoting training education.

The article states “Documents are stored electronically, shipped over a network and printed when and where they are needed, resulting in reduction in obsolescence, elimination of warehouse costs, and reduction or elimination of delivery time” [10]. The article later goes on to explain “The benefits of Electronic Document Management for these applications are, quicker access to the documents, more efficiency in the search process, simultaneous access by several people to the most current version of the document, and reduced cost of printing and distributing documents” [10].

Below are some document management frameworks I have come across during my research for this project:

* Apache JackRabbit – Is a content repository which implements the Content Repository for Java (JCR), with support for structured and unstructured content full text search, versioning, transactions, observation and more [32].
* Modeshape – Is a distributed, hierarchical, transactional, and consistent data store with support for queries, full-text search, events, versioning, references, and flexible and dynamic schemas, which implements the Content Repository for Java (JCR) [33].

By implementing document management functionality into the system I am to develop for ‘MSc Properties’, it will offer a number of business benefits as defined in the literature review such as quicker access to documents, version control for the document, and providing a central location for all documents for the business, in doing so, providing a more streamlined service with more efficient and effective business processes, than that of a paper based documents and physical filing system currently implemented at ‘MSc Properties’.

Therefore, implementing document management functionality can assist me to achieve my project aim of determining if the development and implementation of software engineering technologies, will increase business productivity and performance of ‘MSc Properties’.

* + - 1. **Task Scheduling**

The next development area I am going to discuss is task scheduling, and in particular task scheduling in real time systems, this is because the distributed system that will be developed for ‘MSc Properties’ during this project will be a real time system, dealing with the processing of scheduled tasks.

The piece of literature I am going to review is the article called *“Application of Real-Time Monitoring to Scheduling Tasks with Random Execution Times”* written by D. Haban and K.G. Shin [11]. This article talks about the calculation of execution time for posteriori tasks (calculation of execution time requires experience with the given task) scheduled within real time systems, and that the worst-case execution time is usually used to ensure that enough time has been allocated for the task to be completed, and discusses the drawbacks with this sort of approach and alternatives to this method.

The article states “real-time tasks are usually scheduled based on their worst-case execution time, and since the worst-case execution time can be several orders of magnitude larger than the true execution time, scheduling tasks based on the worst-case execution time can lead to severe underutilization of CPU cycles and/or incorrect decision on the schedulability of tasks i.e., some tasks are declared to be un-schedulable even if they can be completed in time” [11].

Below are some task scheduling frameworks I have come across during my research for this project:

* Quartz project –
* Haban & Shin project

By implementing task scheduling functionality into the system I am to develop for ‘MSc Properties’, it will offer a number of business benefits such as the automation of tasks, meaning employees of MSc Properties will be freed up to carry out other tasks resulting in more efficient and effective business processes, than that of MSc Properties employees carrying out the tasks manually which is currently implemented at ‘MSc Properties’.

Therefore, implementing task scheduling functionality can assist me to achieve my project aim of determining if the development and implementation of software engineering technologies, will increase business productivity and performance of ‘MSc Properties’.

* + - 1. **Database Management Systems**

The next development area I am going to discuss is database management systems, and in particular data models based on object-orientated concepts. This is because I have decided to develop a distributed system using object-orientated concepts, which means I will need to adopt a database management model based on these object-orientated concepts.

The piece of literature I am going to review is the article called *“An Introduction to Object-Oriented Database and Database Systems”* written by *M.L. Horowitz* [12]*.* This article looks at the possibilities of combining most of the desirable features of database systems with desirable features of the object orientated model of computation and the below diagram outlines these features.

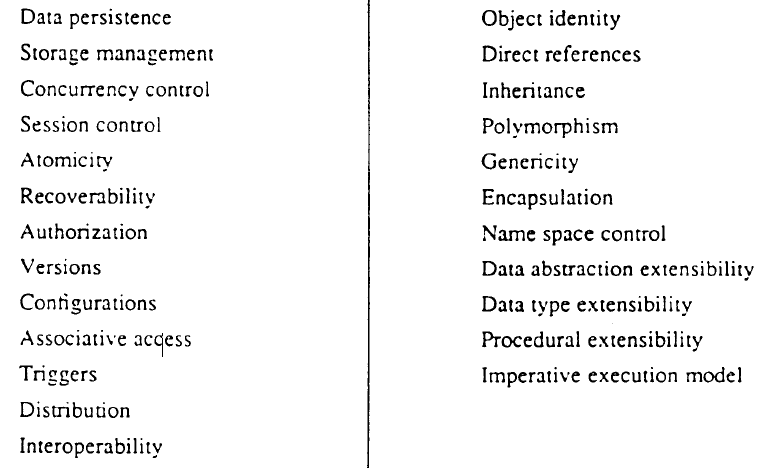


Fig. 4. Features provided by database Fig. 5. Features provided by object-oriented systems [12]. Languages [12].

The article later goes on to explain the issues that can arise when combining the database and object-oriented model concepts, by stating that “First integration should occur without impedance mismatch. In particular, language support for object-oriented database services should be orthogonal and transparent. Second, integration should not lose any advantages of existing data models. For instance, object-oriented programming does not support data independence inherently, so features such as relationship support and query joins should be provided. Finally, integration presents an opportunity for introducing new desirable features” [12].

Below are some database management systems I have come across during my research for this project:

* Microsoft Access – Is a Database Management System (DBMS) from Microsoft that combines the relational Microsoft Jet Database Engine with graphical user interface and software development tools [34].
* MySQL – Is an open-source relational database management system (RDBMS) and the most widely used open-source client-server model RDBMS [35].

By running a document management system, and connecting my application to a document management system, it will offer a number of business benefits such as the back-up of system data in case of system crash or system down time, and additionally, enable other applications to connect to the database management system to extract data from this for other purposes such as a website, resulting in more efficient and effective business processes, than that of MSc Properties storing paper-based documents in physical filling systems, or advertising system services through office windows currently implemented at ‘MSc Properties’.

Therefore, running a document management system can assist me to achieve my project aim of determining if the development and implementation of software engineering technologies, will increase business productivity and performance of ‘MSc Properties’.

* + - 1. **Web Server**

The next development area I am going to discuss is web servers, and in particular the different web server software that can be implemented to receive and manage the Hypertext Transfer Protocol requests sent by a user of the website that will be developed to advertise ‘MSc Properties’ services to potential customers, and also manage communication between the web server and the database management system discussed previously.

The piece of literature I am going to review is the article called *“Specification and Implementation of Dynamic Web Site Benchmarks”* written by *C. Amza, A. Chanda, A.L. Cox, S. Elnikety, R. Gil, K. Rajamani and W. Zwaenepoel* [13]*.* This article looks at the movement from web content being static HTML or image files, to web content becoming dynamic through the combination of a front end web server (web browser such as Internet Explorer), an application server (software such as Apache, along with server side scripting in PHP and SQL), and a back-end database (software such as Microsoft Access) and in particular identifies benchmarks for dynamic web sites by comparing 3 different dynamic web sites, looking at the bottleneck characterizations for these web sites.

This combination of technology would allow me to develop a dynamic website that can advertise ‘MSc Properties’ services to potential customers, and when staff perform actions in the system it will result in updates to the website, for example, a property being rented to a customer meaning the property is no longer available to rent and therefore should no longer be advertised on the website as available to rent, which will increase their potential customer pool, as they would be now advertising to anyone with internet access, and additionally, reduce the employee task of updating the advertising window at each office. However, one drawback is that a skilled staff needs to be employed or contracted to assist with the management of the website over time (excluding property adverts).

Therefore, developing a website that can update dynamically can assist me to achieve my project aim of determining if the development and implementation of software engineering technologies, will increase business productivity and performance of ‘MSc Properties’.

Below is some web server software I have come across during my research for this project:

* Apache HTTP Server – Is the worlds most used web server software, and played a key role in the growth of the World Wide Web quickly becoming the most dominant HTTP server [36].
* Nginx – Is a web server with a strong focus on high concurrency, performance, and low memory usage [37].
* Cherokee – Is an open source cross-platform web server that runs on Linux, BSD, variants, Solaris, Mac OS X, and Microsoft Windows [37].
  + - 1. **Graphical User Interface**

The next development area I am going to discuss is graphical user interface (GUI), and in particular the different frameworks that can be adopted with my chosen programming language to develop an interface for the ‘MSc Properties’ staff to interact with the system I am going to produce during this project.

Implementing a GUI will enable users of the system to interact with the system I am developing, and allow the process of using the system to be easier than a command line interface and improve the look and feel of the system.

Additionally, due to the windows, menus, and click, drag and drop functionality provided by a graphical user interface, it will improve the look and feel experience of the user, instead of the command based functionality provided by a command line interface, where the user needs to know the command and its parameters to interact with a command line interface.

By implementing a GUI for the system developed, would offer a number of business benefits such as an improved look and feel system, and an easier to use system, resulting in more efficient business processes, than that of implementing a system which run a command line interface.

Therefore, implementing a system that offers a graphical user interface to interact with, can assist me to achieve my project aim of determining if the development and implementation of software engineering technologies, will increase business productivity and performance of ‘MSc Properties’.

Below are some Java GUI frameworks I have come across during my research for this project:

* Abstract Window Toolkit (AWT) – Is Java’s original platform-dependent windowing, graphics and user-interface widget toolkit, and is part of the Java Foundation Classes (JFC) [38].
* Swing – Is a GUI widget toolkit for Java, was developed to provide a more sophisticated set of GUI components than the earlier AWT, and is also part of the JFC [39].
  + 1. **Testing Methodologies**

As previously explained, this project will centre around developing a system to tackle the major problem of data and document management and sharing, and whilst meeting these functionality requirements of ‘MSc Properties’, the system also needs to be maintainable, dependable and usable. This means that I will need to carry out testing to ensure that the system I have developed for ‘MSc Properties’ is compliant with the specified requirements and that the system has no faults or errors at runtime of the system, or the system is at an acceptable level for ‘MSc Properties’.

* + - 1. **Unit Testing**

The first testing principle I am going to discuss is the unit testing approach to testing, and will be fundamental to ensuring each of the elements of the system work independently of each other (where elements are not coupled).

The piece of literature I am going to review for unit testing is the article called *“A Simple and Practical Approach to Unit Testing: The JML and JUnit Way”* written by *Y. Cheon and G.T. Leavens* [14]. This article looks at ways programmers can reduce the writing of labour-intensive code for unit testing, by writing formal specifications (for example, pre and post-conditions of methods).

The article goes on to explain that “writing formal specifications instead of test code makes the programmer’s task easier, because specifications are more concise and abstract than the equivalent test code, hence more readable and maintainable. Furthermore, by using specifications in testing, specification errors are quickly discovered, so the specifications are more likely to provide useful documentation and inputs to other tools” [14].

The first testing concept I have looked into is unit testing, and specifically carrying out JUnit tests, which provide me with an automated way of testing elements within my system on their own, to enable me to identify if that element is able to perform its required function, and that it is being performed in the expected way. And as explained in the literature review, by writing formal specifications for the unit testing and identifying pre and post conditions for methods, it enables bugs within an element of the system to be identified easier, due to the passing and failing of these conditions showing where the problem is.

Below is some unit testing framework I have come across during my research for this project:

* JUnit Testing – Is a unit testing framework for the Java programming language, and is linked as a JAR at compile time [42].
* Java Modelling Language (JML) – Is a specification language for Java programs using Hoare style pre and post conditions and invariants, that follows the design by contract paradigm. Specifications are written in Java annotation comments to the source files [43].
  + - 1. **System Testing**

The next testing principle I am going to discuss is the system testing approach to testing, and will be fundamental to ensuring each of the elements of the system work together as they should.

The piece of literature I am going to review for system testing is the article called *“A UML-Based approach to System Testing”* written by *L. Briand and Y. Labiche* [15]*.* This article looks at system test cases being derived from the analysis stage documents such as use case diagrams and sequence diagrams.

The article goes on to explain that “Deriving test requirements from early artefacts produced at the end of the analysis development stage, namely use case diagram, use case description, interaction diagram associated with each use case (sequence or collaboration), and class diagram (composed of application domain classes and their contracts). This early use of analysis artefacts is very important as it helps devising a system test plan, size the system test task, and plan appropriate resources early in the life cycle. Once the low level design is complete, when detailed information is available regarding both application domain and solution domain classes, then test requirements can be used to derive test cases, test oracles and test drivers” [15].

The final testing concept I have looked into is system testing, and in more detail, producing system test plans from the design documents produced during the analysis and design stages of the development, such as use case diagrams and sequence diagrams, as this will enable me to ensure that the software developed is meeting the requirements outlined in the early stages of the project.

This is very important because large projects can sometimes develop software that works, but is not actually meeting the system requirements, and because unit testing is predominantly carried out adopting white box testing, meaning the testing is driven from the source code, it is difficult to pick up these errors with regards to not meeting system requirements, because the code could compile and run and produce the correct results, but not meeting the system requirement, whereas the system testing is carried out adopting the black box testing and is driven by the design documents and system requirements developed in the early stages, which will pick up inconsistences with source code and system requirements and design documents.

Below are some testing technologies I have come across during my research for this project that will assist in bug tracking:

* Bugzilla
* The Bug Genie

Although none of the testing principles outlined directly assist me to achieve the project aim of determining if the development and implementation of software engineering technologies, will increase business productivity and performance of ‘MSc Properties’, they will assist me to identify problems with the system, enabling me to ensure that the system is meeting the system requirements and working as expected, and in doing so provide me with software that will give me the best results to perform my evaluation with, and determine if MSc Properties business productivity and performance would improve.

* 1. Project Plan

As this is a large project, it is very important that I planned, monitored and managed the project smoothly from start to finish. I have used a Gantt chart, which provides a graphical illustration of the schedule of the project, broken down by project objectives, with completion dates for each objective, which has helped me track the activities in the project and make changes to work being carried out if necessary. This tool has been used to manage my time and allow me to stay on schedule as best as possible, as there was a lot of tasks that needed to be completed in a limited time frame.

This process of project planning is outlined in a software management article [5], where the article identifies “a recent update of the Chaos Report from the Standish Group, outlines a recipe for success that includes 10 items. The first three items are executive support, user involvement, and experienced project management” [5]. So project management is one of the 3 key factors to successful projects, and means I will need to ensure this project is correctly managed, so I can successfully achieve the project aim and objectives.

My project Gantt chart is documented under Appendices B.

I am now going to explain how the project objectives have been successfully completed by the project deadline date of 11 Jan 2015.

1. I wrote a project document outlining the details of the project, defining project objectives, scope, risks and approaches. I have constantly referred to this document to ensure the project progresses in the correct direction (Mid-Project Report).
2. I wrote a work plan outlining the project objectives, with deadlines for each objective (Gantt Chart).
3. I defined relevant resources for the project, outlining decisions made on technology, equipment and software applications to use, ensuring that I have tested equipment and software applications, and am competent with the use of the selected technologies, prior to the start of the development (required skills and resources).
4. I kept an eye on the project plan ensuring that objectives do not overrun past their completion date (where possible).
5. I stayed vigilant and alert to early warning signs of problems occurring in the project that may have resulted in the project being delayed and not meeting project deadlines.
6. I safeguarded against my project creeping outside of scope, so as new requirements were introduced during the development process, I had to ensure these are all still within available resources and overall aim and objectives of the project.
7. I managed risks as the project progressed, and as new risks were discovered, I had to evaluate them to ensure they do not cause a major problem to the project (Highlight Report).
8. I tried to keep my project supervisor informed of any major problems occurring during the project, and did at times, seek advice where necessary, to resolve major problems as early as possible.
   1. Relevance to target award

Software Engineering is defined by Ian Sommerville as an engineering discipline concerned with all aspects of software production (specification, development, validation and evolution), and goes on to say it is concerned with the practicalities of developing and delivering useful software [1].

My project aim is to develop and implement software engineering technologies, and to do this I had to explore the different software engineering techniques and decide which are best suited to tackling the software engineering task, and then develop and implement a piece of software that successfully meets the aim and objectives of the project.

This means the work I carried out during this project fits in with my target award MSc Computer Science (Software Engineering), because I applied software engineering models I have studied during my course such as agile to my software development. I also applied software engineering methodology I have studied during my course such as Inheritance and Encapsulation to my software development. I also applied the software engineering tools I have studied during my course such as unified modelling language (UML) to my software development. Lastly applied metrics such as cohesion, coupling, bugs etc. to my software development. By me exploring and applying these different software engineering techniques it allowed me to deliver useful software to ‘MSc Properties’ which in essence is Software Engineering.

* 1. **Required Resources and Skills**
     1. **Hardware**
* Operating System – Windows, Solaris, Linux or OS X;
* Processor – Intel® Core™ i5-4288U CPU @ 2.60GHz (or similar);
* Memory – 8.00 GB (or similar)
  + 1. **Software**
* Platform – Windows XP or higher (or similar);
* A JDK for Java 5 or later
* An Integrated Development Environment (NetBeans or similar)
* A concurrent version system (Git or similar)
* A bug tracking and testing tool (Bugzilla or similar)
* A web server (Apache or similar)
  + 1. **Access**

I will require access to the following:

* MySQL Server
  + 1. **Skills**
* Research skills
* Project management skills
* Report writing skills
* Ability to use Unified Modelling Language to model the distributed system
* Ability to write code in Java, HTML, Java Script, PHP and SQL.
* Ability to implement design patterns such as Observer
* Ability to use frameworks and API’s such as Spring and JRC respectively

I met these project resource and skill requirements, by ensuring I had the required hardware in place before development work began, I then downloaded the majority of the required software resources and of the ones I did download, I tested these to ensure they work appropriately. Once I carried out a literature review of the required skills, methods and methodologies I could employ to meet the project aims and objectives, I then undertook exercises to ensure that I have understood these methods and methodologies before development work began and if any problems arose during the development I attempted to seek assistance from my project supervisor to overcome these issues.

* 1. Ethics Approval

Ethics Approval is when a committee of University of Hertfordshire staff approve “any student undertaking a study involving the use of human participants which is undertaken as part of a programme of work for which the University of Hertfordshire is responsible for” [4].

My project will not require ethics approval because I did not undertake research that involved collecting data from human participants, and although my system does store business data which includes personal information, I used dummy information which replicates the personal information throughout the development.

1. Design
   1. Introduction

I am now going to discuss the design decisions I made during this project, and how I came to make these decisions.

* 1. Software Lifecycle

Considering the project aim and objectives, and also the project background, I believe an agile method is the best software process model to choose from, this is because in a fast moving business environment, software needs to be ready and available as quick as possible, and as original software requirements can quickly become out of date, it makes software developed useless very quick.

Also due to the type of software I was developing, the software process model chosen needed to provide rapid development and delivery of software, and with the conventional plan driven software process models, it can be difficult to do this because of the amount of documentation that needs to be created and signed off, and the lack of interleaving development stages makes it difficult to cope with quickly evolving requirements.

Furthermore, as there will be an office manager from ‘MSc Properties’ working with the project assisting the development, it would make sense in being able to deliver software quickly to allow for this to be evaluated and confirm the project is moving in the correct direction at each iteration of the agile lifecycle, as it can be difficult to gather exact system requirements from clients without going over the development process and having something to evaluate and add to or remove from to develop a system that successfully meets the project aim and objectives.

The agile method I have chosen is the Agile Scrum methodology, and although I decided to use this agile method, as explained previously in the project background a business analysis and system requirements exercise was carried out with an officer from ‘MSc Properties’, which allowed me to have a fairly strong idea of what functionality ‘MSc Properties’ required from the system to be developed, although these were likely to evolve and did.

The information gathered put me in a position where I was able to carry out a fair amount of design work for the system prior to the first development iteration. I then used the divide and conquer technique which allowed me to break the development into smaller pieces, and accomplish one or a number of the smaller problems with each iteration of the development process.

The breaking down of the development into smaller pieces enabled me to tackle each smaller task on its own and then combine the solutions to the smaller problems to provide a solution to the original problem, which meant providing a solution to the original problem was easier and more manageable.

Finally, as you can see from Fig 2, from “Software Quality & Agile Methods” article, the figure outlines the different quality assurance techniques, which will enable me to ensure that the software I produce for ‘MSc Properties’ is of a good quality, due to the quality assurance practices that occur during the agile software lifecycle, which will assist me to meet one aspect of the aim, which is to produce dependable software.

* 1. Modelling System Behaviour

I used a number of design techniques to assist in the modelling of the system behaviour and am now going to explain what design techniques I used and why I decided to use these for the development.

However due to the size of the project and due to the level of resources available, I decided that for me to successfully achieve as much of the project aim and objectives, I would not be able to develop diagrams to model the entire system behaviour, and instead I selected a sub set of the system functionality to model.

* + 1. Use Case Diagrams

For this development I decided to develop a number of use case diagrams which have been documented under Appendices C. By developing use case diagrams, this enabled me to identify the relationships between actors (roles within the system, for example a user of the ‘MSc Properties’ system) and use cases (functions within the system, for example creating a property), which allowed me to identify what actors was involved with which use cases.

The use case diagrams developed during the project were not only used to model the system behaviour, but I also used them to create test scripts for system testing, as the use case diagrams outlined the different functions that should be available within the system, and therefore can be used to carry out the black-box system testing, which will be explained in further detail in the implementation section of this report.

As you can see from Fig. 6, the use case diagrams developed within Appendices C, outline the different user (employee) interactions within the MSc Properties business (the system) for a create tenancy transaction, outlining the user interaction with the application, property and office records, to update the information depending on the tenancy that is being created. Furthermore, the use case diagram highlights the Create Rent Account use case is also included within the create tenancy use case diagram, because whenever a tenancy is created, the corresponding rent account will also be created.

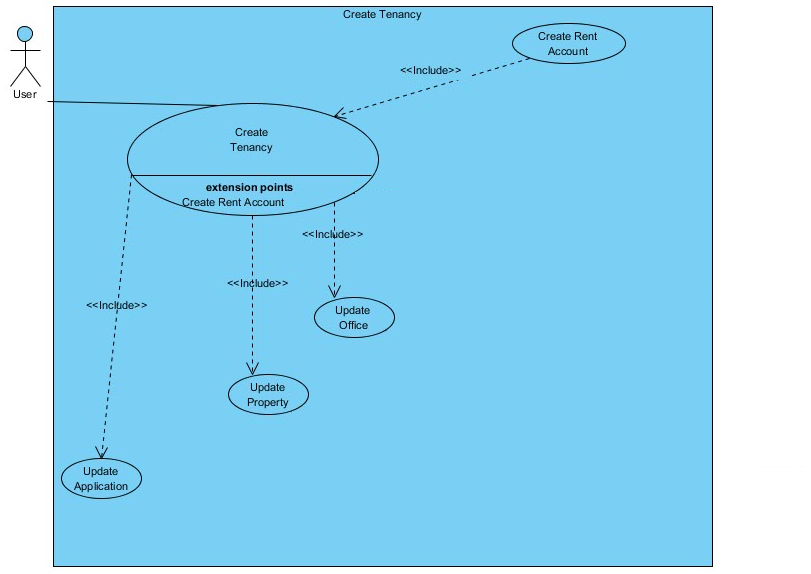


Fig. 6 – Create Tenancy Use Case Diagram

* + 1. Class Diagram

For this development I decided to develop a class diagram which have been documented under Appendices D. By developing a class diagram, it enabled me to visualise the structure of the system I intend to develop, and allows me to document the variables and methods of each class, and how classes are related to each other, for example composition, multiplicity, inheritance, etc.

As shown in Fig. 7, the Class Diagram visualises the fields for each class and relationships between these classes. Unfortunately, due to the scale of the project and the number of classes for the development, I had some issues documenting this correctly, so the relationships for the classes are not shown within the Appendices. Furthermore, there was some changes to the development as the development moved through each iteration cycle making some of the information within the class diagram incorrect, as it has evolved since the diagram was developed.

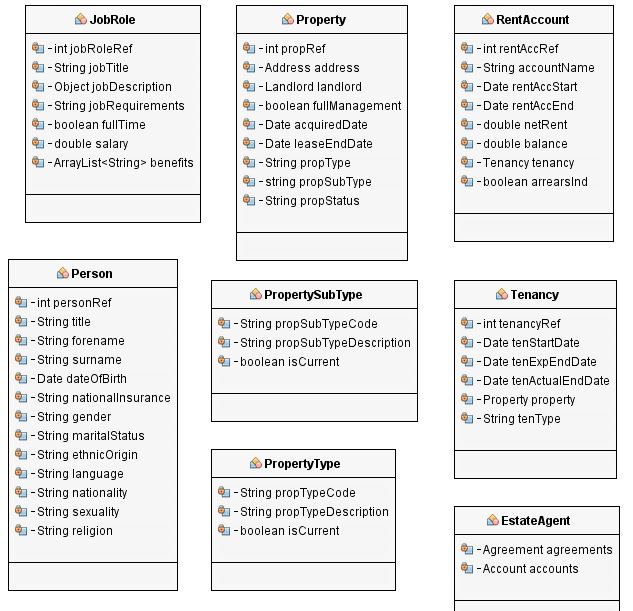


Fig. 7 – Extract from the Class Diagram

* + 1. Enhanced Entity Relationship Diagram

For this development I decided to develop an enhanced entity relationship diagram (ERD), which have been documented under Appendices E. An ERD is very similar to a class diagram, but instead of visualising the structure of the system, it visualises the structure of the database to be developed, and allows me to document the tables, columns and relationships between tables, for example one-to-one, one-to-many, many-to-one and many-to-many.

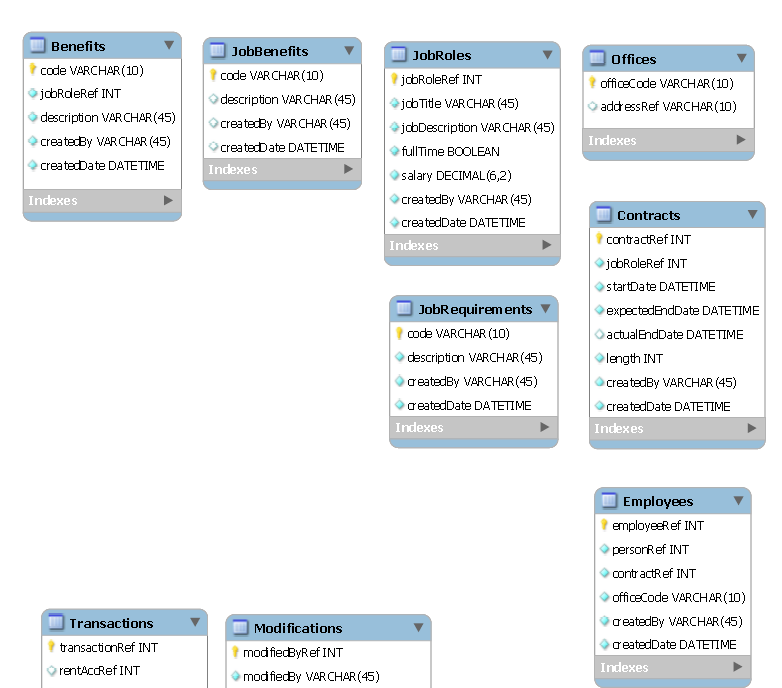


Fig. 8 –Extract from Enhanced Entity Relationship Diagram

As shown in Fig. 8, the EER diagram identified the different tables for the database management system, and outlines the different columns, their names, and the data type of each. Additionally, the EER diagram outlines the different relationships between each table.

Unfortunately, due to the scale of the project and the number of tables for the database management system, to manage the storage of data, I had some issues documenting this correctly, so the relationships for the tables are not shown within the Appendices. Furthermore, there was some changes to the development as the development moved through each iteration cycle making some of the information within the enhanced entity relationship diagram incorrect, as it has evolved since the diagram was developed.

* + 1. Sequence Diagrams

For this development I decided to develop sequence diagrams, which have been documented under Appendices F. By developing sequence diagrams, it enabled me to visualise the interaction between classes/objects within the system to be developed. Furthermore, the sequence diagrams also allowed me to produce test scripts along with the use case descriptions, for black box testing which will be explained in further detail in the implementation section of this report.

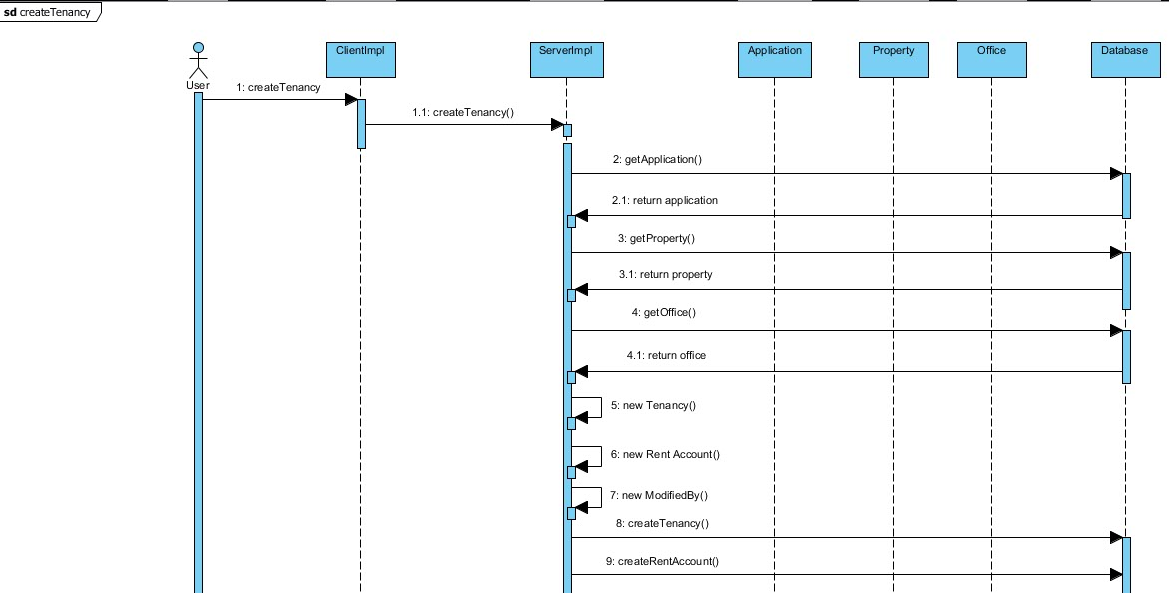
****

Fig. 9 – Create Tenancy Sequence Diagram (part 1)

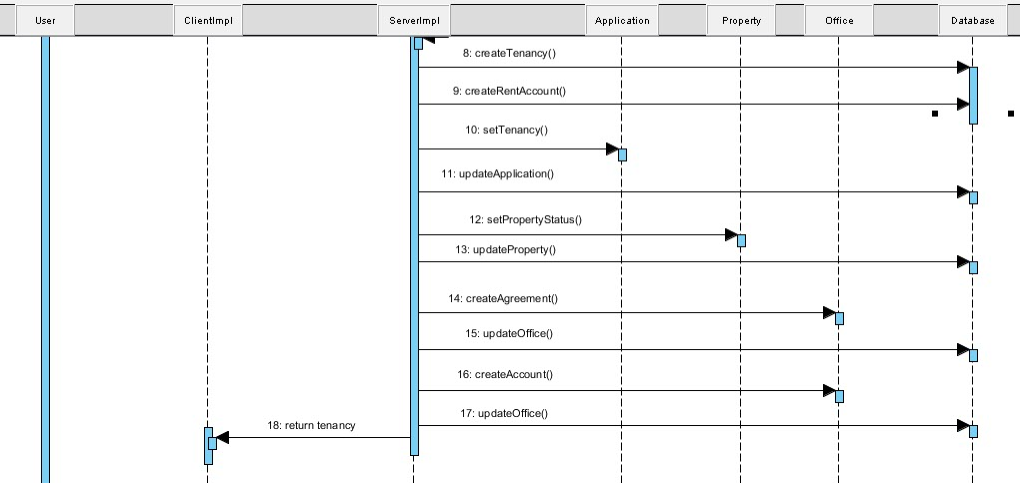
****

Fig. 10 – Create Tenancy Sequence Diagram (part 2)

As shown in Fig. 9 and Fig. 10 the sequence diagrams developed in Appendices F, outline the interactions between classes outlined in the class diagram in Appendices D, when performing a function, such as creating a tenancy. As you can see, the sequence diagram outlines, what objects invoke methods on other objects in order to perform the function, showing when system control is passed between objects.

* + 1. Storyboard

For this development I decided to develop a storyboard for the graphical user interface (GUI), which has been documented under Appendices G. The storyboard outlines the different screens I will be developing during this project and show how each of these flow on to one an another through user interaction with the system.

The storyboard will also provide an annotation of how I plan to develop the GUI with regards to different layout managers, components, font, colours, sizes etc. to be used.

As shown in Fig. 11, the storyboard outlines the different graphical user interfaces I aim to develop for users to interact with the system, additionally, the styoryboard outlines what software engineering technologies I am to employ to develop the GUI’s.

For the GUI implementation I would have used the below:

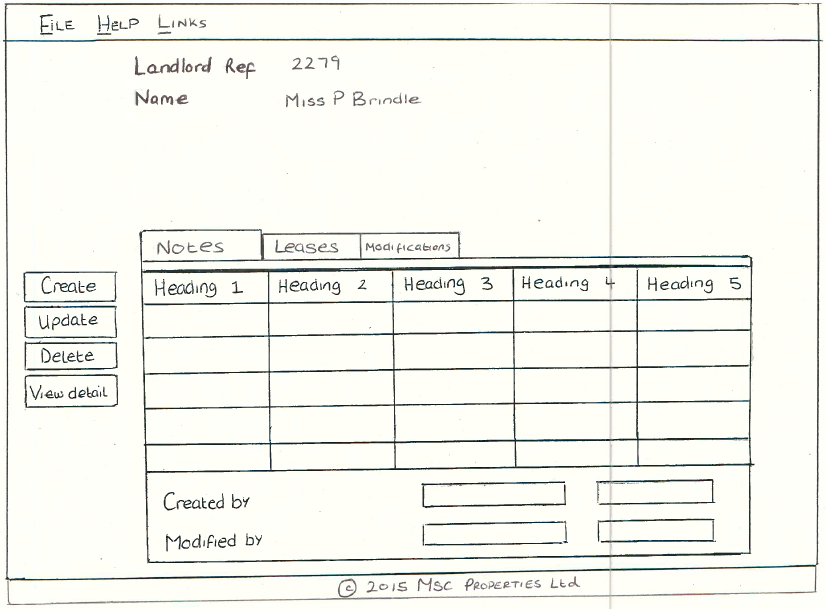
* JButtons for the Create, Update, Delete and View Details buttons. Also the use of Accelerator keys for the buttons to produce short cut keys.
* JTabbedPane to produce the Notes, Leases and Modification tabs
* Within each pane of the JTabbedPane, I would have a JTable, which would display the information from the Notes, Leases and Modifications lists in their own tab.
* JLabels for the LandlordRef, Name, CreatedBy and ModifiedBy, and for all 4 results, i.e 2279 and Miss P Brindle
* JMenuBar, with JMenu’s to produce the File, Help and Links menus, and add items to each of these menus. Also the use of Mnemonic keys to produce short cut keys.
* I would also place the tabbed pane, buttons and information area within their own panels (created in separate class, and use Actionisteners passed to them from the main panel in which these are added to, to reduce the coupling of the main frame and each of the components.

Fig. 11 – Landlord Details Screen

1. Implementation
   1. Introduction

I am now going to discuss the implementation decisions I made during this project to produce a distributed system from the design model produced in the previous section, and in doing so, successfully meet the project aim and objectives.

I decided the best way to go through the implementation decisions I made, was to take you through each of the iterations of the project development, as I have decided to adopt the Agile Scrum methodology.

And although with each iteration I carried out unit, integration, and system testing, I have decided to leave the explanation of testing to its own section, where I will then why certain decisions were made, and what the outcomes of the testing was.

* 1. Iteration Cycle 1

As this project involved developing a distributed system, for the first iteration I decided to implement the basic functionality of the system as a desktop version, which involved adopting a number of the design patterns I identified during my research for this project.

Although I implemented a desktop version of the system prior to implementing the networking functionality, I knew this was going to be a distributed system so I decided to split the desktop version of the system into two packages The two packages are a common package which would include any classes and interfaces that will be adopted by both the server package and the client package (once I implement networking functionality), and the server package which will include the model and controller of the model, for the system I am to implement.

* + 1. **Design Patterns**

As explained previously, I decided to adopt a number of design patterns to assist in producing software that is maintainable, dependable and usable. I am now going to discuss the design patterns I adopted during cycle 1 and why these were relevant to this project.

* + - 1. **Iterator Pattern**

During this project, the iterator pattern was one of the most frequent design patterns adopted, this is because as I was producing a system that will hold a lot of data, and one of the functions of the system will be to search the data using different search criteria, it would be crucial to be able to traverse over lists of data to extract data that matches the search criteria.

As you can see from Appendices H, I used a variety of Iterator patterns such as the while loop and the enhanced loop, and each provided its own benefits.

The while loop allows the programmer to loop over a set of statements as long as the condition within the while loop condition is true, which gives the programmer flexibility to supply any Boolean condition to determine if the while loop statement should be executed.

The enhanced for loop is similar to the standard for loop, however is less flexible, and should only be used when the programmer needs to loop over all of the elements within a list, and don’t need to know the index of the object being retrieved [43].

* + - 1. **Inheritance**

As you can see from the class diagram in Appendices E, I have adopted the design pattern inheritance, this is to re-use code and make use of an important programming technique called polymorphism. Inheritance is achieved through the use of parent (super classes) and child classes (sub classes), where the child class has a “is a” relationship, to the parent class, and the child class extends the parent class.

By implementing inheritance, it also makes the system easier to evolve, for example if ‘MSc Properties’ was to expand and create a new Agreement, the system will be able to be amended to add an additional AgreementImpl subclass which can then make use of any methods that use polymorphism without any changes to the existing code, an example of a method which makes use of polymorphism is in Appendices I.

* + - 1. **Object Composition**

As you can see from the class diagram in Appendices E, I have adopted the design pattern object composition, which similarly to inheritance allows for the reuse of code and polymorphism.

However, the difference with object composition to inheritance is that, a class, say class A, will contain a variable of another class, say class B, that implements functionality that is desired by class A, meaning that class A re-uses the code of class B, by implementing a “has a” relationship between class A and class B.

* + - 1. **Singleton Pattern**

As you can see from Appendices J, I have adopted the design pattern singleton pattern, this is to ensure that only one instantiation of an object can occur, to ensure that only 1 object is used throughout the system in all instances when this object is needed. I have documented an example of the implementation of the singleton pattern within Appendices J, however I have used the singleton pattern a few times within the development.

* + 1. **Graphical User Interface**

I decided to adopt the Java Swing and AWT frameworks to implement a graphical user interface to enable users of the system to interact with the system I am developing, and allow the process of using the system to be easier than a command line interface and improve the look and feel of the system.

Additionally, due to the windows, menus, and click, drag and drop functionality provided by a graphical user interface, it will improve the look and feel experience of the user in this type of system, instead of the command based functionality provided by a command line interface, due to the need to know the command and its parameters a user wants to perform.

For me to implement a graphical user interface I adopted a number of Swing and AWT functionality, such as containers, components, layout managers and event listeners.

Firstly, I adopted the use of containers such as JFrame, JPanel, and JDialog etc. to enable me to add a number of components to a container, and enable these components to be grouped together within the container they have been added to.

Secondly, I adopted the use of components such as JComboBox, JLabel, JTextField, etc. to enable the user to enter data into the system and provide a way for system state to be displayed back to the user.

Thirdly, I adopted a number of different layout managers such as GridBagLayout, BorderLayout, FlowLayout etc. enabling me the programmer to instruct the java virtual machine on how to present components that have been added to a component through the layout manager functionality.

Lastly, I used a number of listener objects such as ActionListeners and MouseListeners to enable user actions invoked on the graphical user interface components, to send a notification to the container(s) that have a listener object assigned to the component that has had an action invoked on them.

As part of the design stage, as explained in chapter 2, I adopted the storyboard design technique, which enabled me to document the design of a set of the main graphical user interfaces, within Appendices G. As you can see from Appendices G, I also provided annotations on what components and layout managers I will implement to develop the GUI.

Another concept I have adopted throughout the system, was the model-view-controller (MVC) architectural pattern, with the ClientImpl being the model for the (Client) system, and a main frame GUI such as the HomeFrame being the controller, and then components, that have been added to the main frame are the view such as a JTable or JTree.

The controller (main frame) would then deal with updating the view (panels of GUI components), as and when something happens in the model (ClientImpl class). The use of MVC within the GUI reduces the level of coupling between the model and the view, as the controller manages communication between the two.

The adoption of the Model-View-Controller, allows me to develop good software for ‘MSc Properties’ as it ensures that the model and the view are not coupled, meaning that the two components work independently of each other, which again allows me to meet aspects of the project aims as it ensures that the software is maintainable, and open to evolution due to the low coupling between components.

I have documented a large amount of the implementation decisions relating to the GUI and steps taken to develop the GUI within Appendices K.

By implementing a stand-alone application that enables ‘MSc Properties to manage their business data, will reduce time spent on carrying out tasks such as creating, updating and deleting business data, searching for data, and reporting on business data, as the tasks are carried out by the system, instead of by humans manually going through physical files to find the required information, and physically creating and updating data.

This shows that implementing software engineering technologies outlined within iteration 1 would free up time for employees to carry out other tasks other than managing their data, which will now be done by the system, and in doing so enable the business to streamline their business processes and improve their efficiency and effectiveness.

* 1. Iteration Cycle 2

For the second iteration of this project, I decided to implement the object relational mapping design pattern to store and manage the object-oriented objects in the database class of the system (‘MSc Properties’ business data), into the MySQL database I am going to create.

I need to make use of an object relational mapping design pattern because as you can see from the class diagram in Appendices D, the data stored in the Database class is non-scalar values, however as you can see from the enhanced entity-relationship diagram in Appendices E, the data stored in the MySQL database is scaler values and therefore the non-scaler values need a way to be broken down and stored in the MySQL database.

For the project, I had to create the database that was designed in the enhanced entity-relationship diagram using MySQL Workbench, and set up the settings for the database, such as primary keys, foreign keys, column data types and enforce referential integrity, which will ensure that any data stored in the database has to meet the rules set out by the relationships, meaning a foreign key value of a table, has to be present in the related table as a primary key.

Once the database was created I then had to use an application program interface (API) to allow the system to interact with the MySQL database. I need to use an API for the application and MySQL database because the two parts don’t know how to communicate with each other, and the API sets out how communication between these two take place.

The API I will use for this project is Java Database Connectivity (JDBC) and is part of the Java Standard Edition platform. Once I had created the MySQL database, I then had to implement a number of JDBC techniques to enable me to successfully achieve the project aim and objects associated with the MySQL database, and these are listed below and documented in Appendices L.

* Connecting to the database
* Loading System data at start up (Read)
* Create, Update and Delete Data.

As you can see from Appendices L, I use a number of methods to allow my Database class to connect to the database, and allow for objects created, updated and deleted within the system, to then be updated within the MySQL database with the usual database create, read, update and delete (CRUD) functions.

Furthermore, as explained in Appendices L, the loading of system data was one of the more difficult tasks, as I had to ensure that no objects were loaded up prior to an object that the loaded object is dependent on was loaded, and also needed to ensure all system elements such as title codes, religion codes was loaded up first.

This is because, loading objects which are dependent on other objects prior to the dependent objects being loaded, can cause issues for the system load, as there is if statements preventing objects from being created if the elements they are trying to be created with, does not already exist in the system.

By going through iteration cycle 2 and connecting the stand-alone application to the MySQL database, it will provide a storage, retrieval and updating mechanism for the data stored for ‘MSc Properties’ within the system, and additionally provide a backup in case of system downtime due to the system crashing or system upgrade. This improves the robustness of the system developed, and in doing so ensures that ‘MSc Properties’ data will not be lost if the system crashes.

Although the implementation software engineering technologies outlined within iteration 2 would not directly improve the business processes, and in doing so improve their efficiency and effectiveness, as the functionality available to the users of the stand-alone application are the same as the previous iteration, but it does improve the distributed systems usability, as the server does not need to be running all the time with no interruptions, as the system can be shut down and loaded up, with no effect to the system data.

* 1. **Iteration Cycle 3**

For the third iteration of this project, I decided to implement the networking functionality, to enable a user of the system to have access to data and documents stored within the system, where the system is not necessarily local to them.

From the research undertaken, I decided to implement remote procedural call functionality, and as I decided to write the project code in Java, making use of the object-oriented concepts, the Java API, Remote Method Invocation (RMI) was the best decision for me as this enables the user in one Java Virtual Machine (JVM), which may or may not be remote to the server, to be able to invoke a method on a server object in another JVM.

Furthermore, instead of implementing the sockets, threads, and serialization/marshalling of objects, by adopting RMI it allows the programmer to not worry about these aspects of networking as the RMI framework takes care of this for the programmer.

For me to implement RMI there was a number of steps I needed to take, which are documented within Appendices M, and are listed below:

* Set up Server
* Set up Client
* Push vs Pull
  + 1. **Push vs Pull**

As outlined previously in the literature review carried out, there is two concepts for a distributed system to manage data, which outline the way a client and server interact with each other and initiate tasks, these are called the push model and the pull model.

For this project I decided to implement both forms of client server interaction, this is because when a client wants to perform an action, I decided to implement the pull model, this meant that clients of the ‘MSc Properties’ system will request for the server to perform an action, so the client is initiating a request, and the server is responding to that request, by performing some action and returning information to the client. Therefore, the client is pulling from the Server, and the implementation of the pull model and how this was implemented is documented in Appendices M.

However, I also decided to implement the push model, which meant that the server of the ‘MSc Properties’ system will initiate an action through a push to the client, and then the client will deal with this as required. The pull model is only used when another client updates an agreement, or rent account, which can then be reflected on all client home screens, and is used in conjunction with the observer pattern, which is discussed later in this report, and documented in Appendices O.

By implementing the pull model, it reduces network traffic as the server does not need to send all updates to every client, and the client on requests information when it needs it, thus only receiving information it needs when it needs it.

* + 1. **Local Reference vs Remote Reference**

With Remote Procedural Call functionality, there is two big concepts with regards to local and remote references. This is because, a client server implementation, can either have the client create an object locally and pass a copy of this to the server for a copy to be stored locally (local reference), or the client can request the creation of an object by the server (remote reference), and a remote reference to the object is the passed back to the client.

For this development, I decided to implement the remote reference version of RMI implementation, this is because as there is only one copy, it is easier to keep track of updates to objects, as the client is able to invoke methods on these remote object references, and then the changes can be seen by any client who has or requests a remote reference of the object.

Whereas with the local reference, an update to a local method, is not seen by the server (nor any other client), and similarly, an update made by the server is not seen by the client, meaning that each update has to be pushed to all clients and the server (if it was a client who made the update), and it then becomes difficult to keep up-to-date records of each local reference (with clients and the server).

By implementing the networking functionality outlined in iteration cycle 3, converting the stand-alone application into a distributed system, enables multiple ‘MSc Properties’ employees to connect to the distributed system at any one time, and also enable the users to connect to the server for the distributed system from any location, whether it be remote to the server or not. Additionally, users will be able to share data at the same time, instead of having to wait for an employee who may currently be looking at the physical file, and data will be stored in the same place, meaning it can be located easier.

This shows that implementing software engineering technologies outlined within iteration 3 would enable the sharing of data between employees that are remote, improving distribution of data and version control of data, and in doing so enable the business to improve their data exchange processes and improve their efficiency and effectiveness.

* 1. **Iteration Cycle 4**

To implement the document management functionality, I decided to write my own document management module that would deal with storing documents and version control, instead of using a document management framework such as Apache JackRabbit.

To do this I firstly created a Document class which stored a file and any previous versions of that file. I then had to implement methods both on the Client side and the Server side, that will convert a document into an array of bytes, this allows for the array of bytes to be passed between the client and server and then reconstructed at the other side, for the client or user to either view or save depending on whether a document is being uploaded to the server or downloaded by a client.

The implementation of the document management system, has been documented within Appendices N, and the stages undertaken is listed below:

* Uploading a document to the server
* Downloading a document from the server
* Version Control

By implementing the document management functionality outlined in iteration cycle 4, it enables multiple ‘MSc Properties’ employees to view the same document at the same time, from any location. Additionally, users will be able to locate documents quickly as documents are stored in the same place.

This shows that implementing software engineering technologies outlined within iteration 4 would enable the sharing of documents between employees that are remote, improving distribution of documents and version control of documents, and in doing so enable the business to improve their document transfer processes and improve their efficiency and effectiveness.

* 1. **Iteration Cycle 5**

For the next iteration I decided to implement the observable pattern, which will be used to update a client of ‘MSc Properties’ graphical user interface (GUI) when an agreement or rent account is updated, without the ClientImpl class from the Client package actually knowing about the GUI.

I decided to implement the push model of the observable pattern, instead of the pull model that the client server data exchange is predominantly based on (as explained in iteration cycle 3 – push vs pull), this is because the pull variant can be quite costly, because each observer will invoke the method to pull through the current state even if there have not been any changes, whereas the push variant only updates observers when an update has occurred. This selection was best for this type of implementation because there will not be a frequent number of updates to ‘MSc Properties’ agreements or rent accounts so it is unlikely the observers will need to be updated all the time.

To implement the observer design pattern, there was a number of steps I had to take, which has been documented in Appendices O.

By implementing the observer design pattern outlined in iteration cycle 5, it provides a live feed of 10 tenancies, 10 leases and 10 rent accounts in an order list using either the expected end date or balance, ensuring that the ‘MSc Properties’ employees are aware of the tenancies and leases that are closest to ending, and the rent accounts with the highest balance, without employees having to search through all of the tenancies, leases and rent accounts to produce these lists, and in doing so frees up the employees time to enable them to undertake other tasks.

This shows that implementing software engineering technologies outlined within iteration 5 would free up employee’s time to carry out other tasks, and in doing so enable the business to improve their business processes and improve their efficiency and effectiveness.

* 1. **Iteration Cycle 6**

The sixth iteration cycle of this development was to automate a number of tasks that have to be carried out by ‘MSc Properties’ officers. To do this I decided to use the Java class TimerTask, and created a class called TaskGenerator, which extended TimerTask class from the Java util package.

I have documented my implementation of the task scheduling functionality within Appendices P.

By implementing the task scheduling functionality, it means that an ‘MSc Properties’ employee will not have to manually create rent transactions for each tenancy every month (the start date of the tenancy determines when the rent is charged to the rent account), or the manager will not have to generate monthly reports each month, which will reduce the work load of ‘MSc Properties’ employees, but still provide the monthly statistics and deal with certain tasks that were previously done manually.

This shows that implementing software engineering technologies outlined within iteration 6 would free up employee’s time to carry out other tasks, and in doing so enable the business to improve their business processes and improve their efficiency and effectiveness.

* 1. Iteration Cycle 7

The seventh iteration cycle of the development was to improve the security of the application and implement login functionality and user privileges functionality. To do this I decided to implement a proxy server, which will act as a communicator between the server and client, ensuring that each client requesting services from the server have the required user privileges. Additionally, to increase the security of the application, I decided to implement a LoginImpl class which acts as the login portal and requires all clients logging in to the system, to go through this class.

For me to implement these suggested security amendments, it required me to amend how the server sets up the RMI functionality explained in Iteration Cycle 3, and amend how the client registers with the server, and as explained, develop a Login class which will act as the login portal, for the client to interact with to register with the server.

I have documented the implementation of improving the security for the application within Appendices Q, and is broken down in to three sections listed below:

* Proxy Server – User Privileges Functionality
* Login Functionality
* Forgot Password Functionality

By implementing the security improvements outlined in iteration cycle 7, preventing unauthorised access and unauthorised service requests on the server, ensuring that ‘MSc Properties’ control who is able to carry out tasks and what tasks people are able to carry out, instead of having to rely on employee integrity to not carry out a task they are not authorised to do.

Although the implementation of software engineering technologies outlined within iteration 7 would not directly improve the business processes, and in doing so improve their efficiency and effectiveness, as the functionality available to users of the distributed system are the same as the previous iteration, but it does improve the distributed systems security, as users will not be able to invoke methods on the server that they do not have permission to perform, and therefore providing control over who is able to perform what functions.

* 1. Iteration Cycle 8

The eighth iteration cycle of this development was to develop and implement a website to allow MSc Properties to advertise services they provide, and enable potential customers to submit a service request, notifying MSc Properties of the customer interest.

I used a number of software engineering technologies such as running a web server to manage website web pages, HTML (mark-up language, used to develop HTML web pages), PHP (server side scripting language) and JavaScript (programming language for HTML), and by adopting these software engineering technologies I was able to implement the website.

By adopting the above software engineering technologies, I was able to advertise services such as properties available to rent, letting services offered, career options through job vacancies, and lastly office information, such as contact details and address, of which the information published through the website is saved within a database management system, which in turn is updated by MSc Properties employees through the system developed in previous iterations.

Additionally, I was able to implement functionality which enables users to register their interest in services provided by MSc Properties, of which an email is then sent to the office of interest, which is then picked up by MSc Properties employees, for contact to be initiated with the customer and progress any service request. I have documented the implementation of the website within Appendices R, and is broken down in to three sections listed below:

* Managing Communication with MySQL Server
* Advertise MSc Properties Services
* Service Requests

By implementing the website outlined in iteration cycle 8, it increases the advertising of services, so instead of being limited to the local advertising techniques of local newspapers and shop windows, and needed to constantly update the advertisements, the website will be dynamically updated, meaning that properties are advertised as and when they should be, and taken down as soon as they are no longer available. Additionally, service requests are rooted to the required office, along with the service request details, and does not require any staff to make note of the service request as they are all recorded within the mailbox.

This shows that implementing software engineering technologies outlined within iteration 8 would enable the advertising of ‘MSc Properties’ services to a potentially wider group of customers, and additionally enable these potential customers to submit a service request which is rooted to the required office, notifying them of this request, ensuring that ‘MSc Properties is able to deal with the request, and in doing so enable the business to improve their customer base and improve their efficiency and effectiveness for advertising and receiving service requests.

* 1. **Iteration Cycle 9**

This final iteration will deal with deployment of the client package, enabling users of MSc Properties to be able to run the client package at a remote location to the server, allowing users to log in to the system and communicate with the server from a remote location. To carry out the deployment of the client Java code developed, I have two main ways to do this which are:

* Java Web Start
* Applet / Java Plug-in

Unfortunately, due to time constraints I was unable to finish the project off, with regards to the deployment of the client package through Java deployment technologies such as Java Web Start or Applet / Java Plug-in.

1. Testing

As explained previously in iteration cycle 1, I was attempting to carry out unit testing, integration testing and system testing during each iteration cycle, but I decided to document the majority of my testing results in my Appendices S, T & U, with a description within this Testing section of the report, on what decisions was made during the project and why, and what the outcome of the results were.

* 1. **Unit Testing**

For the unit testing, I decided to use JUnit testing, which allowed me to produce automated tests that can be run and assess the state changes of an object and ensure that classes within the development are working as expected, and can be run whenever there are amendments to the source code to ensure the semantics of classes are correct, and working as they should. Also by writing JUnit tests, it meant that I was not writing long testing classes, and having to amend the test classes with each amendment to a class.

For the unit testing I adopted the white box testing approach, which meant that I coded my test classes with the knowledge of how my classes were programmed, meaning I could take advantage of knowing where the boundaries of if statements were, etc., and adopt a testing policy where I will test at boundaries as well as using incorrect and correct data to test, and set pre and post conditions, which will either pass or fail, and determine if the units within the development are working as expected from the actual source code.

As you can see from Appendices S, I have carried out unit testing for each class, and to do this, as explained I examined the source code for each class created, and developed pre and post conditions for each unit of the development to pass, as you can see from Fig. 11, which allows me to ensure that each unit is working in isolation as expected.



Fig. 11 – Extract from AccountTest class – testCreateTransaction() part 1

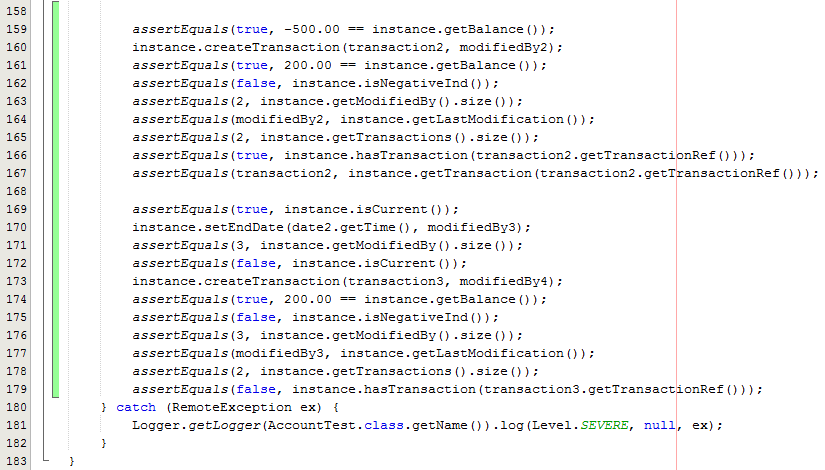


Fig. 12 – Extract from AccountTest class – testCreateTransaction() part 2

As you can see from Fig. 11, for the createTransaction unit test and other Account Class unit tests, I have initialized an Account object and any other objects required for the createTransaction method, of which Transaction and ModifiedBy objects were required. Before invoking the createTransaction method, I check the state of Account object fields by invoking JUnit methods such as AssertEquals. By invoking these JUnit methods, I am able to determine the pre-condition state is what I expect it to be, and allow me to ensure that the method is working as expected changing from one state to another.

I then invoke the createTransaction method and check the state of the Account object fields again to ensure the actual result is equal to the expected result. I then go through the process again of checking the state, creating a different type of transaction, and then checking the state again to ensure the pre and post conditions have been met. I then change the state of the account object by invoking the setEndDate() method on the Account object, and supply an end date that will ensure the Account object is no longer current. I then check the state of the Account object, invoke the createTransaction method and then check the state of the Account object again, but because the Account object is no longer current the transaction should not have been added and therefore the state of the object should be the same as after the previous transaction was added, with the exception of the Account no longer being current.

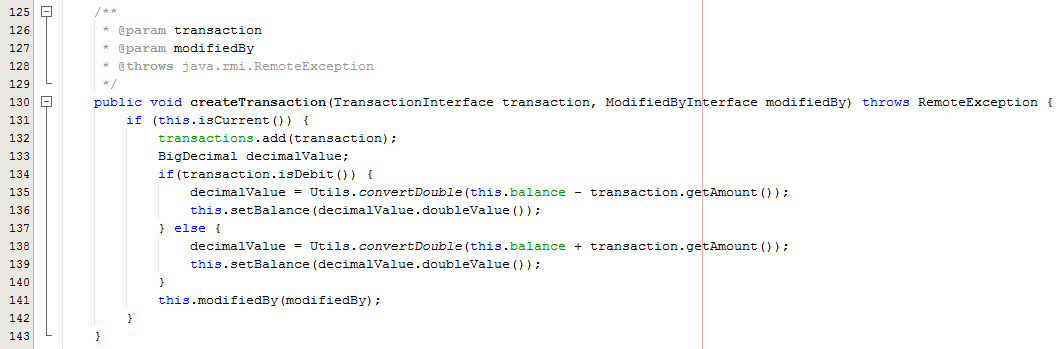


Fig. 13 – Extract from Account class – createTransaction

As you can see from Fig. 13, I was able to ensure the if statement checking if the account object is current was working correctly, by invoking the createTransaction method on an account object that is both current and not current, I was then able to ensure the transaction was being added to the list of Transaction objects within the Account object, by checking the size of the list, and that the Account has the transaction (Account.hasTransaction()) after the createTransaction method was invoked. I was then able to check that the balance is being amended correctly, whether it was a debit or credit transaction being created for the Account object, by checking the balance value, and if the account was in a negative. Finally, I was able to ensure that the modification to the account was being saved to the Account object correctly, through checking the size of the list of ModifiedBy objects for the Account object, and getting the last ModifiedBy object within the list.

By carrying out the unit testing in this way, I am ensuring that the source code is doing what I am expecting it to do, and is working with both incorrect and correct parameter values passed to the object.

I carried out this form of testing for each of the server side concrete classes, using the white box testing techniques where I look at the code for the method being tested and assess actual results from code compilation against expected results from a given input. As you can see from Appendices S, each class successfully passed the unit testes written in the above fashion, checking the post and pre conditions have been met, ensuring that each class is able to track who created the object, and who has modified the object, it is also able to add notes to each object or a comment, as well as store the information associated with each object.

* 1. **Integration Testing**

For the integration testing, I decided to adopt the Big Bang testing along with white box testing techniques, instead of top-down, or bottom-up testing, this is because I only carried out integration testing after the final iteration due to time constraints, so although this method requires nearly all of the units to be developed and integrated, this was the case when I undertook integration testing.

So although the normal process of a large agile software development, is to carry out unit testing and then either top-down or bottom up integration testing after each iteration cycle (testing a smaller number of integrated units, either from the highest priority down, or lowest priority up), and then system testing would occur, I decided it would be best for me to carry out integration testing at the final iteration of my project, which meant all of the modules was basically complete, and therefore could use the big bang method.

For the integration testing I decided to test all of the server side classes except for the Server class. I decided to use the Database class as the basis for my integration testing, as the database class dealt with storing all of the system data, and loading all of the system data at system start up. To carry out the integration testing I created a test class which created a database object, and my test class was then a client of the Database class, which meant that I could test a large amount of the server functionality to ensure the system (excluding the server class) worked.

As you can see from Appendices T, the Database class was able to create all of the individual objects such as People, Applications, Tenancies, etc. and write these to the MySQL database. The Database class is also able to retrieve objects from the MySQL database class and load these up, and update them as necessary, which in turn will be written to the MySQL database.

Furthermore, the database class can delete any of these objects that have been created, as long as there has been no modification to the object, such as an update, or an object being created for that object, such as a note object.

Lastly as explained, at system start-up the database class loads up all the system data from the database correctly, with all objects being loaded up as they were for the last server session, meaning that if the system was to experience down time due to a crash or upgrade, the system is able to be started back up from the previous state (excluding any uncommitted processes that were in the process of being saved to the database). All of the integration tests are shown within Appendices T.

* 1. **System Testing**

For the system testing, I decided to use black box testing, meaning that the testing process would not use knowledge of how the program was coded, but instead use the use case descriptions, and sequence diagrams, along with the system functionality drawn up in the analysis and design stages of the development.

Although I was unable to complete the documentation of all of the system areas due to the size of the development, and level of resources for the development, I carried out an informal analysis and design process where I mapped out use cases for the system, and mapped out how classes will interact with each other to perform tasks, and therefore was able to use these along with the formal documents I have developed within Appendices A, C and D to achieve the system testing.

As you can see from Appendices A, the system I am to develop must perform the below functions:

* Login functionality
* Password reset functionality
* User permissions functionality
* Home Screen – Tenancy/Lease/Rent Account live feed functionality
* Create functionality
* Update functionality
* Delete functionality
* Search functionality
* Reporting Functionality
* Document Management functionality
* Advertise Services through website
* Service Request through website

This meant that I had to look over the use cases and sequence diagrams I had developed for the system, and identify what tasks take place within each function, and what should be the expected outcome given a particular input to the system.

As shown within the create functionality section within Appendices U, when testing the createTenancy functionality, I looked at the different objects relating to the createTenancy method such as the Application, Property and Office for the Tenancy object as well as the newly created Rent Account object, ensuring that each of these elements where updated/created as expected given a particular input.

As shown in Fig. 14, the create tenancy use case diagram outlines that when the create tenancy functionality occurs, then the create tenancy use case is performed, which in turn means that the update application, property and office use case is performed, and the create rent account use case is performed. From this, it shows that when carrying out my system testing I need to ensure that the application, property and office is updated and the rent account is created.

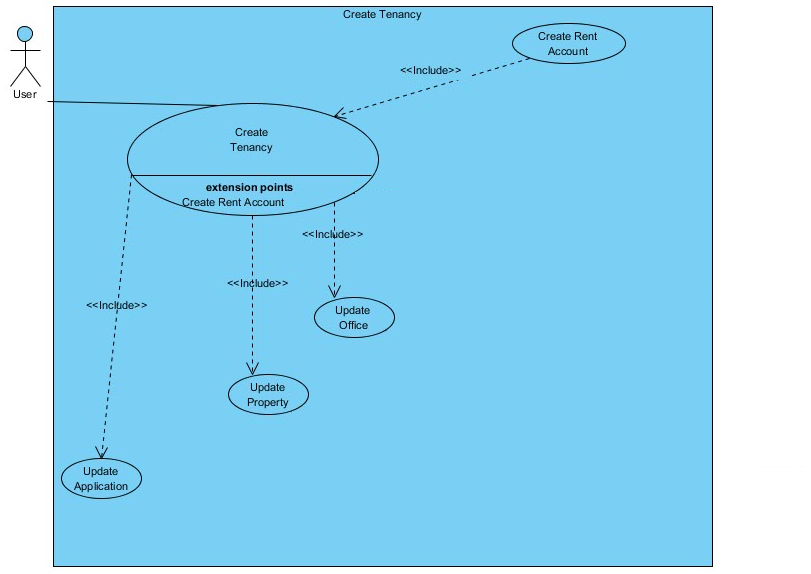
****

Fig. 14 – Create Tenancy Use Case Diagram

Additionally, as you can see from Fig. 15 and Fig. 16, the sequence diagram for the create tenancy method outlines in more detail what should occur when the create tenancy functionality occurs.

As shown in Fig. 15, when the user invokes the create tenancy functionality, it passes the required information to the ClientImpl object, the ClientImpl then invokes the create tenancy functionality and passes the required information to the server, where the server processes the information and gets the required application, property and office object from the database, and then creates a new tenancy, rent account and modified by object using the information supplied from the client.

Once the tenancy, rent account and modified by objects have been created by the server, the server then saves the tenancy and rent account on the database. Now moving to Fig. 16, the server object should set the tenancy for the application object, which should assign the tenancy ref, application address and application status, and then updates the application on the database. I then set the status for the property, and update the property on the database, and finally create the tenancy agreement and rent account for the office, and update the office on the database, and return the tenancy object to the client.

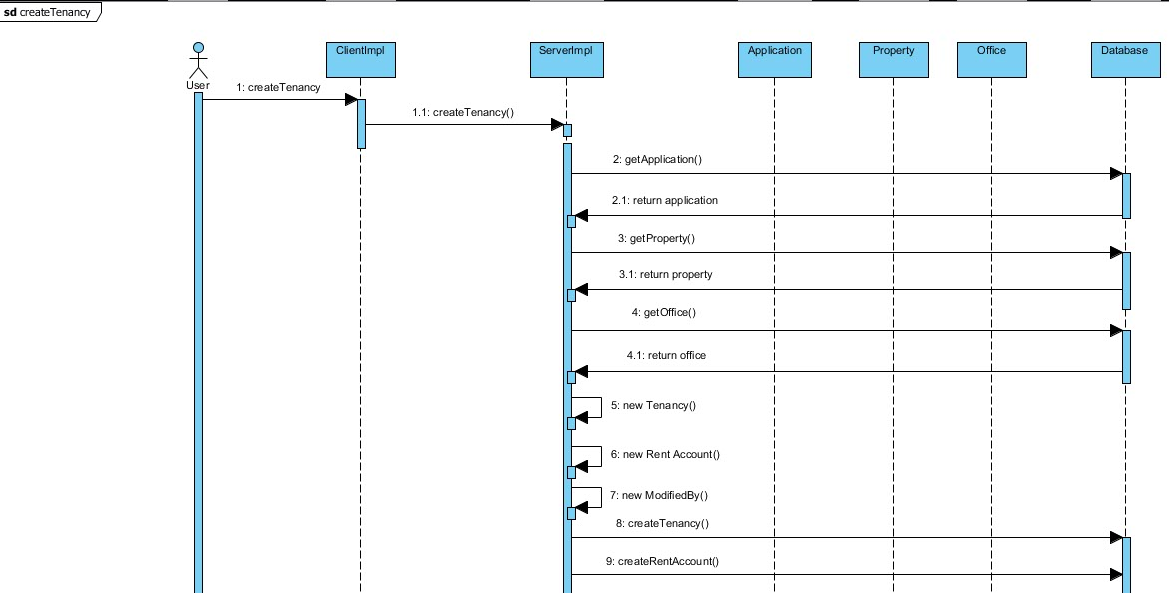
****

Fig. 15 – Create Tenancy Sequence Diagram – part 1

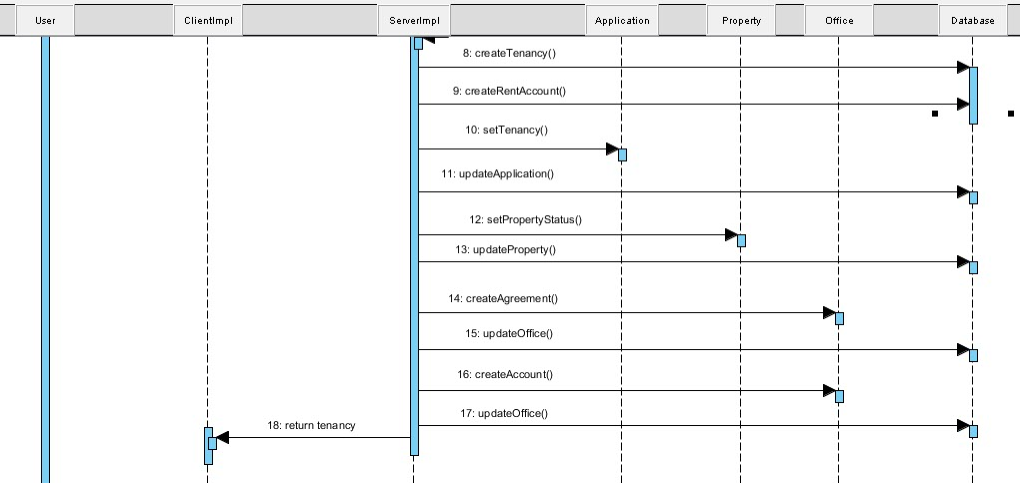
****

Fig. 16 – Create Tenancy Sequence Diagram – part 2

As you can see from the testing outlined within the create functionality section within Appendices U, I have successfully implemented the create tenancy functionality, and have ensured that the objects associated with the newly created tenancy have been updated as required.

The system testing outlined in Appendices U, shows that I have successfully implemented all of the functional requirements outlined in Appendices A, and in doing so have developed a distributed system using software engineering technologies that enables ‘MSc Properties’ to carry out their business processes, such as creating business data (people, addresses, applications, properties, tenancies, leases, contracts, accounts, etc.), and managing the business data, enabling for users to search, report, update, and delete business data.

Additionally, the distributed system developed is able to manage ‘MSc Properties’ documents, ensuring that versions of documents are managed, and users can view these documents at any time.

Furthermore, the system is able to restrict user access to the distributed system through the login functionality, and allow users to reset their password when they are locked out of the system. Also the distributed system prevents users from accessing unauthorised information or performing unauthorised tasks through the proxy server implemented, ensuring that the system is protected from unauthorised user actions.

Lastly, the website developed during the development is able to advertise ‘MSc Properties’ services, and dynamically be updated through the MySQL database (of which is updated by the users of the distributed system), and also enable potential customers to submit service requests for advertised services.

1. Conclusion

From the analysis, design, implementation and testing sections outlined within this report, I believe that I have successfully achieved the main core objectives (no risk assessment or user manual developed), and all of the advanced objectives, to a satisfactory standard. Although with some design decision changes and additional resources I would have been able to achieve the objectives to a higher standard, and in doing so, produce a distributed system of higher quality, which in turn will produce better performance and more streamlined business processes for ‘MSc Properties’ enabling the business to be more efficient and effective with their business processes.

* 1. **Objectives Achieved**

Firstly, I believe that I have successfully carried out an analysis of ‘MSc Properties’ current business processes, which have enabled me to draw up a number of functional and non-functional requirements for the distributed system that was developed for this project. However due to the level of resources for the project, meaning strict time constraints, I was unable to provide an in-depth list of system requirements, which would have given me a fuller understanding of the system to be designed and developed.

Secondly, the literature review carried out enabled me to get different views of different experts within Software Engineering, and present a number of different software engineering technologies, which assisted me with design and implementation choices made during the project. However, one of the areas within the literature review I believe I have not done so well in, is producing a discussion on the different software engineering technologies presented for me to implement the required functionality of the distributed system developed, and not taking into account how different distributed systems with similar functionality, have implemented these systems.

Thirdly, I believe the design stage of the project was not as good as it could have been, as I spent a large amount of time going through the different design decisions and producing the different design documents for the entire system, which meant I strayed from the agile software process model I adopted, when I should have concentrated on design decisions and developing design documents for each iteration, during that iteration, which would have meant that the decisions I made for each iteration was more accurate and less likely to change, whereas with the design section carried out during the early stages of the development, the decisions had to be changed as I moved through each iteration, making some of the design plans from the early stages redundant.

However, although I believe the design section could have been managed better, the design decisions made throughout the project have enabled me to successfully ensure all of the required resources was available throughout the project, the distribution mechanism I implemented was selected, a suitable model for the system to be developed was produced, but unfortunately I was unable to carry out the formal risk assessment, at different project intervals, nor was I able to produce test scripts for the entire system.

Furthermore, I believe I have successfully developed a distributed system which deals with the main functionality requirements set out in the analysis stage and in doing so successfully achieved the project objectives regarding the development of system modules, allowing users to create, update, delete, search and report on ‘MSc Properties’ business data, with user privileges preventing unauthorised actions being completed.

Additionally, the application connects to a database management system to store business data, has a log in facility, with password reset functionality, has a home screen which provides a live feed of tenancies, leases and rent accounts in an ordered list and carries out automated business tasks, which would normally be carried out manually. Moreover, I have successfully implemented a document management system, with version control and a website which advertises the ‘MSc Properties’ business services and allows potential customers to submit a service request for both renting and letting.

Lastly, although I believe the testing section could have been done considerably better the testing carried out has enabled me to ensure that the individual units of the system works independently, the integration testing ensured the database is able to load all system data from the MySQL database, and is able to create update and delete data, and finally the system testing showed that the main system requirements were successfully implemented and the functionality is working as expected, however the graphical user interface has some issues with regards to the layout of components, and the populating of JTable data without actually resizing the frame that the JTable is contained within.

By successfully achieving the main core objectives and all of the advanced objectives, it has enabled me to implement a distributed system that offers the below benefits to ‘MSc Properties’:

* Reduces time to carry out certain tasks, and automates other tasks, freeing up ‘MSc Properties’ employees to carry out other tasks
* Improve robustness of system ensuring data is accurate
* Sharing of resources such as business data and documents that are stored in a central location, meaning multiple ‘MSc Properties’ employee can view business data and documents at the same time, and in doing so improve data exchange methods.
* Version control of data and documents, ensuring all of ‘MSc Properties’ data and documents are up-to-date at all time and any record held by an employee at any time is the most up-to-date (providing the user refreshes the view).
* Tracking of tasks completed, and by what employee, ensuring that there is an audit of tasks completed within the business ensuring accountability for actions performed by employees.
* Control over user actions, ensuring only authorised users are able to access ‘MSc Properties’ data and documents, and additionally employees are only able to carry out tasks they have the user privileges to perform, prevented tasks from being performed from unauthorised people.

From the benefits offered to ‘MSc Properties’ by implementing the distributed system developed during this project, I believe I have been able to determine that the development and implementation of software engineering technologies, will increase business productivity and performance of ‘MSc Properties’, and in doing so have successfully achieved the overall aim of the project.

* 1. **Choice of Methodology**

The software engineering technology choices for the development were good and enabled me to implement the main functionality for the system, but could have been bettered further research into the advantages and disadvantages of each software engineering technologies in certain circumstances.

Firstly, the choice of programming language I decided to code my application in was Java, and was the best choice as it allows compiled code to run on all platforms, meaning that the code is usable across a wide range of devices, and as ‘MSc Properties’ may have different computer set ups at different sites, it allows for the system to still work on the different set ups.

Secondly, the choice of Remote Method Invocation (RMI) as networking functionality, was the best because I was coding in Java and as explained before, RMI allows for objects to be passed as values, and therefore behaviour of an object can be passed across a network from the server to the client, instead of just primitive values such as integers. Also RMI makes it easier for the programmer as the programmer does not need to manage the sockets, threads and serialization of objects, making the development of the system easier.

Thirdly, the choice to use a concurrent version system was a very good one, and is highlighted when I encountered some issues during the implementation stage which required me to roll back to a previous stage. And if I was just using external back-ups with no version control I would have not been able to roll back to a previous version to resolve the issues encountered, unless I had that exact version saved as a backup. Additionally, it provides a way to easily pass around development files to other developers, and ensure version control for these files as the concurrent version system has a central repository where documents are pulled from, and pushed to, to update the repository.

However, with regards to the security aspects of the distributed system, I would implement the user privileges and login functionality differently, by implementing the Java Authentication and Authorization Service, which again would prevent unauthorised users access to the system. Instead of using Boolean values stored to the user account, JAAS makes use of security managers, policy files with user privileges and password files with username and password combinations server-side, along with the proxy pattern I have already implemented for RMI set-up, where users have to communicate through a proxy server, to communicate with the actual server object.

Furthermore, to implement the task scheduling functionality I used the TimerTask and TaskGenerator classes, which as explained within the report, restricts the task scheduling functionality to be hard coded in to the system, whereas if I had implemented the Quartz task scheduling functionality, it would have enabled me to produce functionality that allows ‘MSc Properties’ employees to schedule jobs manually (in case of system down time resulting in automated tasks not being completed), which would improve the current implementation, because at current, if the system was to have some downtime during the time periods when the tasks are normally run, then the tasks won’t be completed meaning the employees have to manually create any transactions and run any reports that would have normally have been run.

Lastly, instead of developing and implementing my own document management functionality, I would have implemented a document management framework that already exists such as Apache JackRabbit, as these frameworks have a large amount of built in functionality, which have been tried and tested meaning that I would not have to spend large amounts of time developing the document management system and going through extensive unit testing, as I would be confident the functionality would already work and would just need to do some integration testing and system testing to ensure that the framework has been correctly implemented in to the system, and all units are working together as they should.

* 1. **Project Plan and Management**

Although I believe the project was a success, the project plan I defined earlier in the project was not very realistic with regards to time allocation for tasks, and although I believe this was because of the lack of experience I have with developing systems of this size, it resulted in me allocating not enough time for tasks, which then had a knock on effect on other tasks when tasks overran.

Additionally, as explained I spent a large amount of time analysing and designing the entire system functionality, which was a bad choice as I was adopting an agile, iterative process for development, and if I had handled the design stage for each iteration as I was going through that cycle instead of needing to change the system requirements as I move through the process, my designs would have been more accurate as requirements will be defined for that cycle meaning the designs can be more accurate.

However, although my project plan was not realistic and I moved away from the agile process slightly, I believe I managed the project well through constantly assessing the risks to the project (although did not produce the formal risk assessment documents), which would result in me not being able to successfully meet the project aim and objectives and was important in me realising early in the development that there was a major risk of not achieving all goals due to a lack of time, and making decisions with regards to the number of design documents produced and how I was going to implement functionality to allow me to achieve a large portion of the project aim and objectives.

1. **Evaluation**

As I outlined in the conclusion, I believe that overall the project was a success, as I have developed a distributed system that meets the requirements for the system, enabling me to determine if the implementation of software engineering technologies into a business will streamline business processes and improve the efficiency and effectiveness of the business process.

* 1. **Quality of Work**

Firstly, I believe that the analysis of ‘MSc Properties’ current business processes carried out, enabled me to produce a system requirements list, where I could then go on to carry out literature reviews in to the different software engineering technologies that would provide benefits to ‘MSc Properties’ current business processes, and thus improving the business processes efficiency and effectiveness.

Secondly, as explained in the conclusion, the literature review carried out for the project provided me with a number of key Software Engineering views and concepts, and brings a number of different software engineering technologies that can be used to develop the distributed system, however I did not take into account how other systems with similar functionality implemented the functionality, and additionally, didn’t properly consider the advantages and disadvantages of implementing each of the different software engineering technologies presented in conjunction with the system requirements.

Thirdly, although I did not implement an existing document management system framework such as Apache JackRabbit and decided to implement my own, the implementation of the document management system successfully allows the users to store documents, with a comment, and update this document by supplying a new document to the system, tracking updates to the document and version numbers. However, the repository for storing the documents on the server side is very basic as the documents are just stored in one location, but with some improvements I could ensure that documents are stored within a hierarchy of folders (which are created at the creation stage for an object that holds documents), ensuring the documents are stored in a file system.

Additionally the documents stored within the web server directory, have been placed there manually as I was having some issues with adding the document management repository directory in to the http.conf file for the Apache web server running to host my website developed in iteration cycle 8, meaning that the documents stored in the document management repository are not visible to the web server meaning that the website at present is not able to find the documents that are stored in the document management system without copying them in to the documents location.

Lastly, I believe the testing section could have been done considerably better because due to time constraints, it required me to rush the testing I conducted meaning I was not able to document my test scripts, outlining the test description, pre and post conditions, and expected and actual outcomes of each test carried out. Also, I would have logged all the bugs I encountered, and the process I went through to resolve these bugs found in a more structured way, as due to the rushed testing, I was too busy trying to identify what the actual bug was and how to resolve it, without actually documenting the bug in a formal process. Additionally, I would have carried out more testing of the functionality, ensuring I supply a number of different inputs, meaning I would test the system more thoroughly.

* 1. **Changes to Project**

Although I believe the project was a success, there are a few decisions I would have made differently if I was to complete the project again. Firstly, I would aim to set out a more realistic project plan from the offset, which would result in a more manageable project, meaning I would hopefully not run over allocated time for tasks, meaning an amendment to documents I wanted to produce and functionality implemented.

Secondly, I would amend the configuration of the website, as I was having some issues with the SQL statement to return the properties for the quick and full search, which meant that I used a number of if statements and SQL statements to get the required properties to return. Additionally, the search results page display is not good, as I am just returning all properties that meet the search criteria no matter what the number of properties being returned is, this means that the properties search page extends past the formatted area of the webpage, declared in the CSS file, meaning that the display of the properties is not nice, whereas if I was to develop a search results screen that limits the number of properties returned, and allows the users to go through the different search results pages depending on the number of properties returned for the search.

Thirdly, the exception handling for the website is poor as I do not handle incorrect data entry properly for all areas of the website (apart from the service request data entry forms) and pass an informative message back to the user letting the user know exactly what is missing or the problem with the request. Additionally, with each service request completed, a confirmation page is presented but unfortunately so does the email sending return text, which can be confusing to the user and I would not return this if done again.

Additionally, the exception handling for the java application is basic and only returns a simple message that there was an error with the request, which could occur due to incorrect data entry on the client side, or a problem with processing the request on the server side. Furthermore, when the user is not able to perform an action due to incorrect user privileges, the user is not notified that they are not able to perform the function after they have tried to perform the function, whereas if the graphical user interface prevents the user from invoking the method in the first place, either through non active buttons, or non-visible buttons, and the proxy server is just additional security to prevent unauthorised access.

The final changes to the project I would make would be to produce a fuller analysis of the different software engineering technologies I could use to implement the required functionality, and additionally I would carry out research into the different systems that implement similar functionality. In doing so, this will give me a better understanding of the different ways for me to implement the required functionality, and allow me to make the best design and implementation decisions, which will improve the quality of the deliverables, and in doing so will enable me to produce a distributed system that can distributed system and therefore, the businesses efficiency and effectiveness, more than the benefits gained through the current implementation.

Bibliography

References

1. Sommerville, I. (2011). Introduction. In: Horton, M. and Hirsch, M. and Goldstein, M. and Bell, C. and Holcomb, J. *Software Engineering*. 9th ed. Boston: Pearson. P1-26.
2. Citizens Advice. (2015). *The benefit cap - what you need to know.* Available: https://www.citizensadvice.org.uk/benefits/the-benefit-cap/the-benefit-cap-what-you-need-to-know/. Last accessed 20th Jun 2015.
3. Enfield Council. (2013). *Enfield's Homelessness Strategy 2013-2018.* Available: http://www.enfield.gov.uk/download/downloads/id/8004/enfields\_homelessness\_strategy\_2013-2018. Last accessed 20th Jun 2015.
4. Hunt, B. (2015). *UH Ethics Approval.* Available: http://www.studynet2.herts.ac.uk/ptl/common/ethics.nsf/Homepage?ReadForm. Last accessed 19th Jul 2015.
5. Cockburn, A and Highsmith. (2001). Agile Software Development: The People Factor. *Computer*. 34 (1), p131-133.
6. Huo, M, Verner, J, Zhu, L and Babar. (2004). Software quality and agile methods. *Computer Software and Applications Conference*. 1 (1), p520 - 525.
7. H, Gomaa. (2001). ICSE 01 Proceedings of the 23rd International Conference on Software Engineering. *Designing Concurrent, Distributed, and Real-Time Applications with UML*. 1 (1), p737 - 738.
8. Sommerville, I. (2011). Software processes. In: Horton, M. and Hirsch, M. and Goldstein, M. and Bell, C. and Holcomb, J. *Software Engineering*. 9th ed. Boston: Pearson. P27-55.
9. A.D, Birrell and B.J, Nelson. (1984). ACM Transactions on Computer Systems. *Implementing remote procedure calls*. 2 (1), p39 - 59.
10. J.P, Martin-Flatin. (1999). Push vs. pull in Web-based network management. *Integrated Network Management, 1999. Distributed Management for the Networked Millennium. Proceedings of the Sixth IFIP/IEEE International Symposium on*. 1 (1), p3 - 18.
11. D, Habam and K.G, Shin. (2002). Application of real-time monitoring to scheduling tasks with random execution times. *Software Engineering, IEEE Transactions on*. 16 (1), p1374 - 1389.
12. M.L, Horowitz. (1991). An Introduction to Object-Oriented Database and Database Systems. *Object-Oriented Database and Database Systems*. 1 (1), p1 - 86.
13. C, Amza, A, Chanda, S, Elnijety, R, Gil, K, Rajamani, W, Zwaenepoel, E, Cecchet and J, Marguerite. (2002). Specification and implementation of dynamic Web site benchmarks. *Workload Characterization, 2002. WWC-5. 2002 IEEE International Workshop on*. 1 (1), p3 - 13.
14. Y, Cheon and G.T, Leavens. (2002). A Simple and Practical Approach to Unit Testing: The JML and JUnit Way. *ECOOP 2002 — Object-Oriented Programming*. 2374 (1), p231 - 255.
15. L, Briand and Y, Labiche. (2002). A UML-Based approach to System Testing. *Software and Systems Modeling*. 1 (1), p10 - 42.
16. Wikipedia. (2012). *Use Case Diagram.* Available: https://en.wikipedia.org/wiki/Use\_Case\_Diagram. Last accessed 10th Oct 2015.
17. Wikipedia. (2004). *Data flow diagram.* Available: https://en.wikipedia.org/wiki/Data\_flow\_diagram. Last accessed 10th Oct 2015.
18. Wikipedia. (2005). *Class Diagram.* Available: https://en.wikipedia.org/wiki/Class\_diagram. Last accessed 16th Aug 2015.
19. Wikipedia. (2004). *Entity Relationship Model.* Available: https://en.wikipedia.org/wiki/Entity%E2%80%93relationship\_model. Last accessed 23rd Oct 2015.
20. Wikipedia. (2004). *Class-responsibility-collaboration card.* Available: https://en.wikipedia.org/wiki/Class-responsibility-collaboration\_card. Last accessed 11th Aug 2015.
21. Wikipedia. (2005). *Sequence Diagram.* Available: https://en.wikipedia.org/wiki/Sequence\_diagram. Last accessed 10th Oct 2015.
22. Wikipedia. (2002). *Storyboard.* Available: https://en.wikipedia.org/wiki/Storyboard. Last accessed 10th Oct 2015.
23. Wikipedia. (2005). *Observer pattern.* Available: https://en.wikipedia.org/wiki/Observer\_pattern. Last accessed 30th Oct 2015.
24. Wikipedia. (2002). *Singleton pattern.* Available: https://en.wikipedia.org/wiki/Singleton\_pattern. Last accessed 9th Oct 2015.
25. Wikipedia. (2003). *Strategy pattern.* Available: https://en.wikipedia.org/wiki/Strategy\_pattern. Last accessed 10th Aug 2015.
26. Wikipedia. (2004). *Creational pattern.* Available: https://en.wikipedia.org/wiki/Creational\_pattern. Last accessed 10th Aug 2015.
27. Wikipedia. (2003). *Iterator pattern.* Available: https://en.wikipedia.org/wiki/Iterator\_pattern. Last accessed 10th Aug 2015.
28. Wikipedia. (2003). *Composite pattern.* Available: https://en.wikipedia.org/wiki/Composite\_pattern. Last accessed 10th Aug 2015.
29. Wikipedia. (2001). *Inheritance (object-oriented programming.* Available: https://en.wikipedia.org/wiki/Inheritance\_(object-oriented\_programming). Last accessed 10th Aug 2015.
30. Wikipedia. (2005). *Object composition.* Available: https://en.wikipedia.org/wiki/Object\_composition. Last accessed 10th Aug 2015.
31. Wikipedia. (2002). *Object-relational mapping.* Available: https://en.wikipedia.org/wiki/Object-relational\_mapping. Last accessed 31st Aug 2015.
32. The Apache Software Foundation. (2006). *Apache JackRabbit.* Available: https://jackrabbit.apache.org/jcr/index.html. Last accessed 20th Apr 2016.
33. JBossDeveloper. (2010). *Modeshape.* Available: http://modeshape.jboss.org/. Last accessed 11th Aug 2015.
34. Wikipedia. (2002). *Microsoft Access.* Available: https://en.wikipedia.org/wiki/Microsoft\_Access. Last accessed 11th Aug 2015.
35. Wikipedia. (2001). *MySQL.* Available: https://en.wikipedia.org/wiki/MySQL. Last accessed 31st Aug 2015.
36. Wikipedia. (2001). *Apache HTTP Server.* Available: https://en.wikipedia.org/wiki/Apache\_HTTP\_Server. Last accessed 11th Aug 2015.
37. Wikipedia. (2007). *Nginx.* Available: https://en.wikipedia.org/wiki/Nginx. Last accessed 11th Aug 2015.
38. Wikipedia. (2010). *Cherokee (web server).* Available: https://en.wikipedia.org/wiki/Cherokee\_(web\_server). Last accessed 11th Aug 2015.
39. Wikipedia. (2002). *Abstract Window Toolkit.* Available: https://en.wikipedia.org/wiki/Abstract\_Window\_Toolkit. Last accessed 11th Aug 2015.
40. Wikipedia. (2003). *Swing (Java).* Available: https://en.wikipedia.org/wiki/Swing\_(Java). Last accessed 11th Aug 2015.
41. D, Matuszek. (2009). *Enhanced for Loop.* Available: https://www.cis.upenn.edu/~matuszek/General/JavaSyntax/enhanced-for-loops.html. Last accessed 27th Nov 2015.
42. Wikipedia. (2001). *JUnit.* Available: https://en.wikipedia.org/wiki/JUnit. Last accessed 10th May 2015.
43. Wikipedia. (2005). *Java Modeling Language.* Available: https://en.wikipedia.org/wiki/Java\_Modeling\_Language. Last accessed 10th May 2015.

**Further Reading**

1. Coulouris, G. and Dollimore, J. and Kindberg, T. and Blair, G. (2012). Charecterization of Distributed Systems. In:Horton, M. and Hirsch, M. and Goldstein, M. and Bell, C. and Holcomb, J. *Distributed Systems Concepts and Design.* 5th ed. United States of America: Pearson. P17-52.
2. Sommerville, I. (2011). Distributed software engineering. In: Horton, M. and Hirsch, M. and Goldstein, M. and Bell, C. and Holcomb, J. *Software Engineering*. 9th ed. Boston: Pearson. P479-507.
3. Cockburn, A. and Highsmith, J. and Bohem, B. (2001). Agile Software Development: The Business of Innovation. *Computer*. 1 (1), p131-133
4. Sommerville, I. (2011). Requirements engineering. In: Horton, M. and Hirsch, M. and Goldstein, M. and Bell, C. and Holcomb, J. *Software Engineering*. 9th ed. Boston: Pearson. P82-117.
5. Sommerville, I. (2011). System modeling. In: Horton, M. and Hirsch, M. and Goldstein, M. and Bell, C. and Holcomb, J. *Software Engineering*. 9th ed. Boston: Pearson. P118-146.
6. Connolly, T. and Begg, C. (2005). Normalization. In: McGettrick, A. *Database Systems A Practical Approach to Design, Implementation and Management.* 4th ed. United States of America: Pearson. P387-414.
7. Connolly, T. and Begg, C. (2005). Entity-Relationship Modeling. In: McGettrick, A. *Database Systems A Practical Approach to Design, Implementation and Management.* 4th ed. United States of America: Pearson. P387-414.
8. Sommerville, I. (2011). Design and implementation. In: Horton, M. and Hirsch, M. and Goldstein, M. and Bell, C. and Holcomb, J. *Software Engineering*. 9th ed. Boston: Pearson. P176-204.
9. Reges, S. and Stepp, M. (2011). Graphical User Interface. In: Hirsch, M. and Goldstein, M. and Bell, C. and Holcomb, J. *Building Java Programs A Back to Basics Approach.* 2nd ed. Boston: Pearson. P846-909.
10. Coulouris, G. and Dollimore, J. and Kindberg, T. and Blair, G. (2012). Remote Invocation. In: Horton, M. and Hirsch, M. and Goldstein, M. and Bell, C. and Holcomb, J. *Distributed Systems Concepts and Design.* 5th ed. United States of America: Pearson. P201-246.
11. Coulouris, G. and Dollimore, J. and Kindberg, T. and Blair, G. (2012). Distributed Objects and Components. In: Horton, M. and Hirsch, M. and Goldstein, M. and Bell, C. and Holcomb, J. *Distributed Systems Concepts and Design.* 5th ed. United States of America: Pearson. P351-396.
12. Coulouris, G. and Dollimore, J. and Kindberg, T. and Blair, G. (2012). Transactions and Concurrency Control. In: Horton, M. and Hirsch, M. and Goldstein, M. and Bell, C. and Holcomb, J. *Distributed Systems Concepts and Design.* 5th ed. United States of America: Pearson. P691-742.
13. Coulouris, G. and Dollimore, J. and Kindberg, T. and Blair, G. (2012). Distributed Transactions. In: Horton, M. and Hirsch, M. and Goldstein, M. and Bell, C. and Holcomb, J. *Distributed Systems Concepts and Design.* 5th ed. United States of America: Pearson. P743-780.
14. Connolly, T. and Begg, C. (2005). SQL: Data Manipulation. In: McGettrick, A. *Database Systems A Practical Approach to Design, Implementation and Management.* 4th ed. United States of America: Pearson. P112-156.
15. Connolly, T. and Begg, C. (2005). Security. In: McGettrick, A. *Database Systems A Practical Approach to Design, Implementation and Management.* 4th ed. United States of America: Pearson. P541-571.
16. Connolly, T. and Begg, C. (2005). Transaction Management. In: McGettrick, A. *Database Systems A Practical Approach to Design, Implementation and Management.* 4th ed. United States of America: Pearson. P572-629.
17. Coulouris, G. and Dollimore, J. and Kindberg, T. and Blair, G. (2012). Security. In: Horton, M. and Hirsch, M. and Goldstein, M. and Bell, C. and Holcomb, J. *Distributed Systems Concepts and Design.* 5th ed. United States of America: Pearson. P479-536.
18. Sommerville, I. (2011). Software testing. In: Horton, M. and Hirsch, M. and Goldstein, M. and Bell, C. and Holcomb, J. *Software Engineering*. 9th ed. Boston: Pearson. P205-233.
19. Sommerville, I. (2011). Project management. In: Horton, M. and Hirsch, M. and Goldstein, M. and Bell, C. and Holcomb, J. *Software Engineering*. 9th ed. Boston: Pearson. P593-617.
20. Sommerville, I. (2011). Project planning. In: Horton, M. and Hirsch, M. and Goldstein, M. and Bell, C. and Holcomb, J. *Software Engineering*. 9th ed. Boston: Pearson. P618-650.
21. Waldo, J. (1998). Remote procedure calls and Java Remote Method Invocation. *Concurrency, IEEE*. 6 (3), P5-7.
22. Guan, H. and Ip, H. and Zhang, Y. (1998). Java-based approaches for accessing databases on the Internet and a JDBC-ODBC implementation. *Computing & Control Engineering Journal*. 9 (2), P71-78.
23. Wikipedia. (2004). *Data Flow Diagram.* Available: 17. https://en.wikipedia.org/wiki/Data\_flow\_diagram. Last accessed 16th Aug 2015.