The candidate confirms that the work submitted is their own and the appropriate credit has been given where reference has been made to the work of others.

I understand that failure to attribute material which is obtained from another source may be considered as plagiarism.

(Signature of student………………………………….)

Analysis and Design of the SoC
Module Timetable system using UML and the RUP
By James Halstead
Information System Degree 2001/02
Summary

Aim of the project

The overall aim of this project is to use object oriented analysis tools to investigate the way modules within the School of Computing are timetabled and the associated problems. In this project I will be adopting the role of a system analyst/designer and using appropriate parts of the Unified Modelling Language (UML) as a visual modelling technique, and follow the Rational Unified Processes (RUP) as a methodology for designing systems. These tools will produce a series of documents and diagrams that together will that will constitute a report in their own right into this problem area.

Key objectives of the project

Therefore, my achievable steps towards the aim of this project are:

• Model the current process of module allocation system through to the booking of lecture theatres using UML and the RUP.
• Analyse this current situation and make appropriate suggestions for new systems.
• Highlight and discuss the features of a new system, again using the RUP.

Minimum requirements

• Research into the current system for timetabling modules and booking lecture theatres.
• Use business-modelling tools to analyse and model the existing module timetabling system.
• Make appropriate recommendations for enhancement.

List of project deliverables

• UML diagrams and RUP ‘artifacts’ for the modelling of the current system.
• Analysis of the current system.
• Appropriately use of stages in the RUP to support recommendations for the new system up to the initial construction phases.

For a detailed schedule on project deliverables see Appendix B section entitled Iteration Plan
Acknowledgements

The author would like to thank Dr Kevin McEvoy and Owen Johnson for their help in selecting an appropriate area of study for analysis. I would also like to thank some friends for their continued patience in proof reading this report without getting too bored; Francesca Gibson, Phil Stansfield and Jon Cranshaw.
Contents

SUMMARY ........................................................................................................................................... I

ACKNOWLEDGEMENTS ............................................................................................................... II

CONTENTS ................................................................................................................................. III

CHAPTER 1: SELECTION OF PROJECT METHODS........................................................................ 1

OVERVIEW ........................................................................................................................................... 1

INTRODUCTION .............................................................................................................................. 1

SEQUENTIAL APPROACH: SSADM ................................................................................................. 2

ITERATIVE APPROACH: RAD ........................................................................................................... 3

OOAD .................................................................................................................................................. 4

END USER ETHIC .............................................................................................................................. 4

METHODOLOGY CRITIQUE ............................................................................................................... 5

CHAPTER 2: AN INTRODUCTION TO UML AND THE RUP ............................................................. 4

OVERVIEW ........................................................................................................................................... 4

UML ................................................................................................................................................... 4

THE RATIONAL UNIFIED PROCESS ................................................................................................. 8

CHAPTER 3: METHODOLOGY ....................................................................................................... 10

OVERVIEW ........................................................................................................................................... 10

DEVELOPING A SMALL PROJECT .................................................................................................... 11

INCEPTION PHASE ........................................................................................................................... 12

ELABORATION PHASE ...................................................................................................................... 14

CHAPTER 4: RESULTS .................................................................................................................. 17

OVERVIEW ........................................................................................................................................... 17

INCEPTION PHASE: BUSINESS MODELLING WORKFLOW ......................................................... 17

The Problem Area ........................................................................................................................... 17

Problem Ownership .......................................................................................................................... 18

Fig.10 Business Sequence Diagram .............................................................................................. 20

Fig.11 Business Use Case Diagram ................................................................................................ 21

Business Actors ............................................................................................................................... 21
<table>
<thead>
<tr>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Business Use Cases</strong>............................................................................22</td>
</tr>
<tr>
<td><strong>Fig. 12 Initial System Context Diagram</strong> ........................................23</td>
</tr>
<tr>
<td><strong>INCEPTION PHASE: REQUIREMENT WORKFLOW RESULTS</strong> ................................24</td>
</tr>
<tr>
<td><strong>Re-cap Problem Statement</strong>....................................................................24</td>
</tr>
<tr>
<td><strong>System Position Statement</strong>....................................................................24</td>
</tr>
<tr>
<td><strong>Market Demographics</strong> ...........................................................................25</td>
</tr>
<tr>
<td><strong>ELABORATION PHASE: ANALYSIS AND DESIGN RESULTS</strong> .............................28</td>
</tr>
<tr>
<td><strong>System Overview</strong> ..................................................................................28</td>
</tr>
<tr>
<td><strong>System Functions</strong> ................................................................................28</td>
</tr>
<tr>
<td><strong>System Advantages</strong> ...............................................................................29</td>
</tr>
<tr>
<td><strong>System Features</strong> ..................................................................................30</td>
</tr>
<tr>
<td><strong>UML Results</strong> .......................................................................................32</td>
</tr>
<tr>
<td><strong>Fig. 13 System Deployment Diagram</strong> ..................................................33</td>
</tr>
<tr>
<td><strong>Fig. 14 System Class Diagram</strong> .............................................................34</td>
</tr>
<tr>
<td><strong>Fig. 15 System Activity diagram</strong> .........................................................35</td>
</tr>
<tr>
<td><strong>Fig. 16 System Sequence Diagram</strong> .......................................................36</td>
</tr>
<tr>
<td><strong>Fig. 17 System Analysis Model</strong> .............................................................37</td>
</tr>
<tr>
<td><strong>Supplementary System Specifications</strong> ..................................................38</td>
</tr>
<tr>
<td><strong>CHAPTER 5: PROJECT EVALUATION</strong> .......................................................41</td>
</tr>
<tr>
<td><strong>OVERVIEW OF EVALUATION CRITERIA</strong> ..................................................41</td>
</tr>
<tr>
<td><strong>Project Objectives</strong> ...............................................................................42</td>
</tr>
<tr>
<td><strong>Evaluation of RUP and UML</strong> ...............................................................43</td>
</tr>
<tr>
<td><strong>Project Management</strong> ............................................................................46</td>
</tr>
<tr>
<td><strong>Improvements to the project</strong> ................................................................47</td>
</tr>
<tr>
<td><strong>Further Study</strong> ......................................................................................48</td>
</tr>
<tr>
<td><strong>THE LAST WORD</strong> ..................................................................................49</td>
</tr>
<tr>
<td><strong>CONCLUSION</strong> .......................................................................................50</td>
</tr>
<tr>
<td><strong>APPENDIX A</strong> .......................................................................................51</td>
</tr>
<tr>
<td><strong>Guidelines for future students attempting an RUP/UML project</strong> .............51</td>
</tr>
<tr>
<td><strong>BIBLIOGRAPHY</strong> ....................................................................................55</td>
</tr>
</tbody>
</table>
Chapter 1: Selection of Project Methods

Overview

As this project is primarily concerned with the RUP methodology this chapter will give a brief overview of some other systems development methodologies. It will look primarily at the fundamental differences between the sequential and iterative approaches before giving some background on OOAD. I will then go further to discuss the limitations of these technologies in a critique section to assess their appropriateness for this project.

Introduction

"As technology becomes more advanced and as a race we become more reliant on technology for every day tasks, the prospect of badly designed systems looms like the Angel of Death in front of us all."

Madison (2000)

Madison is an advocate of designing systems with considered thought and correctly points out in his work the importance of designing systems correctly when human life, and when great financial costs are at stake. Turoff is one of many methodologists that agrees with Madison and sees the need for not just acceptable, but complete methodologies free from error:

"we must construct systems in a systematic, effective, and reliable manner"

Turoff (2001)

In the beginning (the early days of system development), there was a complete absence of an Industry standard methodology for programmers to use. They used what Turoff (2001) terms as “ad hoc analysis and design techniques” with little user involvement to produce systems that inevitably didn’t meet the requirements of the project. A time when Madison and Turoffs concerns would have been of paramount concern for designing systems correctly. However system analysts and designers have the ability to learn form their mistakes and the problems they encountered led to the information systems community to develop structured methods that as Hutchings (2001) identifies would:

“(a) formalise the requirements elicitation process to reduce the chances of miss-understanding the requirements and (b) to introduce best practice techniques to the analysis and design process”
Sequential Approach: SSADM

SSADM stands for Structured Systems Analysis and Design Methodology and is one methodology that uses a sequential approach to designing systems. It is termed by many such as Avison and Fitzgerald (1995) as “one of the traditional approach to software engineering”. The following diagram shows the underlying principle of the sequential methodology:

SSADM has been used widely since its development in 1981 and is used predominantly still by many branches of the British Government as their methodology of choice. This has resulted in many businesses following a copycat approach and also adapting it, leading to many upgrades and endorsement. Goddard and Slater (1995) agree, “The experienced gained in the first few years as a government standard has been fed back into the method to ensure its usability and practicality”

The ideas of structured methodologies such as the Structure Systems Analysis and Design Methodology (SSADM) were born well before its development, as far back as the 1970’s. This was when more formal methods were needed to provide a complete framework for capturing and analysing requirements and specifying a system design. In the beginning, designing systems was based on the basic idea of functional decomposition. The idea that each system can be broken down into progressively smaller processes and functions. This developed over the next ten years to
incorporate data flow and entity relationship diagrams to help planning, and was finally formalised and developed in the UK in 1981 by the CCTA (Central Computing and Telecommunications Agency). The main reason being to standardise the many and varied IT projects being developed across government departments. It was believed that it would help improve productivity and communication and make planning easier. To a certain extent it has succeeded as planned as a lot of organisations and businesses still employ structured methods such as SSADM very successfully. However, as technologies have developed so have the ideas they incorporate and left many businesses that use this methodology behind.

**Iterative Approach: RAD**

This stands for Rapid Application Development and was developed in 1991 as SSADM version 4 was completed and nearing its peak in the industry. The traditional methods such as SSADM are based around completing each stage and then moving onto the next one. For example users often are required to sign off a complete requirements specification around which the software design will take place. However, this doesn’t account for the fundamental premise that requirements will change, especially if the project is large and continues for some time. The main benefit of RAD being that it uses an iterative approach to developing systems that allows for self-correction and backtracking to the requirements phase of development. The diagram below shows the difference between the traditional sequential methodology and the iterative.

![Fig. 2 Traditional vs. RAD methodology McConnell (2002)](image_url)
The fundamental concept of this methodology is that with each iteration (one iteration being a complete circle in the figure above) a working model of the final system is completed. With the first iteration possibly delivering 20 percent of what is required by the users, the second iteration 50 percent and so on until the users or stakeholders are satisfied. The main benefits of this is that it allows for inevitable human error as risk can be recognised early and misunderstandings can be retrieved with as many iterations as time permits. Hutchings sums up this fundamental concept well:

“SSADM adopts the adage that a picture is worth a thousand words, RAD goes a step further and advocates that a working model is worth a thousand pictures “

Hutchings (2001)

**OOAD**

At the moment the most popular way to designing a modern system is through the idea of object oriented analysis and design (OOAD) methodologies, such as ROAD (Rapid Object Application Development) and XP (Extreme Programming). The underlying concept is that everything in a system can be made up of objects (physical or conceptual) that form together in a hierarchical fashion under classes. All the components are inter-related in some degree and can be re-used with other systems. A simple example would be a group of Windows applications could be a class and each individual application an object of that class. In more detail the programming of that application could be a function, or method of that object. In practical terms this means that a system designed with this OO method can be classed as a re-usable system, one that is durable, upgradeable and easy to maintain. MOSES is one sort of a object-oriented (OO) software development methodology which encompasses not only technical aspects of OO Analysis and Design (OOA/D) but also project management, business planning, maintenance and product enhancements. Developed in 1994 and has a lot in common with subsequent methodologies such as RUP.

**End User Ethic**

End user computing is not so much a methodology, more of an alternative ethic. A lot of system designers either get bogged down in the hundreds of different methodologies or just think that they know better and design systems themselves. It has come about as projects expanded IT departments took longer and longer, so normal people started to design systems themselves. This was achieved by the ordinary users themselves using PC tools such as Lotus 123, Clipper and similar packages. They
have had limited success but it is largely dependant on the size of the project being small. Mostly larger projects of this nature do fail as those around lack the technical expertise to finish the project and as the employ weak methods they don’t integrate easily into other systems.

Methodology Critique

The problems associated with normal structured methods such as SSADM are evident throughout the computer world. The analysis teams that try to implement these sorts of systems need to know every requirement about the system in order for it to function. They decompose each element again and again and are based on the assumption that the requirements of the system will not change. Unlike RAD that incorporates the most important fundamental concept; that requirements will change during the lifecycle of a system. As Lau Kam-hug (2002), Director of Information Services for the Hong Kong Special Region of the People’s Republic of China points out earlier this year “RAD refers to a development lifecycle to deliver systems that are directed towards change”.

With an OOAD approach the system is flexible and easily open to change. With a structured approach if the system needs to change then it can require a massive system wide redevelopment. This can be expensive not just in terms of the monetary value, but for the length of time it takes for the system to be up and functional again, which to some small businesses could be fatal. Not only do structured methods take a lot longer than OOAD methodologies, they tend to be more expensive as more and more people need to be employed as tasks gets out of hand and systems fail. A good example of this is the TAURUS (Transfer and Automated Registration of Uncertified Stock) project for the London stock exchange. It was commissioned in 1987 and after six years of development, trying to link 280 financial institutions, the project was cancelled. It cost the London stock exchange $130 million and a further $600 million from securities companies on the assumption that it would work. The reason for this catastrophic failure was that they employed a very weak structured methodology. As the project grew, they couldn’t keep up with the changing requirements and lacked the methodology to manage a project of this size. So much so that they never even produced a full system design.

Another major case example of problems with structured methodology we have seen this century would be the disaster associated with the “millennium bug”. This is a massive example of how structured methods can fail on a phenomenally large scale. Most systems that encountered problems
were designed using structured methods that led to the software either taking a long time to upgrade, or being more trouble than it is worth. The problem may still have occurred using an OOAD however; the cost to repair the system would have been minimal in comparison to the billions of pounds spent by companies ensuring that their existing and somewhat old systems were going to work come the 1st of January 2000. Modern OOAD systems allow for change and will continue to grow, unlike the old methods such as SSADM where change is more difficult and the methodology is becoming dated. More so now leaving the way clear for new methods to fill its shoes as precision-design.co.uk correctly points out “CCTA have ceased to govern the development of SSADM which leaves the method without a champion”.

For the end user approach to systems development the tools available in the computer market today mean that anyone can have a go at designing their own system. Today it is a ‘free for all’ and anyone can have a go and is regularly done by individuals who are not from an IT related background (attempted largely by middle managers and sometime finance director themselves). This in itself can lead to problems if they do not have the technical comprehension to complete the task. If they have never done anything like this before their system is unlikely to be a compatible, reliable and would be difficult to install into an office network. Generally speaking the coding is bad as they don’t have an appreciation of the critical issues relating to analysis and design, but the user interface (UI) and usability of the systems is good. However, what so often happens though is that they start out with a clear idea of what is needed, but don’t have the methodology to see it through. Time progresses and more time is spent perfecting it, making it better and ‘tweaking’ it for ultimate performance. A good example of this approach is from my own IT experiences working in a registry department in a college environment. They decided that it was time to upgrade their current system and the registry team who had some IT experience were required to design and run a campus wide system to streamline the department. The project was given the go ahead in February and the deadline for the start of new courses was September of the same year so there was a specific time constraint. The individuals that designed the system worked alone and tried to explain what they were doing to senior managers as they went along. In the end the system became overly complicated and communication broke down completely in this group and eventually the registry staff were forced to become involved to a greater extent. In the end the system just wasn’t ready in time as all the individuals lacked the skills and experience to finish the task. The solution was at a huge financial cost to the college and LEA; that a separate team of professionals was brought in to do the job and the previous three months work on the system had to be scrapped completely.
The best methodologies around today incorporate the best of both the sequential and iterative approaches. With structured methods, the most well known being the SSADM approach is purely sequential and also does not cover issues such as construction, testing and implementation of the developed software unlike other methodologies such as the RUP. Which is one such methodology that utilises the benefits of an iterative approach, while also appreciating the sequential. The RUP (2002) correctly states these benefits of the iterative process being: “The iterative approach lets you take into account changing requirements as they will normally change along the way”.

A general comment about using any methodology is that they are restrictive processes that take away some programmers ethic of just sitting down and hacking away at code. More so with small systems as some designers and programmers view non-code artifact production and documentation a waste of time and energy. Why would you want to document the user requirements and conduct risk analysis when you could just sit down and write the system yourself in a couple of days? An approach that is often used on many small software projects unless specific documentation is required, as all requirements could be known and unlikely to change.

Primarily it may be cheaper using an end user approach, with no specific or made-up methodology, but taking into consideration the extra cost of re-designing a faulty system it would be far more beneficial to use an object oriented standard approach. The major benefit over the traditional structured methods being that objects start from real world ideas and structured methods start from processes. Making the object oriented approach much more practical as each object encapsulate the data and process together. With regards to actual software programming, this means that code can easily be re-used and integrated into many different systems. Making months of drawn out maintenance and upgrades a thing of the past. Not only is it a faster, cheaper solution in the short-term, in the long-term results in a flexible and modern system, as Evans a UML expert agrees:

“Object oriented technologies (object modelling, patterns, reflection, component based development, frameworks, product lines and so on) have a key role to play in system development”

Evans (2000)
Chapter 2: An Introduction to UML and the RUP

Overview

After giving some general background into systems development methodologies in the previous chapter, this chapter gives a history and explanation of the method and tool I have used to conduct this project. These being: the RUP as a process methodology, and UML as a somewhat Industry standard modelling language for system designers and analysts.

UML

Universal Modelling Language, or ‘UML’ as it is more popularly known “uses object-oriented techniques to facilitate reuse” Johnson (2000). It is a collection of visual models that utilise diagrammatic form to show the interaction between different the parts of any given system (very similar to the way in which architects use blueprints to design buildings).

"UML helps you specify, visualize, and document models of software systems, including their structure and design, in a way that meets all of these requirements.”

(OMG 2002)

UML was developed by "The Three Amigos"; leading Industry Methodologists of Information System Development. The three amigos being:

- Grady Booch, who has been with the Rational Corporation since its founding in 1980.
- Dr James Rumbaugh, a chief developer of Object Modelling Technique (OMT), who joined Rational Corporation in 1994

When developing UML the three amigos got considerable support from many different businesses that were keen to know more about UML as it amalgamated many engineering techniques and
commonly accepted practices. Because of this need for an industry standard, in November 1997 the Object Management Group (OMG) officially made UML an approved standard.

The following diagram shows what the OMG are planning the future development of UML. It shows that the current version of UML is version 1.4 and the goal of the OMG is to produce version 2.0 within the next few years. According to OMG UML version 2.0 will:

“Restructure the architecture of the language so that it is easier to understand, implement and extend while preserving the end semantics of the language.”

OMG (2002)

Fig. 3 OMG-UML evolution. Muchogu (2001)
However, UML 2.0 is still in the developmental stage with the Infrastructure and Superstructure documentation still yet to be finalised.

The output of UML 1.4 is a set of diagrams; of which there are twelve standard diagram types. They are split into three categories as the OMG defines as:

**Structural (Static) Diagrams** include the Class Diagram, Object Diagram, Component Diagram, and Deployment Diagram.

**Behaviour (Dynamic) Diagrams** include the Use Case Diagram (used by some methodologies during requirements gathering); Sequence Diagram, Activity Diagram, Collaboration Diagram, and Statechart Diagram.

**Model Management Diagrams** include Packages, Subsystems, and Models.

*Fig. 4 UML Structure (OMG 2002)*

In a project there can be more than one of each of these diagrams, best modelling techniques teach us that design and planning should be done iteratively. Therefore, there should be certainly more than one version of the diagrams. Likewise there can be more then one type of diagram that constitutes the complete workings of a system, which when combined would then define a complete UML model.

Here are some more details as to the content and purpose of each of the diagrams:

**Use case Diagrams**: These show how actors (a person who is involved with the system i.e. a lecturer) relate to use cases (anything that happens within the system i.e. booking a lecture theatre). Usually constructed from use case description forms (short narrative pieces that highlight system interactions, produced predominantly by system analysts).

**Class and Object Diagrams**: Show the general structure of the system and addresses the fundamental concept of object oriented analysis: The utilisation of classes, and grouping like objects together into these classes.

**Component Diagrams**: Show the different components of the system. This is usually software based and can show how applications such as e-mail and the Internet function together.
Deployment Diagrams: Essentially show the physical layout and structure of the system i.e. the computers and servers. It can be localised to small networks and offices, but also can be applied to large information systems.

Activity Diagrams: Shows what actually happens in the system, the actual flow of events and possible outcomes of these events i.e. sequence of action that lead booking a room or lecture theatre.

Statechart Diagrams: Shows the different states that an object can be in, for example, on/off state. An object being a representation of a “Real thing or concept” Johnson (2000).

Collaboration Diagrams: Shows the message passing between objects and shows in more detail how objects interact with each other. In the RUP this diagram, along with sequence and activity diagrams, that show workflows that Booch (2000) describes as “a sequence of activities that produces a result of observable value”.

Sequence Diagrams: These diagrams show the sequence of messages sent between collaborating objects for any particular task.

UML is a powerful visual modelling tool, however it is not designed to be used on its own and should be used in conjunction with a methodology. A methodology being something that “formally defines the processes that are used to analyse the environment, capture requirements, and design a system based on the requirements” Hutchings (2000). UML compliments many methodologies, none more so than the Rational Corporations Rational Unified Process. The three amigos that developed UML are key figures in this corporation and largely behind the design of this process, therefore it relies heavily on UML.
The Rational Unified Process

The Rational Unified Process (RUP) is one methodology for building object-oriented systems that has been developed over the last few years by the Rational Software Corporation. The RUP has integrated some of the industries best practices and tools and formalised them together to form one package. As the following diagram shows:

![RUP Best Practices Diagram](image)

One best practice that it uses is by combining the benefits of an iterative approach to system development whilst having an appreciation for a sequential approach. It also utilises powerful component base architecture that results in a system that can be adapted and maintained easily.

The RUP “suites all types of projects, ranging from small system development, to large multi platform systems” RUP (2002). It accomplishes this because the RUP is a process framework. It provides you with many options on what to produce and the choice to select/neglect appropriate artifacts, rather than telling you exactly what should be produced. These come in the form of RUP ‘roadmaps’ that give you guidelines and information, as well as practical examples on producing documentation. Rational use this as one of its key marketing points:

“As an industry-wide process platform, you can easily choose the set of process components that are right for your specific project needs”

Rational Software Corporation (2002)
The RUP uses nine core workflows to define the stages in a project lifecycle. A workflow being “the series of activities or events that the worker uses to produce an observed value” Jacobson (1999).

These workflows are: Business Modelling, Requirements, Analysis and Design, Implementation, Test, Deployment, Configuration and Change management, Project Management and Environment. Each core workflow includes activities; a job that the worker (anybody that has responsibility in the development of a system) can actually perform. From these activities comes the production of ‘artifacts’ “something that the worker has responsibility for, that they control, manipulate, produce or use” Johnson (2001). The RUP predominantly uses templates to document these artifacts and anything that is produced by them.

These core workflows are conducted to varying degree over four distinct project phases (the lifecycle of a project over time): Inception, Elaboration, Construction, and Transition phases. Of which the RUP gives guidelines on effective software and systems development through the “utilisation of controlled iterative development” RUP (2002). The iterations are shown along the boom of figure 6 below

The following diagram shows a diagrammatic overview of the RUP:

![Fig.6 RUP Overview (RUP 2002)]
Chapter 3: Methodology

Overview

As previously mentioned in ‘Overall aim of project’ on page 1, this project will “will produce a series of documents and diagrams that together will that will constitute a report in their own right into this problem area” (page 1 line 5). Therefore, the project has been clearly broken down into two RUP phases, which have been iteratively developed according to RUP guidelines. These phases are the RUP Inception and Elaboration phases and are the first two phases that are concerned with systems analysis and design. After which the next phases move onto actual system construction and deployment. These end phases have not been used because this project is not concerned with the actual building of a functioning system. This project focuses on the systems analysis of the current methodologies and design of a new system to improve on current practises for the timetabling problems:

“Iterations in the Inception and Elaboration phases focus on business modelling, requirements, and design activities”

Stansfield (2001)

The major core workflows of these phases being: Business Modelling, Requirements, and Analysis and Design (see figure 6, page 9). The core workflow Implementation also forms a sizable part of the two phases used, however because this project will not be actually building any software, it is not necessary to use this phase. For the same reason, Test, Deployment, Configuration and Change Management core workflows have not been used. The Management and Environment core workflows have been used, however they have no formal section in this chapter. They are included in the artifacts given in Appendix B as they give important details on project management and tools used.

This section shows the methodology behind the project and the processes that have been followed to produce RUP artifacts. The following is not meant to be a detailed project plan, more of an overview of the methodology I have employed to satisfy and move beyond the projects minimum requirements.

For a detailed project plan, with specific dates and deliverables please see the Appendix B artifact entitled: Iteration Plan.
The two RUP phases detailed in the subsequent pages are each a major milestone in this project. They detail the major artifacts produced and the logic behind the methodology. The detail is purposefully described at a relatively high level (low level of abstraction) because one of the most important parts of this project is concerned with the production of appropriate RUP artifacts and the selection of relevant RUP activities:

“In all cases, it is important not to include activities and artifacts that cannot be clearly justified”

RUP (2002)

**Developing a Small Project**

A crucial part of any project that employs a good methodology is to decide what needs to be included or omitted, and at what stage in the process. It is especially critical in this project as the main tool used was the RUP. This methodology contains approximately 50 artifact templates and approximately the same number of guidelines on how to produce other documents and reports. Therefore, not every detail of the RUP has been used, especially due to the nature of this project being concerned with a relatively small system. The process framework followed was the RUP ‘roadmap for developing small projects’ that gives directions as to what artifacts should be produced and highlights the essential activities of each core workflow that should be undertaken. Even this roadmap can be broken down further to core activities that suit the project, so therefore not every recommended activity in the roadmap has been performed. The following activity diagram shows the core workflows, each of which are also milestones of the project that have been produced under each of the two phases.

![Fig. 7 My Basic RUP Project Lifecycle](image-url)
**Inception Phase**

The roadmap for developing small projects breaks each of the two phases into the essential activities and essential artifacts. The essential activities the small project roadmap defines in the Inception phase are as follows:

1. **Formulating the scope of the project.** This involves capturing the context and the most important requirements and constraints to such an extent that you can derive acceptance criteria for the end product.
2. **Planning and preparing a business case.** Evaluating alternatives for risk management, staffing, project plan, and cost/schedule-profitability trade-offs.
3. **Synthesizing a candidate architecture,** evaluating trade-offs in design, and in make/buy/reuse, so that cost, schedule and resources can be estimated. The aim here is to demonstrate feasibility through some kind of proof of concept. This may take the form of a model that simulates what is required, or an initial prototype that explores what are considered to be the areas of high risk. The prototyping effort during inception should be limited to gaining confidence that a solution is possible - the solution is realized during elaboration and construction.
4. **Preparing the environment for the project,** assessing the project and the organization, selecting tools, deciding which parts of the process to improve.

*Fig.8 Essential Inception Activities RUP (2002)*

The main purpose of this Inception phase is to establish what the new system should do, define its boundaries and gain a better understanding of the system requirements. Whilst proving and ensuring that the project is worth doing and achievable. The following details what artifacts have been produced under each of the essential activities shown in figure 8 above (the artifacts are shown in *italics* which can be found in Appendix B).

To ‘formulate the scope of the project’ a *System Vision* document was produced. This encapsulates the projects core requirements, key features and gives a clear understanding of the goals and objectives. According to the RUP (2002), any design should “make sure that all parties involved agree on what is the problem that we are trying to solve with our system”.

---

12
Therefore, at this stage I found it necessary to gain agreement on the problem being solved and identify the main stakeholder and user needs. The RUP recommendation for the best way to gather the information needed for this document is through a series of workshops with stakeholders of the business. This was accomplished by a Stakeholder Requests document, based on interviews with the stakeholder of the system Dr Kevin McEvoy. The Stakeholder Request document gives more than enough detail on the requirements, so much so that it is used as the primary input into the System Vision Document. Also, as part of this activity it was prudent to formulate a Glossary to decide on common terminology. However, because this project is small, I amalgamated the Business Glossary with the Project Glossary, as there are only a few terms associated with each section, (it makes referencing a lot easier).

The next detail is ‘planning and preparing a business case’; the purpose of which is to give a broad overview of the project and to determine whether or not this project is worth investing and going ahead with. This resulted in the production of a Business Case document. However, as this document should include financial information like return on investment (ROI), the financial forecast section had to be left relatively empty. As the only ROI that I could identify for this project was the return of a good degree classification as the result of time and effort spent on this project.

Following on from this is the next essential activity the RUP roadmap details is ‘Synthesizing a candidate architecture’ “to demonstrate feasibility through some kind of proof” RUP (2002). For this project to demonstrate feasibility, I had to prove that this project could actually achieve its minimum requirements, so it was necessary to produce the first version of the Project Iteration Plan (a management workflow artifact) to formalise a schedule and plan for the project according to the time scale available. This contains scheduled deliverables, major milestones and artifact details. However, this doesn’t really prove that the project will work as it is just a plan, so an initial prototype of what the system should be able to do, based on the business modelling was produced. This was a first version of a system context diagram to show what a solution would look like and states what is required of a new system.

The last recommended activity in the Inception phase is ‘Preparing the environment for the project’. To decide on which parts to improve it was necessary to identify the boundaries of the problem. The best way I found to achieve this was through the encapsulation of UML and artifacts to reflect the correct level of abstraction and identify the business functions that the project is focusing on in more detail. The artifacts produced were the Business Sequence and Use Case Diagrams. This led to the
production of the *Business Use Case Model Survey* document, to identify roles and describe how the business workers and entities from the diagrams interact. To give more details of use cases and their interaction with the system; *Business Use Case Specifications* and *Business Sequence Diagrams* were produced, thus describing the actual flow of events at a low level of abstraction.

**Elaboration Phase**

The next move on from this activity is to go to the RUP Elaboration phase (the second milestone in my project) which main objective is “to detail the architecture of the system and provide a base for the creation and implementation of software in the following phases” RUP (2002). The small project roadmap details the following essential activities:

1. **Defining, validating and base lining the architecture** as rapidly as practical.

2. **Refining the Vision**, based on new information obtained during the phase, establishing a solid understanding of the most critical use cases that drive the architectural and planning decisions.

3. **Creating and base lining detailed iteration plans for the construction phase**.

4. **Refining the development case and putting in place the development environment**, including the process, tools and automation support required to support the construction team.

5. **Refining the architecture and selecting components**. Potential components are evaluated and the make/buy/reuse decisions sufficiently understood to determine the construction phase cost and schedule with confidence. The selected architectural components are integrated and assessed against the primary scenarios. Lessons learned from these activities may well result in a redesign of the architecture, taking into consideration alternative designs or reconsideration of the requirements.

*Fig.9 Elaboration Essential Activities (RUP 2002)*

The first essential activity that is defined above is ‘Defining, validating and base lining the architecture as rapidly as possible’. This was achieved through some detailed UML diagrams to define the structure and functionality of the system. This started with detailing a use case diagram at a very high level of abstraction, to describe what the system had to do. From this it was then possible to
construct a UML system sequence diagram to show how system objects would interact. Also followed by a UML system activity diagram to show the flow of events and a UML system class diagram to show the system architecture in a more detailed form.

The next essential activity is to ‘refine the vision’; to get a solid understanding of the use cases. This was achieved through a series of Use Case Specification forms to highlight the main functionality of the systems. They are very detailed and give a complete overview of the system and are key to the refining of the UML diagrams previously mentioned. A large section of this project focuses around the notation of ‘Use Cases’ to represent system requirements and functionality. It is the view of this analyst that they are the building block of the RUP and should be prominent in any good methodology as Constantine and Lockwood (2000) agree “For requirements engineering, use cases provide a concise medium for modelling user requirements; in the hands of user interface designers, use cases can become a powerful task model for understanding users needs”.

The third and fourth activities in this phase are ‘Creating and base lining detailed iteration plans for the construction phase’ and ‘Refining the development case and putting in place the development environment’. Because this project is not going as far as the construction phase it was not needed to produce a detailed plan or refine the project Development Case. However, to highlight one possible implementation, a UML deployment diagram was produced.

The final essential activity that was performed as part of this phase was ‘Refining the architecture and selecting components’. The diagrams reflect this and accompanying information in the form of a Supplementary Specification documents. The Supplementary Specification documents were produced to capture the main non-functional requirements of the system that weren’t captured in the use cases. Also refining the architecture of the system led to the construction of a UML Analysis Model, to show the entities of the system.

One of the main problems in any methodology project is how to manage the stakeholder requests and requirements as they change. In the RUP this is through to the Construction and Transition phases as the project lifecycle progresses. However, as this project is not going this far it is not required that the possible change in requirements are managed. If it were, a Requirements Management Plan and the ‘manage changing requirements’ detail would achieve this by organising and tracking changes to the system. Also one recommendation of the RUP roadmap for small projects is to include a Software Development Plan that identifies roles and responsibilities for software development, planned
production time, staffing plan and implementation specifications. I felt it is unnecessary to produce this artifact, as the project will not actually go into the realms of software development. It can be used as a management document that gathers all information on the project. However, the *Iteration Plan* covers this more than adequately. This ‘roadmap’ is also recommended that a *Business Rules* document is produced, however this was not needed as for this system there are no actual business rules, only guidelines on how things should be achieved. An example business rule would be; a customer can only have an initial overdraft limit of £1500.
Chapter 4: Results

Overview

This chapter describes the output of this project from using the RUP methodology outlined in the previous chapter. As the output of this project is the production of RUP and UML artifacts I have included all that has been produced in Appendix B. Much like with a software project where a copy of the system that was produced accompanies the write up. However, the following section summarises the important points contained within these artifacts.

Inception Phase: Business Modelling Workflow

The Problem Area

There has been concern within the Computing department as to the amount of problems associated with the timetabling of modules. The timetabling goal of each department is to allocate modules a timeslot where students taking a module won’t have any clashes with other modules that they are required to take (compulsory modules). For example, if a first Year School of Computing (SoC) module like COMP1520 ‘Professional Development 1’ were to be timetabled, the SoC module administrative staff would have to make sure that this wouldn’t clash with any other SoC compulsory modules that those students would also be taking. This is quite simple to do, all that is required is that the administrative staff to look at the SoC timetable and allocate a time appropriately. They can do this relatively quickly and easily as they know which other compulsory modules first years must take. So continuing with this example, if they want to allocate COMP1520 to 9 o’clock on a Monday morning, they look at the timetable and see that COMP 1500 ‘Mathematics for Computing’ is timetabled for that timeslot already, a compulsory module for first years, so therefore must chose a different timeslot.

This practice isn’t very problematic and just requires the cross referencing of information local to the SoC. However, the problems arise when considering the needs of Joint Honours in Science students. Due to the popularity of a Joint Honours degree, with students wanting to specialise less and less the Joint Honours in Science department alone offers over 70 programmes of study. This means that the number of modules that these students take is vast, across a wide variety of departments. The main
problem area lies with these students as timetabling control is split between associated departments i.e. Mathematics and SoC departments.

Each year new modules are created and have to be timetabled; modules are amalgamated and move timeslot resulting in a differing timetable from year to year. The main area of concern is the lack of communication between these departments and the associated lack of effective information systems in place. At the moment if a COMP module has to be timetabled, it is done locally, the only check that is done is to consult the SoC timetable to see if it clashes with other SoC compulsory modules. So what about the timetabling of Joint Honours in Science modules? For example, if COMP1650 were to be re-timetabled to 10 o’clock on a Wednesday, the only check would be within the SoC. It may be that Joint Honours in Mathematics and Computing students, who are required to take this module, may have a compulsory Mathematics module timetabled for the same time, thus resulting in a major timetable clash. Currently there is no efficient system to check that when timetabling module Joint Honours students won’t have clashes.

If either the student office did the timetabling of modules centrally or the Joint Honours in Science Department itself, there wouldn’t be this problem as they would have access to all the information. It is only because the timetabling is done locally in each department that these problems arise each year.

This problem area extenuates to how modules are allocated lecture theatres, as there is also concern as to theatres being inappropriately allocated given the needs of the module. Here is a brief summary of the problems that have been experienced with the allocations of lecture theatre:

- The lecture theatres being too small for the number of students, and sizable lecture theatres being allocated to small classes.
- Access problems to the lecture theatres for students with disabilities.
- Location of lecture theatres, away from many departments and in hard to find locations.
- Lecture theatres not having the correct equipment, i.e. computer and www access.
- Failure to notify lecturers of maintenance and problems when they do occur.

**Problem Ownership**

The University Student Office is responsible for the allocation of lecture theatres and they do have a system in place for allocating them depending on the number of students enrolled on a particular
module. However, this is only done before term starts, what if after term starts there is a major timetable clash or for another reason the module has to be moved? Using the example above, if a compulsory MATH module was found to have a clash with a compulsory COMP module for Joint Honours students, one module would have to change timeslot. Who would then take responsibility for solving this problem? It would have to be the Mathematics department, the SoC, the Joint Honours in Science department, or the Student Office. At the moment there is no current practice for checking whether moving a module to a new time would have clashes for those students enrolled and there is no way of knowing if a lecture theatre will be free at those times. The current practice within the SoC is for Dr Kevin McEvoy to find out the times of when an appropriate lecture theatre is free, then walk into the lecture theatre of the module that is being moved and ask the students if they are free at any of these times. This is cumbersome and time consuming and not an efficient method of moving a module to a new timeslot.

Figure 10 on the next page emphasises this problem and my research into this area. It shows the current practice for the timetabling of modules and those involved. From a modules creation through to how they are booked to lecture theatres by the student office (as a sequence of events at a high level of abstraction). The important feature of the UML business sequence diagram is that there is no checking for joint honours module clashes, only checking for COMP module clashes. This diagram like the subsequent UML diagrams has been annotated with text boxes to give more detail and describe the diagrams in a clearer form.

Diagrams without annotation can be found in the report in Appendix B
Fig. 10 Business Sequence Diagram

1. Create COMP module
2. Consult timetable
3. Check for COMP clashes
4. Allocate Module a timeslot
5. Send timetable for processing
6. Allocate and book lecture theatres
7. Take COMP module

No check for joint honour module clashes
Figure 11 below is a UML business use case diagram that gives more of an overview current system area, with a subsequent description of the business actors and use cases used to create this diagram, to give some more detail and focus in on the problem area:

**Fig.11 Business Use Case Diagram**

**Business Actors**

**SoC Learning and Teaching Committee:** This actor is a group of academic staff from the SoC whose role in this system is to decide on whether a new module should be created. They keep up with what Computer graduates should be expected to do in Industry, look at *what is hot and what is not* and decide content for new modules.
<table>
<thead>
<tr>
<th>Module Timetable System Project</th>
<th>James Halstead</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapter 4: Results</td>
<td></td>
</tr>
</tbody>
</table>

The Central Teaching Space Timetable Information System: Student Office: This actor is actually a system that is responsible for the handling of module information for the entire University. Its main responsibility is allocating rooms based on module size and maintaining current timetable information.

Maths Department Administrator: Their main responsibility is to allocate and timetable modules for the Maths Department. This Actor is included in the Business Use Case Model to show the current situation for a different department. The problem area grows from this as some students have three joint honours departments a wide selection of electives.

SoC Director of Undergraduate Studies: In this context, their responsibility is to allocate and timetable modules for the SoC. I have used this actor’s title interchangeably with Module Administrator as a generic title for someone who would perform timetabling actions in the new system. Which is easier than referring to all individual departmental titles for the person or persons that is responsible for this and is meant as no disrespect.

**Business Use Cases**

Create a new COMP module: This use case is included to show that modules are sometimes created and need timetabling, as well as the need to just re-timetable existing modules. This use case starts when the SoC Learning and Teaching Committee decide that due to Industry standards and expectations a new COMP module should be produced.

Look up current timetable information: This use case starts when a module administrator needs to know what modules are currently occupying a timeslot. Currently each department operates separately and only looks at their local Information systems or that held centrally by the student office.

Allocate a MATH module a timeslot/ Allocate a COMP module a timeslot: The problem is with these two use cases is that: If two module administrators want to allocate a module the same timeslot, they have no way of knowing whether the students who are on both those modules will both be timetabled to attend those modules at the same time. This results in the joint honours student having a clash.
For more detailed information on the use cases and specification, please see Appendix B and Use Case Specification Forms.

Obviously the current practice is unacceptable and the system context diagram below is my first prototype of a new system design. Showing one my initial thoughts on a solution to the problem by combining existing systems to form a new Module Timetable (MT) system that would be a more efficient means of sharing information:

Fig.12 Initial System Context Diagram

Requests:
- Select a module
- View module details
- Search for a lecture theatre
- Search for timeslots
- Select modules for comparison
- Compare enrolment lists
- Change module to a new timeslot

Returns:
- Module Details
- Timeslots of availability for appropriate lecture theatres
- Modules that occupy these timeslots
- Comparison of enrolment list results
Inception Phase: Requirement Workflow Results

This section shows the major findings from conducting the requirements workflow of the RUP. The contents of which have been taken from the RUP artifacts and UML diagrams produced. For more detail see Appendix B artifact System Vision. The following shows the logic behind why a new system is required:

Re-cap Problem Statement

<table>
<thead>
<tr>
<th>The problem of</th>
<th>no collaboration between academic departments when timetabling modules that joint honours students may wish to attend.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Affects</td>
<td>students, module administrators and lecturers.</td>
</tr>
<tr>
<td>The impact of which is</td>
<td>students can’t attend all the modules they want due to clashes, lecturers numbers are reduced and module administrators have to re-timetable the module again, a costly process in terms of time (and confused students).</td>
</tr>
<tr>
<td>A successful solution would</td>
<td>mean a reduction in the number of clashes of joint honours students and reduce the time spent by module administrators trying to compensate for these problems.</td>
</tr>
</tbody>
</table>

System Position Statement

<table>
<thead>
<tr>
<th>For</th>
<th>joint honours in science module administrators at the University of Leeds.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Who’s</td>
<td>one role is timetabling in their department, allocate modules and move modules to timetable positions.</td>
</tr>
<tr>
<td>The new system</td>
<td>must be an information-sharing tool.</td>
</tr>
<tr>
<td>That</td>
<td>enables users access to course details, and compares student enrolment data.</td>
</tr>
</tbody>
</table>
Unlike the current system, where there is NO information sharing. The new system that will provide up-to-date information on module enrolment data and any popular/compulsory modules that are taken by those students. It would be accesses via the Internet.

**Market Demographics**

One of the key points brought forward by this table is that the users of this system will be University academic staff; therefore they are educated and likely to have some computer skills varying from basic to competent depending on department. Therefore, the system would have to be designed with these users in mind and take their profiles into account.

The market for this system will be localised primarily to the University of Leeds with possible future applications for other academic facilities. Also future releases of the system may yield a student version to help students when selecting electives to study, whether their module selection for the following semester would result in timetable clashes (for more information on this idea see J Cranshaw 2002 IS project). However the solution proposed to this problem for the time being is restricted to that of the University academic staff. Therefore the number of potential users will be between 0 –50 academics, the environment for which being any Internet connected PC in an office workplace. The following gives more detail to the stakeholders (anyone who has something to gain/lose from the system) and users (someone who will use the system).

**Stakeholder Summary**

<table>
<thead>
<tr>
<th>Name</th>
<th>Represents</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>SoC Director of Undergraduate Studies and Module Administrators</td>
<td>The need for successful and accurate module timetabling system</td>
<td>Ensures that modules are timetabled to a slot that results in as few clashes as possible for students and solves any additional room allocation problems</td>
</tr>
<tr>
<td>Student</td>
<td>The need for academic systems to run properly</td>
<td>Possible involvement for future system testing purposes. System Analyst and Designer</td>
</tr>
<tr>
<td>James Halstead</td>
<td>The need for a good project</td>
<td></td>
</tr>
</tbody>
</table>

25
User Summary

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Stakeholder</th>
</tr>
</thead>
<tbody>
<tr>
<td>Module Administrators for the University Joint Honours in science departments</td>
<td>Science administrators who deal with the timetabling of modules for each University department</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Stakeholder Profile: Students

<table>
<thead>
<tr>
<th>Representatives</th>
<th>Each department has student councils, these are student representative groups</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Stakeholder</td>
</tr>
<tr>
<td>Type</td>
<td>Wide range of technical backgrounds, won’t actually be using the system directly</td>
</tr>
<tr>
<td>Responsibilities</td>
<td>To act as the student voice for departmental issues, one of which is to make sure that students have the best and widest choice of University modules</td>
</tr>
<tr>
<td>Success Criteria</td>
<td>The stakeholder can define success by the number of students who have a successful timetable without clashes, and no problems if a module were to move during term time.</td>
</tr>
<tr>
<td>Involvement</td>
<td>It is envisaged that this stakeholder is involved in the project at the final stages, for testing purposes and isn’t involved in the creation or developmental stages.</td>
</tr>
<tr>
<td>Comments / Issues</td>
<td>Student is a main stakeholder of the system as they stand to get as much benefit as those who actually use the system. However, it is a system that won’t need their input until the final stages of testing, if at all, as it is an academic support system that students wouldn’t usually have access to. They are mentioned as stakeholders as they need to be aware of the problem and ensure that it is correctly solved.</td>
</tr>
</tbody>
</table>
### User Profiles: Module Administrators

<table>
<thead>
<tr>
<th><strong>Representative</strong></th>
<th>Of University staff who deal with module timetabling</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
<td>User; usually degree or higher qualification holders with varying amount of computer expertise, depending on the individual.</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>May be a sophisticated user, or relatively inexperienced with computers. The system will assume some basic knowledge of computers and a windows environment.</td>
</tr>
<tr>
<td><strong>Responsibilities</strong></td>
<td>These include the timetabling of modules, room allocations, timetable clashes and leasing with students who have associated timetable problems.</td>
</tr>
<tr>
<td><strong>Success Criteria</strong></td>
<td>Success is defined by a reduction in the number of problems students have with timetable clashes.</td>
</tr>
<tr>
<td><strong>Involvement</strong></td>
<td>SoC representative; Project Supervisor: They are involved in the initial requirements gathering, possibly with later development and testing phases.</td>
</tr>
<tr>
<td><strong>Comments / Issues</strong></td>
<td>This new timetabling software will take some control away from module administrators, as they will no longer be free just to choose any timeslot they like for a module. Therefore, there may be some resistance and limited help available in the undertaking of a production of a new system.</td>
</tr>
</tbody>
</table>
Elaboration Phase: Analysis and Design Results

The following section will detail the results from the Analysis and Design section of the RUP. It gives details of specific functions that a new system would have to achieve and a design of one such solution to the problem.

System Overview

Currently there is no system in place for the successful collaboration and sharing of this information between departments. The solution I propose is a new Module Timetable system that will combine database software that is already in place. The databases are the existing ‘Central Teaching Space Timetable Information System’ and individual department software like the SoC ‘Student Information System’ SIS.

System Functions

This system has two clear functions based on business modelling and the required functions that I have determined from the Stakeholder Request artifact, which need to be fulfilled by the new system. These are for use during term time, and for use before term time as the following states:

1) The timetabling of modules for the following year: The system will have all the information on department compulsory modules for a specific degree program, and will be able to cross-reference this information with other departments. For example if a COMP module was to be timetabled to 9 o’clock on a Monday morning, the system would be able to check to see if this was compulsory for any degree program and check to see whether they already have a module scheduled for that time.

2) For use during a semester: If a module needs to be moved to another timeslot after students have already registered for their electives. The system will allow users to check the enrolment lists of a selected module against those that occupy a specific timeslot to determine if the module was moved, whether students would have a clash.
### System Advantages

The following table summarises the advantages of a new system:

<table>
<thead>
<tr>
<th>Customer Benefit</th>
<th>Supporting Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Up-to-date timetable and enrolment information</td>
<td>The system accesses the central timetable database held by the student office for up-to-date information on modules. For each module, the module administrator can view module details, room allocations and access the enrolment data information lists for each department.</td>
</tr>
<tr>
<td>Search for enrolment information</td>
<td>All the enrolments for a particular module can be compared to a selected module in a particular timeslot, via simple search function. i.e. module administrator selects the module they want to move, find a timeslot they want, and search to see if anyone in that timeslot is enrolled on that module.</td>
</tr>
<tr>
<td>Search for lecture theatres</td>
<td>Once a Primary module has been selected, it is possible to search for an appropriate lecture theatre (only for use during term time as lecture theatres have already been allocated).</td>
</tr>
<tr>
<td>Instant feedback on student clashes</td>
<td>There is no guesswork involved in allocating modules a particular timeslot. The system instantly returns any clashes for any student enrolled on more than one module per timeslot.</td>
</tr>
<tr>
<td>Easy access to student and module details</td>
<td>In case contact is needed all the information on module and student details is to hand.</td>
</tr>
<tr>
<td>Access from any University PC</td>
<td>By providing user login ID and password. Users can</td>
</tr>
</tbody>
</table>
System Features

From this here is a summary of the key features that a new system would have to provide:

- The system will allow users the ability to select a Primary module; one that they would like to move to another timeslot i.e. Comp module.
- The system will be able to search the timetabled week and find the timeslots that have appropriate lecture theatres based on the module details i.e. RSLT24 Monday at 9 o’clock and Rupert Becket Tuesday at 5 o’clock.
- If during term time the system will allow users to select the modules that they wish to compare enrolment lists to in the selected timeslot, i.e. Mathematics and Physics modules at 9 o’clock on a Monday when RSLT24 is free. If before term time the system will bring up a list of modules and their timeslots that students are required to take if the selected primary module is compulsory to them.
- The system will be able to compare the enrolment lists of the Maths and Physics modules against the selected comp module and return a list of students that would have a clash. If there weren’t any clashes then the system will allow the user to go on and move the module to the new timeslot.

From this it is possible to produce a list is the system use cases, the key features of the system described in more detail but still at a relatively high level of abstraction. More details can be found in the Use Case Specification forms in Appendix B.

Login: Users have to logon to the system via a valid user name and password issued by the system administrator. There will be a change password feature for security reasons. There will be a user in each department, the module administrator who deals with the timetabling of modules.
Search/select a Primary module: Because the system is so large with so many different University modules, there will be a facility to search for a module either by its code, or by title/keyword search. There will also be a feature to select a module from a list, under departmental headings.

View Module Details: Once a module has been selected via the search feature, module details may be viewed. These include: course title/code, lecturer details, room allocation, current time slot, which semester it runs, and course content/assessment criteria. The system should allow the user to be able to browse through the enrolment lists for the module.

View Student/staff Details: The system will allow details of all students to be made available, i.e. what modules they are enrolled on. It will not be confidential exam information, just general enrolment and contact details. Also contact details of staff, given by department will also be given, so that if a module is moved, then the appropriate people can be contacted.

Search for other compulsory modules: If term has not started the system will allow the user to check see if the primary module they have selected is a compulsory module for a joint honours degree program or a normal degree program and if so view the times when these students have other compulsory modules timetabled in other departments.

Search for Lecture Theatre: If a module is moving timeslot during term time the user needs to know that if there will be a lecture theatre with sufficient space available some other time in that week as lecture theatres have now been allocated by the student office. Therefore, the system will allow the user to search for lecture theatres given the requirements of the module i.e. module size and special requirements i.e. the lecture theatre must have wheelchair access.

Select timeslot and modules: After the system has searched for lecture theatres (during term time) it returns a list of timeslots when they would be available. The system will allow the user to select which timeslots they deem appropriate. The system would then bring up a list of Joint Honours in Science modules that currently occupy those timeslot.

Compare enrolment lists: The user can then select from the previous list which modules (if not all) for comparison of against the selected Primary modules enrolment list.
Change module timeslot: Once a Primary module has been selected and an appropriate timeslot has been found with an appropriate room found, and an acceptable number of clashes, the user can change the module to a new timeslot. Note, the system should only allow a module administrator to move modules from their own department and not any other modules.

The primary function of the system is to compare modules enrolment lists of a module with other modules that occupy the same timeslot, solving the primary problem of a clash if the module is moved to that location. This will take the highest priority and the other features will be secondary. The following list shows the proposed features split up into primary and secondary functions.

<table>
<thead>
<tr>
<th>Primary</th>
<th>Secondary</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 &lt;Login&gt;</td>
<td></td>
</tr>
<tr>
<td>2 &lt;Search/select a Primary module&gt;</td>
<td>&lt;View Module Details&gt;</td>
</tr>
<tr>
<td>3</td>
<td>&lt;View Student/staff Details&gt;</td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5 &lt;Search for other compulsory modules&gt;</td>
<td></td>
</tr>
<tr>
<td>6 &lt;Search for a Lecture Theatre&gt;</td>
<td></td>
</tr>
<tr>
<td>7 &lt;Select Timeslots and modules&gt;</td>
<td></td>
</tr>
<tr>
<td>8 &lt;Compare Enrolment Lists&gt;</td>
<td></td>
</tr>
<tr>
<td>9 &lt;Change Module Timeslot&gt;</td>
<td></td>
</tr>
</tbody>
</table>

UML Results

The following section shows the annotated UML results of the new system from both the static and dynamic UML views. The text boxes that appear with arrows coming out that don’t connect to anything on some of the UML diagrams are there to explain the diagrams further, and are not actual official UML notation. The following gives a description for the purpose of the UML diagrams that are on the next few pages:

- Deployment Model (static): This diagram shows how the platform for the proposed MT system and the network connections.
- Class Diagram (static): This describes the types of classes in a system by their attributes and operations, and the varying kinds of relationships between these objects.
Module Timetable System Project

Chapter 4: Results

- Activity Diagram (dynamic): Shows the activities that the system will be able to perform and threads together the different system requirements.
- Sequence Diagram (dynamic): Show the normal flow of events and the sequence of messages that are sent and received between system objects.
- Analysis Model (dynamic): Shows the user interface UI and how the system will interact with the entities of the systems. (It also shows an alternative view to the system design as it takes the lecture theatre information from the module details as opposed to it being a separate entity).

*Diagrams without notation have been included in Appendix B.*

![Fig.13 System Deployment Diagram](image)

The reason for the integration of existing systems is to ensure that the data that the system uses is up to date and not second hand, as basic system functionality rests on the integrity of the data. Therefore, it is proposed that the module timetable system takes module timetable information from the student office and up to date enrolment information from departmental student software as to what modules students are enrolled on.
Module Timetable System Project

Chapter 4: Results

Fig. 14 System Class Diagram

Class Relationship: Shows the operations between classes, for example there is 1 MT System authorises 1 or many users.
If an invalid username of password is entered the system will allow no activities will be able to be performed.

An Active State:
Something that the system will allow user to do.
Fig. 16 System Sequence Diagram

(1) Objects:
- Actual things the system will use
  - Module Administrators
  - MT System
  - Enrolment List
  - Module List
  - Staff/Student Details
  - Lecture Theatre Booking List

(2) System Objects:
- Shows the message passing between the MT system (1) and the system objects (2).
**Boundary Classes**: How the actors interface with the system

**Control Classes**: are events or a set of events that take place within use cases that show the sequence of interaction between classes.

**Entity Classes**: The fundamental part of the system, the building blocks that the entire system relies on.

**Actors**: Someone who uses the system

**Fig. 17 System Analysis Model**
Supplementary System Specifications

This section includes the non-functional requirements of the proposed system that aren’t captured by the system functions outlined in the previous pages and in the UML diagrams.

Assumptions and Dependencies

The following is a list is of the assumptions and dependencies of the Module Timetabling System proposed in this document.

- Current timetable and department enrolment databases can be interfaced.
- All joint honours in science departments have individual student enrolment databases.
- System is dependant on integrity of enrolment and timetable information being correct.
- University will continue to operate for the next year.
- All module administrators will have to use the system to ensure functionality.
- Users have access to an Internet capable PC.

Alternatives and Competition to the system: Maintain Status Quo: The new system is an alternative to the current methodology that is employed. There may be resistance to a new system, especially as some control over timetabling will be removed from the module administrators. However, because the current practice is unacceptable, it is envisaged that the new system will be widely accepted.

Cost and Pricing: If software should be produced at a later date this software should be free to all, as part of the free software foundation ethic the source code should be readily available. Plus, it would be ‘unethical’ to charge for software produced as part of a degree program dissertation.

Licensing and Installation: There are no license requirements, however the system will only be used initially for the University of Leeds.

Functionality: All attempts to access the MT system should be logged, and repeated failed attempts to log in will result in an authorisation error and the system will not allow access to anyone from that machine.
Usability: Training: It is anticipated that the amount of time required for training the normal user on the system will be only a few hours as it should be designed to be intuitive and user friendly.

Usability: Browser Compatibility: The system will be web based and will be usable from multiple browsers. It should be designed to run on the latest version of Microsoft Internet Explorer, however the system will be accessible and viewable from other browsers such as Netscape Navigator.

Availability: Because the Module Timetabling system will be web based, it should be accessible from any computer connected to the Internet. However, it should only be accessible 9am-5pm University working hours to reduce the likelihood of system integrity being compromised by those with malicious intent towards the University.

Reliability: The system should be reliable 100 percent of the time all year round. The proposed target time for the Mean Time Between Failures (MTBF) and Mean Time To Repair (MTTR) should be approximately one hour, depending on who would manage the system.

Accuracy: The accuracy of the data in the system is of high concern as the system is only 100 percent functional if the data is accurate to that moment of use. Therefore, it will be taken that the student office timetable data is correct and the local departmental information data is current. The system will be updated immediately if when changes have been made to the database when a module is moved.

Performance: Number of Users: The number of simultaneous users that the system will support will be as many as are needed by the University departments. The system will allow multiple users at one time and module administrators for one department can only select and move modules from their specific department. However exact numbers should be decided after an initial version of the system has been tested. It may be that the number of users will have to be limited if the system response is slow after testing. A major issue is that the system will have to make sure that two departments aren’t relocating modules to the same timeslot at the same time.

Response Time: Search for a Lecture Theatre: It is expected that the response time will be only a few minutes to retrieve information on rooms that are available for the Primary selected module. This search is based on multiple fields for every timeslot in the University so the time taken to return list of appropriate lecture theatres may be more than a few minutes, depending on the number of simultaneous users at one instance.
Response Time: Compare Enrolment Lists: Once time slots have been selected with appropriate number of modules for comparison selected the system will have to compare enrolment lists. Again, this is a vast amount of data but the search should be quicker above as it is only based on student ID number and not multiple fields. Again, the time taken will be more accurate after the next iteration.

System Requirements: It is proposed that the system will require user terminals to have equivalent (or better) Pentium333 Processors, 64 MB RAM, and 10 GB hard drives. The system requires a Windows system running Windows 95/98 or Windows NT/2000 professional. With Internet Explorer version 5.0 or better.

Online Help: The online help function should be an on-line version of a user manual, based around the Windows help facility. It should allow topic search/selection and be updated regularly with a comprehensive troubleshooting guide.

Installation Guides, Configuration, and Read Me File: As the system is web based, no installation of software and configuration should be necessary.

Labelling and Packaging: As the system will not be marketed or packaged as it is part of Leeds University academic support network. It is expected that there be strong style and labelling conforming to University Web page design specifications.

Summary of Capabilities: The purpose of the system is to be an information tool to aid the allocation and timetabling of modules throughout the Joint Honours in Science departments. It will act as an academic support system and will be capable of looking up timetable information of other department modules to see whether those students enrolled would have a clash if the module were moved.
Chapter 5: Project Evaluation

Overview of Evaluation Criteria

This chapter is concerned with two areas; the evaluation of the tools used in this project and an evaluation the project itself. This will be in the form of an evaluation criteria that I have developed based on the Module Timetable system project and the tools that I have been using. I have used the Rational Unified Process and UML extensively during this project so a moderate section of this evaluation is devoted to discussing their limitations and weaknesses that I have encountered, and evaluation these tools. However, unlike a software project where there are easily available tools that evaluate reliability, consistency, functionality, usability of a system the same can’t really be applied as there is no real product made by this project that can adhere to that type of assessment. Therefore I have to find other ways to evaluate what I have accomplished so I have developed my own criteria to determine the project success, of which here is a summary of my evaluation findings:

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Summary of Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Objectives: Has this project met its</td>
<td>Objectives have been achieved and this project has met its minimum requirements and gone beyond them considerably, giving detailed analysis into the design of a new system.</td>
</tr>
<tr>
<td>minimum requirements, and how has it gone</td>
<td></td>
</tr>
<tr>
<td>beyond them?</td>
<td></td>
</tr>
<tr>
<td>Tools used: Were the RUP and UML the best</td>
<td>The UML tool has complimented this project well, it has a few short falls such as being hard to model a system when there isn’t really a rigid one in place, but all in all it has served well. The RUP however, may have been too heavy weight a process for such a small project.</td>
</tr>
<tr>
<td>tools for the project, what are their major</td>
<td></td>
</tr>
<tr>
<td>strengths/weaknesses?</td>
<td></td>
</tr>
<tr>
<td>Project management: Were all the RUP artifacts</td>
<td>The project was managed successfully and I have achieved more than I first set out to accomplish, although the project would have benefited by some kind of prototype software, which I should have planned for. However, the management major floor was that a meeting with my project</td>
</tr>
<tr>
<td>that were intended to be produced, actually</td>
<td></td>
</tr>
<tr>
<td>produced and deliverables met on time? What</td>
<td></td>
</tr>
<tr>
<td>problems were encountered was it managed</td>
<td></td>
</tr>
<tr>
<td>successfully.</td>
<td></td>
</tr>
</tbody>
</table>
supervisor never took place and missed an invaluable opportunity to get some feedback on what I was doing.

Improvements: What are the limitations to the analysis and design carried out in this project? What are the limits for the proposed system?

The major improvement identified was to make the UML clearer and more primary research into the problem area in other departments. Possibly through the use of interviews or questionnaires to the joint honour in science departments.

Areas for further study: What would be the next step on from this project, how would I be set about?

The next step would be to actually carry on with the RUP and start to build the system after researching techniques to go about it and learn the technical expertise. It almost seems a shame to have come so far with the project that I have to stop due to time constraints and not build at least some of the system.

**Project Objectives**

The first part of my criteria is to assess whether the minimum requirements of this project have been met. The minimum requirements of this project being (from page I):

- Research into the current system for timetabling modules and booking lecture theatres.
- Use business-modelling tools to analyse and model the existing module timetabling system.
- Make appropriate recommendations for enhancement.

The following details how I have surpassed these requirements by using the RUP and UML: Not only have I done considerable research into the timetabling and booking of lecture theatres, I have researched the methodologies and the history behind the techniques I have used. I have then used UML and the RUP as business modelling tools and produced a complete report (in Appendix B) that highlights the main problem area with the way joint honour student modules are timetabled and the existing system. Also not only I have made appropriate recommendations for enhancement by stating that a new system is required based on the problem area. I have expanded this considerably and gone into great detail using the RUP and UML as to the functional design and specifications of a new system.
**Evaluation of RUP and UML**

Based on my findings into the RUP part of my criteria is to decide whether the tools that I used were appropriate for this project. I have to say in summary of this subsection that the UML modelling of the problem and the design of the new system worked very well. I feel that UML truly is a great universal modelling language that is easy to learn and very flexible. Although it is true that it has little place when it comes to user interface designing, but I think that this will come about in later versions of UML, possibly version 2.0 scheduled for release later this year.

However I feel that there are a number of factors that have resulted in the RUP not being the methodology that I had hoped as I will try to explain in the following sections. The main difficult that I encountered was that the area I was dealing with was far too small to warrant the inclusion of an entire RUP project. I realised that the RUP is used predominantly for designing big information systems but I thought that it would be adaptable as the marketing literature claims its usage for small projects. I think that I underestimated the smallness of the project or the completeness or the RUP. A vast majority of the RUP artifacts even from the ‘roadmap for developing small projects’ were too complex and unnecessary for a project of this nature where all the requirements of the system were known. This led to sections of RUP artifacts being left out and whole documents that the RUP recommended as being ‘essential’ being left out entirely. Although the RUP is meant to be customisable for the project, there is a limit to how much tailoring can be done. In hindsight a more traditional approach, a structured methodology such as SSADM may have suited the size of the project better. However, the major advantage that the RUP has is that it does provide great templates for capturing requirements and documenting what has been done. Which with SSADM you don’t have the same explanations and detail that comes with the RUP.

I have adapted simple generic criteria for evaluating web pages (adapted slightly because I am evaluating an actual product, not just a web page) to highlight my findings whilst using this product. This use of such a criteria can be easily justified, as the Rational Unified Process is a web-enabled tool, the following table shows my findings:

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content/Information:</td>
<td>The information presented is very thorough</td>
</tr>
<tr>
<td>Module Timetable System Project</td>
<td>James Halstead</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Chapter 5: Project Evaluation</td>
<td></td>
</tr>
</tbody>
</table>

- Information on the topic is thorough?
- Easy to read the information, can you change the fonts?
- Correct spelling/grammar?

indeed and covers all aspects of the processes, workflows and the details behind them. Even if you had no experience of systems analysis and design, you would be able to understand the basic concepts of workflows, iterations and phases of the RUP in a short space of time.

Appreciation for the audience:
- Is the information contained too technical?
- Does it cover important topics in enough detail?

The information is presented at a level where if you just started to navigate your way around it, you would get confused. However, if you take your time and start from the beginning it gives good detail and definitions of the major parts that need to be remembered i.e. what a workflow is.

Structure/Navigation:
- Is the site logically organised?
- Is navigation clearly recognisable?

The actual layout and functionality of the RUP product encourages ease of use, with a standard windows style tree on the left hand side for browsing through its many different sections. This is in keeping with the RUP ethic of speed is ‘better than detail’ as it does allow fast navigation. It also has a very detailed search function and a complete index and glossary that make referencing, and finding definitions very easy.

Appearance:
- Graphics appropriate to content?
- Pleasing colour combinations?
- Page layout balanced, or cluttered?
- Are images smooth/appealing to the target audience?

The appearance of the site is quite plain; it doesn’t use any new techniques for displaying information or dancing tigers to highlight important sections. This reflects the content of the site as the information is presented professionally in an organised fashion. The heavy use of diagrams activity workflow diagrams makes it very appealing.

Access:
- Is the product quick to load?
- Is the load time acceptable?

The product only takes a few seconds to load and as the product is web enabled then it can be accessed from a wide variety of locations.

Cost of product: $695 (USD)
The RUP markets itself as being a complete tool for software engineering and has grown every year in size since its first development. As Anderson (2001) correctly identifies that “In order to facilitate sales, the process has begun to get more and more elaborate and more and more detailed”. This has resulted in a vast amount of information on every aspect of systems design that you could want. I have found that in some sections it just complicates the process as more time can be spent sifting through pages trying to understand the RUP jargon. Where is the RUP ethic ‘speed is better’ than detail here? It is a very heavy weight methodology that requires a lot of work, but a lightweight methodology might not have warranted the same results as far as a complete documentation and framework is concerned, which was one of my attractions to the RUP in the first place.

The RUP is so vast it requires a great deal of common sense to select the appropriate artifacts for a project and is more than just a set of instructions to follow. However, with a small system project, the documentation allows for a vast amount of repetition of information in some documents. I found that the Vision documents tended to include information that was applicable to subsequent stakeholder, requirements and risk documents. Much time can be spent documenting the same information again and again. However, in its defence I think this is only for small software projects as with large projects the amount of information specific to a particular document would be much larger, resulting in less repetition.

A significant problem is that the processes that the RUP encompasses are not geared towards user centred system design; something that I feel is paramount for a good methodology. Kruchten (2001) book on the RUP has the chapter heading “An Architecture Centric Process”. He has correctly identified this major weakness with this product. If it is an architecture centred process, then it can’t be user centred surely? Although in defence of the RUP it does allow some user centred design through the requirements set, however there is no specific notation for user interface (UI) design. A problem that extenuates to UML as Constantine identifies:

“UML provides neither a diagram type nor a notation for representing user interfaces in either abstract or realistic form.”

Constantine (2001)

User centred design is more popular now than ever with businesses consisting of such diverse people, technologies and practise, it is imperative that there is some level of user involvement, as Longworth
(1985) agrees: “Form the start of the project right down to detailed program design, the users should be continuously and closely involved”.

A general criticism of UML and other visual modelling techniques is the temptation to spend hours getting the diagrams to look as neat and as pretty as possible when you should be spending more time on the content and functionality of the diagram. Also, as I stated in my research chapters there are more types of UML diagrams that I neglected to include. This was however intentional as they either showed information already contained within other diagrams, or were unambiguous to the point of confusing. This is because all of the diagrams aren’t applicable to every situation. It is part of the designers job to decide which ones were to be included and neglected that show the important features. It is anticipated that as the system was developed, the more technical diagrams would be produced such as a component and statechart diagrams as system technical details and states becomes known. I also over-estimated the area that I was going to analyse and UML as is shown in the project, as there are only two UML diagrams to model the existing system. I hadn’t anticipated that it was so hard to model this in UML and thought that UML would be able to cope with it even though there is no actual functioning system in place. However, this was not the case and it took a lot of effort to accurately model the problem using UML.

As a visual modelling language, UML is truly platform independent. No specific tools are needed once it has been learned to produce diagrams (although software is available such as Rational Rose) you can get away with using just a pencil and paper. Because of the structure of UML it captures important aspects of a system and documents them in an easy to understand diagram form. It makes it appealing to people other than computer specialists such as managers and stakeholders, as at the correct level of abstraction can greatly help problem understanding and design issues. It is also usable with many development processes, which makes it far more widely used than other methodology specific tools. Such packages are also available which enable UML to be connected to programming languages to forward and reverse engineer code. In my opinion, the adaptive capabilities of UML are its strongest asset ensures its place as a standard tool for many years to come.

**Project Management**

From my own previous experience with smaller projects, and research I have conducted into project management, I feel the project ran relatively smoothly. Although after producing quite a detailed timeline in the initial stages of the project it was necessary to produce a new plan in the form of an
RUP artifact *Iteration Plan* once the project got underway. This was because as the project was becoming ever more complicated with RUP artifacts my initial plan just wasn’t detailed enough to cope with what I wanted to produce. I had initially intended for the major milestones in the project to take the form of UML diagrams and then RUP artifacts, as is shown in the Gantt Chart in my mid project report. However, when the project got going it was much easier and logical to complete my chosen RUP artifacts and produce the UML diagrams along the way when appropriate. It made sense then to use the RUP Inception and Elaboration phases as major milestones, with the Business Modelling, Requirements, Management and Analysis and Design core workflow sets becoming milestones within these phases. But still working towards the same deadlines for completing each phase as the content of the project hadn’t changed from my initial plan.

The main problem I encountered was to do with the actual selection of RUP artifacts and time management. I started with a long list of artifacts that I thought were appropriate to the project planned my time accordingly. However, as it turned out the artifacts I produced could then be tailored into one manageable document. I didn’t allow for this in my schedule and spent quite some time putting the information I required from several documents into one, thus having to adapt my remaining time to account for this.

**Improvements to the project**

The major assumption of the proposed MT system is that each department has the same problems as the SoC and has individual databases that hold details like the SoC on what modules students are enrolled on. It also assumes that each individual department would be happy to use such a system if it were actually designed. Therefore an improvement to the project would be to conduct further research, possibly some type of feasibility study into whether they would be able to use such a system, or indeed whether they have found the same problems that I have highlighted within the SoC.

I also feel that more time could have been spent on making the UML modelling clearer and presenting the results more efficiently. In this report I have had to annotate the diagrams considerably that takes away some of the quality although does improve understanding. The diagrams that have been produced act as a good aid that show the systems functions. However, I would have liked to shown the diagrams to someone who had never seen or heard of the project and they would be able to tell me what they mean. The people that I showed them to just got very confused and had puzzled looks on their faces. This may be attributed to the diagrams being a bit unambiguous and down to poor design.
on the authors part. Although I think that a criticism of UML is that you have to understand a bit about its semantics before you can truly appreciate and understand the aesthetic qualities of a sequence diagram.

**Further Study**

If I was to spend more time with this project and take it further, I would like to actually have a go at producing a functioning system for this problem. As I have made clear that it is not the objective of this project to go into the realms of software engineering, as my main area of interest in my IS degree program is system analysis and design. However, I think that the next logical step would be to continue through to the next phases of the RUP and actually write some software myself, or over see the production of a functioning system as I have good knowledge of the problem area and the systems required capabilities. I may lack the technical knowledge to finish the system, but I would like to attempt some software engineering as some of my fellow students have done with their final year projects. I have detailed quite considerably what the system will do, but not how to do it. Therefore I would initially start researching different techniques as to how I would actually accomplish the functions I have designed for the system. For example, exactly how the information from the databases would be connected to the MT web page, and what middleware software techniques I would have to learn. Then once I had researched it enough I would begin actually attempting the writing of a new system (their may be a software designer in me somewhere!). I have already started this by performing my initial risk assessment as to what the major risks would be in under taking this project in the future, as the following shows:

<table>
<thead>
<tr>
<th>Risk Ranking</th>
<th>Risk Description &amp; Impact</th>
<th>Mitigation Strategy and/or Contingency Plan</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>The interfaces between the departmental student information systems may be too complicated to construct.</td>
<td>Early exploration of departmental software and design of new sub-system if needed.</td>
</tr>
</tbody>
</table>
### Module Timetable System Project

<table>
<thead>
<tr>
<th></th>
<th>Project Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>James Halstead</td>
</tr>
<tr>
<td></td>
<td>Chapter 5: Project Evaluation</td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>May be too few users for the system to warrant a University wide project.</td>
<td>Limit the problem to SoC and associated joint-honours in science departments, instead of University wide.</td>
</tr>
<tr>
<td>2</td>
<td>System may not run on all department computers.</td>
<td>Research specifications of computers and analyse cost of upgrading.</td>
</tr>
<tr>
<td>1</td>
<td>Resistance to change and using the new system from users. It is only effective if people use it.</td>
<td>Demonstrate and persuade the effectiveness of the new system.</td>
</tr>
<tr>
<td>0</td>
<td>Having all the University module and enrolment details on one system</td>
<td>System Security and regular backup of data would be required</td>
</tr>
</tbody>
</table>

### The Last Word

Finally with determining the success of a project I find it prudent to end with a lighthearted question to the reader. The question being “Is the project a success?” to which you had to answer with a one-word answer. But what to make the basis for success as there are so many criteria to go on and issues to take into account as I have just laid out. If it were based solely on whether the project was managed successfully, you might say yes. If it were that the author now has a greater understanding of UML and RUP, then the answer might also be yes. If it were that the project is perfect without any floors, then you would most likely say no. If it were measured on the appropriateness of the tools used, and being an interesting read, then you might say maybe.

However on the balance of things and going on all the criteria that I have discussed in the previous sections, remember the authors answer because if I was asked to answer the fore mentioned question with one word I would say after some deep thought..........................“42”
Conclusion

I will now conclude this project by stating the conclusions that I have reached from conducting analysis and design into the Module Timetable System Problem.

From the start of this project it became obvious that the way that modules are currently timetabled in the SoC is unacceptable. There is a massive amount of risk in not checking to see whether joint honour students would have a clash if a module were to be moved during term time or a new module timetabled for the following year. However, with this practise there hasn’t yet been any culmination in major problems on a large scale. This is because from year to year there isn’t a great deal of variation between the timetable from the previous year and the one for the next year. However if it was ever needed to move a large amount of modules, then it would no doubt result in chaos for the students in the SoC and the associated departments for joint honours students. As those involved just wouldn’t have access to all the relevant information that is needed to complete the task successfully.

Therefore, the main conclusion reached in this project is that there needs to be a new system for the timetabling of modules if the SoC is going to run efficiently and have an effective module timetabling practice. I have described the core functions that one such system would need to perform to make the current practice acceptable. It would solve the notion of ‘problem ownership’ as to who would be responsible for finding a new timeslot for a clashing module thus reducing the time spent by those who do solve the problems walking around without all the information. They would be able to access the information they need and solve the problems encountered in a matter of minutes. However, the limitation of such a system like this would need to be used by all joint honours in science departments in order for it to work as the data the system would contain would have to be 100 percent accurate. This may be unrealistic, as there would most likely be resistance to a university wide implementation, as politics and attitudes in each department would be varied. As I have stated, more research would have to be conducted into this area of implementation.

My final thought is that although the SoC and associated departments don’t seem to have access to all the information, they do manage the successful timetabling of students each year. This is however a practice that lacks the efficiency you would expect form an institution such as Leeds University. I would hope in the next few years a new system or even practice that would take the issues highlighted in this report into account to result in a much more commonly accepted practice to the timetabling of University modules.
Appendix A

The purpose of this appendix is to reflect on the experiences that I have had with this project and give some recommendations to other students on the lessons that I have learned and the problems that I have overcome.

Overall I have found the project experience to be rewarding and I have met my own personal objective of gaining a greater understanding for the tools that I have used as I have had a chance to conduct a project into my chosen area of expertise. I found the key to this was to conduct good research and use all of the sources available to me; the Internet, lecture slides, books and final year projects are all very useful tools. These combined with my own reading and my previous knowledge of the subject area (from lectures and general reading) made for my project to run relatively smoothly. The project has consolidated my knowledge from my degree program and at least three of my SoC modules: IN21 Object Oriented Analysis and Design, IN22 Information Systems Development, IN33 People Centred information Systems Development. And obviously I wouldn’t have as broad a base for project management if it wasn’t for the Professional Development modules and SO22 Software Project Management.

I now have an appreciation for working on a sizable project and for me is unlike anything else I have achieved during my University career, or spent quite as much time and effort on. Working to a schedule makes for a more efficient use of time and planning goals to allow for the inevitable delays that all projects encounter by using buffer periods makes the project more enjoyable. I feel that the key to a good project is making the most of the time available and getting to grips with the subject at hand as soon as possible. I found that an efficient and flexible plan means that you don’t get left behind and pressured by other coursework deadlines. Also meeting the deadlines is a very powerful motivator as you feel that you have actually achieved something and have direction.

Guidelines for future students attempting an RUP/UML project

This project focuses mainly on the development methodology of the RUP, used in conjunction with UML. These are topics that are not widely used in final year projects (but covered to a good level in an IS degree syllabus) I have yet to find a project that uses the RUP this extensively as a major project focus. Therefore, being one of the first to use it I have a few guidelines on how it should be used, from the mistakes I have made and the lessons I have learned:
1) Don’t just complete artifacts for the sake of it, more artifacts that are produced that are rambling just confuse the reader and don’t explain the key issues well.

2) A few well-structured paragraphs in an artifact are better than filling in every heading just because it is there.

3) Don’t be afraid to delete parts that are inappropriate, the RUP artifacts are only there as templates and should be fully customised to suit the needs of the project.

4) Small perturbations: small goals are acceptable and wise; it is often the case that if you get bogged down with a large goal you will lose all motivation (and will to proceed).

5) Allow plenty of time to produce the artifacts; the RUP requires that the have many different iterations. So produce them fast and then produce them again, and again.

6) Make sure that the UML diagrams reflect an appropriate level of abstraction so that they can be understood, don’t get bogged down in low-level system functions to start with.

7) Have regular breaks, if you just sit at the computer for hours on end you increase the risk of RSI and losing concentration thus producing work of less quality than you are capable of.

8) Lastly, you can knock down a few pages and save on printer credits by combining the heading, the revision and the contents page of RUP artifacts into one manageable page.

Thanks for your time.
Bibliography


Centre for Joint Honours in Science, University of Leeds

Constantine.L (December 2000), *Can Rational Unified Process really facilitate a better experience?* Issue #10 Electronic Newsletter of User Centred Design


Harmelen.MV (2001) *Object Modeling and user Interface Design*, Designing Interactive systems, Addison-Wesley publishing

http://www.comp.glam.ac.uk/pages/staff/tdhutchings/chapter4.html [16th April 2002]

Johnson.O (2000/01) University of Leeds ‘Object Oriented, Analysis and design’ and ‘e-development’ computing modules


SSADM
http://www.precisiondesign.co.uk/SSADM.htm [26th April 2002]


Appendix B- Contents

RUP Artifacts Produced

INCEPTION PHASE.........................................................................................................................58

PROJECT GLOSSARY.........................................................................................................................58
ITERATION PLAN...............................................................................................................................60
BUSINESS CASE...............................................................................................................................64
STAKEHOLDER REQUESTS...............................................................................................................66
SYSTEM VISION...............................................................................................................................71
RISK LIST ........................................................................................................................................84
BUSINESS USE CASE SPECIFICATION: CREATE A NEW COMP MODULE...............................86
BUSINESS USE CASE SPECIFICATION: ALLOCATE A COMP MODULE A NEW TIMESLOT........87
BUSINESS USE CASE MODEL SURVEY..........................................................................................88
BUSINESS USE CASE DIAGRAM...................................................................................................90
BUSINESS SEQUENCE DIAGRAM ................................................................................................91

ELABORATION PHASE ................................................................................................................92

SYSTEM SEQUENCE DIAGRAM.................................................................................................92
SYSTEM ACTIVITY DIAGRAM..................................................................................................93
SYSTEM CLASS DIAGRAM.......................................................................................................94
USE CASE SPECIFICATION: LOGIN ..........................................................................................95
USE CASE SPECIFICATION: VIEW STUDENT/STAFF DETAILS ................................................97
USE CASE SPECIFICATION: SELECT A PRIMARY MODULE................................................98
USE CASE SPECIFICATION: VIEW PRIMARY MODULE DETAILS ........................................100
USE CASE SPECIFICATION: SEARCH FOR LECTURE THEATRES .........................................101
USE CASE SPECIFICATION: SELECT TIMESLOTS .................................................................103
USE CASE SPECIFICATION: SELECT MODULES FOR COMPARISON ..................................104
USE CASE SPECIFICATION: COMPARE ENROLMENT LISTS ..............................................105
USE CASE SPECIFICATION: MOVE MODULE .........................................................................106
SUPPLEMENTARY SPECIFICATION..........................................................................................108
SYSTEM DEPLOYMENT DIAGRAM............................................................................................111
SYSTEM ANALYSIS MODEL.......................................................................................................112
Inception Phase

Project Glossary

Introduction
This glossary contains all the definitions for the Module Timetable System Project.

Purpose
To be a quick reference guide when users come to read the subsequent documentation.

Scope
The document is applicable to the Module Timetable System being developed as part of a final year project by James Halstead (IS Degree 2002).

It is a system that will integrate existing departmental SIS software with central timetable information via the World Wide Web. This new system will solve the problem of joint honour student module clashes as module administrators will be able to check whether moving or creating a module in their own department will clash with another module from another department. They will be able to select the enrolment list for a particular module and check it against other department modules from a specific timeslot.

Overview
The rest of this document is in list format, defining each of the abbreviations and definitions used.

SIS: Student Information System
Systems local to each University department that hold information on students.

MT System: Module Timetable System
The name of the proposed new system to be developed.

‘The Product’ or ‘The System’
Another name for the MT System.
Central Teaching Space Timetable Information System
The system maintained by the student office that holds the timetable information for every department in the University.

SoC: School of Computing
University of Leeds department.

University or ‘the business’
Refers to Leeds University.

COMP Module or ‘module’
A module offered by the SoC or just a general University module.

Module Administrator
Someone who is responsible for the timetabling of modules for each department. (This may not be their actual job title, but one of their responsibilities).

User
Anyone that will be using the system, in this case it will be a departmental module administrator.

Timeslot
Is a one hour period of a day when a module will take place, may be more than one timeslot per week.

Timetable clash
When a student has two modules that occupy the same timeslot on a timetable.

Timetabling of Modules
The process of allocating a module timeslots on a timetable to reduce the number of timetable clashes.

Primary Module
A module selected by the user to compare its enrolment lists with that of other modules that occupy a selected timeslot.
Iteration Plan

Introduction

Purpose
This plan is a schedule for deliverables and artifacts produced by the MT system project in the Inception and Elaboration phases of the RUP.

Scope
The document is applicable to the Module Timetable System being developed as part of a final year project by James Halstead (IS Degree 2002).

This document will act as a plan and guidelines for what will be happening at each of the two stages conducted of the RUP. It will describe what is entailed at each phase and what artifacts will be produced and when, with a subsequent overview of the purpose of these documents.

Overview
In the following section there will be three tables, showing different levels of abstraction for the project. They provide a detailed schedule on what is to be produced for each of the four different activities with completion dates for each phase for the proposed MT system.

The following table shows at a high level of abstraction the core workflows of the RUP that this project will use:

<table>
<thead>
<tr>
<th>Activity</th>
<th>Start Date</th>
<th>End Date</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business Modeling</td>
<td>18&lt;sup&gt;th&lt;/sup&gt; Jan 2002</td>
<td>1&lt;sup&gt;st&lt;/sup&gt; Mar 2002</td>
<td>To conduct an investigation into the business nature and the systems currently in place and define the scope of the new system.</td>
</tr>
<tr>
<td>Requirements</td>
<td>15&lt;sup&gt;th&lt;/sup&gt; Feb 2002</td>
<td>15&lt;sup&gt;th&lt;/sup&gt; Mar 2002</td>
<td>To capture the requirements of the current system and define the required capabilities for the new system.</td>
</tr>
</tbody>
</table>
### Analysis and Design

<table>
<thead>
<tr>
<th>Task Name</th>
<th>Start</th>
<th>Finish</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business Modeling</td>
<td>18th Jan 2002</td>
<td>1st Mar 2002</td>
</tr>
<tr>
<td>Capture a Common Vocabulary</td>
<td>Mon 21st Jan</td>
<td>Friday 8th Feb</td>
</tr>
<tr>
<td>Find Actors and Business Use Cases</td>
<td>Mon 3rd Feb</td>
<td>Friday 8th Feb</td>
</tr>
<tr>
<td>Business Use Cases Description Forms</td>
<td>Mon 11th Feb</td>
<td>Friday 15th Feb</td>
</tr>
<tr>
<td>Business Use-Case Model</td>
<td>Mon 18th Feb</td>
<td>Friday 22nd Feb</td>
</tr>
<tr>
<td>Review Business Modeling</td>
<td>Friday 22nd Feb</td>
<td>Friday 1st Mar</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Task Name</th>
<th>Start</th>
<th>Finish</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements</td>
<td>15th Feb 2002</td>
<td>15th Mar 2002</td>
</tr>
<tr>
<td>Develop Project Vision</td>
<td>Mon 18th Feb</td>
<td>Friday 8th Mar</td>
</tr>
<tr>
<td>Capture Stakeholder Requests</td>
<td>Mon 18th Feb</td>
<td>Friday 23rd Mar</td>
</tr>
<tr>
<td>Manage Dependencies</td>
<td>Mon 18th Feb</td>
<td>Friday 15th Mar</td>
</tr>
<tr>
<td>Capture a Common Vocabulary</td>
<td>Mon 18th Feb</td>
<td>Friday 23rd Mar</td>
</tr>
<tr>
<td>Find Actors and Use Cases</td>
<td>Mon 26th Mar</td>
<td>Friday 1st Mar</td>
</tr>
<tr>
<td>Prioritise Use Cases</td>
<td>Mon 11th Mar</td>
<td>Friday 15th Mar</td>
</tr>
<tr>
<td>Define System &amp; Constraints</td>
<td>Mon 11th Mar</td>
<td>Friday 15th Mar</td>
</tr>
<tr>
<td>Review Project Requirements</td>
<td>Mon 11th Mar</td>
<td>Friday 15th Mar</td>
</tr>
</tbody>
</table>

### Management

<table>
<thead>
<tr>
<th>Task Name</th>
<th>Start</th>
<th>Finish</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identify Risks</td>
<td>Mon 4th Mar</td>
<td>Friday 8th Mar</td>
</tr>
<tr>
<td>Develop Project/Phase Plan</td>
<td>Mon 11th Mar</td>
<td>Friday 15th Mar</td>
</tr>
<tr>
<td>Generate Business Case</td>
<td>Mon 4th Mar</td>
<td>Friday 15th Mar</td>
</tr>
</tbody>
</table>

This table shows in more detail what is produced for each of the workflows outlined above:
<table>
<thead>
<tr>
<th>Generate Business Case</th>
<th>Mon 4th Mar</th>
<th>Friday 15th Mar</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Analysis and Design</strong></td>
<td><strong>1st Mar</strong></td>
<td><strong>15th April</strong></td>
</tr>
<tr>
<td>Perform Architectural Analysis</td>
<td>Mon 4th Mar</td>
<td>Friday 22nd Mar</td>
</tr>
<tr>
<td>Use Case Analysis</td>
<td>Mon 25th Mar</td>
<td>Friday 29th Mar</td>
</tr>
<tr>
<td>Identify Design Elements</td>
<td>Mon 1st April</td>
<td>Friday 12th April</td>
</tr>
<tr>
<td>Develop Architecture Document</td>
<td>1st Mar</td>
<td>15th April</td>
</tr>
<tr>
<td>Review System Requirements</td>
<td>1st Mar</td>
<td>ONGOING</td>
</tr>
</tbody>
</table>

**Project Deliverables**

This table shows the proposed deliverables, some of which are RUP artifacts, which are to be produced during the inception and elaboration stages outlined by the previous table. This is the initial list of deliverables; it is likely that it will be modified depending on the direction of the project.

<table>
<thead>
<tr>
<th>Artifact Set</th>
<th>Deliverable</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Business Modeling</strong></td>
<td>Glossary</td>
</tr>
<tr>
<td></td>
<td>Business Use Case specifications</td>
</tr>
<tr>
<td></td>
<td>Business Use Case Model</td>
</tr>
<tr>
<td></td>
<td>Business Object Model</td>
</tr>
<tr>
<td></td>
<td>Supplementary Business Spec.</td>
</tr>
<tr>
<td></td>
<td>Business Use Case Realization</td>
</tr>
<tr>
<td><strong>Requirements</strong></td>
<td>Project Vision Document</td>
</tr>
<tr>
<td></td>
<td>Stakeholder Requests Document</td>
</tr>
<tr>
<td></td>
<td>Use Case Specifications</td>
</tr>
<tr>
<td></td>
<td>Supplementary Specification</td>
</tr>
<tr>
<td></td>
<td>Requirements Use Case Model</td>
</tr>
<tr>
<td><strong>Analysis and Design</strong></td>
<td>Software Architect Document</td>
</tr>
<tr>
<td></td>
<td>Use Case Model</td>
</tr>
<tr>
<td></td>
<td>Deployment Model</td>
</tr>
<tr>
<td></td>
<td>Analysis Model</td>
</tr>
<tr>
<td></td>
<td>Design Model</td>
</tr>
</tbody>
</table>
Management
- Project Plan
- Project Risk List
- Business Case Document
- Status Assessment

Resources
- Rational Rose Enterprise Edition $4495 USD
- Rational Unified Process $695 USD

Use Cases
The use cases for this project will be detailed in two sections. These will be: A) the initial use cases developed as part of the business modelling, to show how the problem area is currently being addresses B) The use cases for the new proposed Module Timetable System. Both A and B will be defined with use case description forms and use case models.

Evaluation Criteria
The evaluation for this project will come after the proposed artifact sets have been produced. This will be a decision, based on the documentation collected so far as to whether the system would be viable to implement and go further on with the RUP phases.
Business Case

Introduction

Purpose
The purpose of this document is to give a broad overview of the proposed MT system together with an organizational overview of its environment.

Scope
The document is applicable to the Module Timetable System being developed as part of a final year project by James Halstead (IS Degree 2002).

Overview
The two main sections of this document are the ‘Product Objectives’ and the Business Context’. This is because they define what the system will do and in what context.

Product Description
This product will be a web-based system that will act as an administrative tool for the University. It is proposed that the system will interface current information held by the Student Office in the Central Teaching Space with that of departmental SIS module enrolment information. This will allow the users to determine if by moving a selected module to a new timeslot whether there would be an appropriate lecture theatre or seminar room for that module, and assess if students enrolled would have a timetable clash with any of their other modules (compulsory or elective). The system will also allow users to be able to view module and student/staff information, as a secondary function of the system.

This is a problem that is core to the timetabling of modules because as it stands currently, there is no successful communication between departments. There is no way of knowing whether the allocating of modules to a new timeslot will result in clashes of joint honours students compulsory modules in other departments.

Business Context
This product although it will be web based will only be for authorized administrators of the University. It will be web based because it will be easier to develop due to the complex nature of the University network topology. Therefore, it is desirable to make the product web based with secure password access to selected users.
The product is being developed as a support function to the already existing timetable and enrolment information services to integrate current student database information. Currently, the University has no system in place for the comparison of module enrolment lists with that of other modules, this will be the primary function of the new system.

**Product Objectives**

The objective of this product is to reduce the number of clashes that students have due to module administrators having accurate up to date information on other departmental modules. It is beneficial to the students because if a module is moved during a semester, the user will have checked to see if there are any subsequent module clashes for the students, thus reducing students panic and confusion. Module administrators benefit from the system because they will have information to hand that will tell them if they move a module to a new location, who will be affected and how seriously. There would no longer be a ‘wait and see’ attitude and discover that a new module (or a moved module) has a considerable number of student clashes well into term time, thus reducing the workload of module administrators considerably.

Therefore the two main objectives of this software are as follows:

1. To allow users to compare the enrolment list of a selected module to that of any other university modules to determine whether moving a module to a new location is viable.
2. To allow users to determine if a module is to be moved whether there will be a suitable lecture theatres or seminar rooms available based on the requirements of that specific module.

**Financial Forecast**

Not really applicable for this system as it is being developed as part of a final year IS degree and the ethics of charging for this product would be questionable at best!

The ROI that I can identify is the investment of time and energy of the author would be great enough to constitute a good degree classification.

**Constraints**

The main constraint of this system is whether it can actually be actually implemented. It may not be possible to produce the software in a feasible time that will interface all the departmental SIS software with that of the Central Teaching Space Timetable Information System.

Also See Risk List Document.
Stakeholder Requests

Introduction

Purpose
This document is produced to gain understanding as to what the user requires and the purpose of the system. This is core to the development of a successful system as the stakeholders in this project are the users, so they will be the ones that actually use the system. If the system is not successful in capturing the stakeholder needs and requests then it may result in the production of a redundant system.

Scope
The document is applicable to the Module Timetable System being developed as part of a final year project by James Halstead (IS Degree 2002).

Some of the information contained is inferred due the nature of the project and because the there are not any real stakeholders as a final system will never be produced. However, it is written as if the project would carry on and a system produced. The answers are based on requirements and interviews with Dr K McEvoy.

Overview
This document contains the results from the information gathered from the stakeholders of the system.

Establish Stakeholder or User Profile

- Name: Dr Kevin McEvoy
  Company / Industry: University of Leeds
- Job Title: Director of Undergraduate Studies SoC
- What are your key responsibilities? Timetabling of Computing Modules
- What deliverables do you produce? For whom?
  Not relevant.
- How is success measured?
  Success is a hard to quantify, student happiness maybe and time reduced running around finding out is students will have a clash if I move a module.
- What problems interfere with your success?
  When a module is moved, it is hard to tell whether there will be any ramifications during a semester for the students.
- What, if any, trends make your job easier or harder?
Same problems with modules clashing and lecture theatres not being assigned properly occur every year.

Assessing the Problem

- For which problems do you lack good solutions?
  At the moment there is a distinct lack of information between joint honours science departments, they don’t really know what each other is doing with regards to their timetabling of modules. There is no one person who has clear responsibility for module clashes.

Understanding the User Environment

- Who are the users?
  Module Administrators for the University.
- What is their educational background?
  The users of this system will be university academic staff; therefore they are educated with varying degrees or possibly higher qualifications.
- What is their computer background?
  Likely to have some computer skills varying from basic to competent depending on which department they work for.
- Are users experienced with this type of application?
  Yes, they have experience with module timetable software.
- Which platforms are in use?  What are your plans for future platforms?
  They use Windows 98, plans in the future to upgrade to Windows 2000 Professional.
- Which additional applications do you use that we need to interface with?
  SIS and Central Teaching Space Timetable Information System.
- What are your expectations for usability of the product?
  It is expected that Module Administrators University wide will use the application, not just the joint honours science departments involved currently in the main problem area.
- What are your expectations for training time?
  A few hours at most.
- What kinds of hard copy and online documentation do you need?
  Some documentation on-line and a person to contact in case of failure, possibly a user manual.

Recap for Understanding

- You have told me:
- No Information sharing between departments.
- You have no way of knowing whether moving a module to a new timeslot will result in joint honours students having module clashes.
- The system should be easy to use and integrate with current module information systems.

- Does this represent the problems you are having with your existing solution?
  Yes, there is no current existing solution.
- What, if any, other problems you are experiencing?
  None that are concerned with the problem area.

**Analyst’s Inputs on Stakeholder’s Problem (validate or invalidate assumptions)**

- Lecture theatre allocation problem:
  There is no way of knowing when there will be a lecture theatre free for the module. The current practise is unacceptable and time consuming is as follows:
  It is to find a time when there are lecture theatres free, and find those big enough (by contacting the student office) check to see if these new time clashes with any other modules that students are taking (by walking into the lecture theatre and asking students then) and then move it accordingly.

**Assessing Your Solution (if applicable)**

Key Capabilities of the system therefore are:

- To allow users to compare the enrolment list of a selected module to that of any other university modules to determine whether moving a module to a new location is viable.
- To allow users to determine if a module is to be moved whether there will be a suitable lecture theatres or seminar rooms available based on the requirements of that specific module.
- Contact details of staff and students and details of modules.

**Assessing the Opportunity**

- Who needs this application in your organization?
  Module administrators in the joint honours science departments.
- How many of these types of users would use the application?
10-15 people, possibly more, depending on how many departments will be involved.

- How would you value a successful solution?
  Less time would be spent by module administrators as they would no longer have to walk around trying to find out whether moving a module to a new time will be catastrophic, information would be readily at hand.

Assessing Reliability, Performance, and Support Needs

- What are your expectations for reliability?
  System must be reliable and keep up to date information on ALL University modules.

- What are your expectations for performance?
  Must run reasonably fast, but because there is an amazing amount of module and timetable and student and lecture theatre information, so a somewhat slow system may be unavoidable.

- Will you or others support the product?
  Yes, but some departments may think that it is just another unnecessary piece of software due to their resistance to change.

- Do you have special needs for support? What about maintenance and service access?
  As long as there is someone to contact in the event of a system failure, and someone to show me how to use it, that is all I require.

- What are the security requirements?
  The information contained in the system is crucial as it will be the one source of information on the timetabling of modules. So if someone with malicious intent were to gain access and compromise the integrity of the information it would be catastrophic to the University. So careful thought must be given to secure access to the system to prevent this.

- What are the installation and configuration requirements?
  The product ideally would be web-based so no configuration of systems would really be required.

- What are the special licensing requirements?
  None as I expect it to be freeware?

- How will the software be distributed?
  Will be web-based, so no distribution will be necessary.

- What are the labelling and packaging requirements?
  None see above.

Other Requirements

- Which, if any, regulatory or environmental requirements or standards must be supported?
No.

- Can you think of any other requirements we need to know about?
  The functionality of the system is core, as every module administrator will have to use it to ensure accurate information.

Wrap-Up

- Are there any other questions I should be asking you?
  Not at this time.

- If I need to ask follow-up questions, may I give you a call?
  Sure, come in and see me

- Would you be willing to participate in a requirements review?
  Isn’t that what this is?

Analyst’s Summary

1. There is no way of knowing whether the students enrolled on a computing module will have a clash with another module when it is moved. So there needs to be some way of comparing enrolment lists, possibly via local departmental software on students enrolment.

2. When the modules are allocated lecture theatres, do they take into account the needs of the module, i.e. disabled access, or computer projection equipment?

3. The primary problem area of problem ownership, who is responsible if a serious module clash does occur? Should it be handled by one department, if so which one? Or should parent joint honours department handle it?
System Vision

Introduction

The following Vision document is written to provide the stakeholders with an overview of the proposed system in terms of its needs and features.

The final system will not be constructed as this project is only concerned with Analysis of the way modules are timetabled and the Design for a better system, by using the Inception and Elaboration phases of the RUP. Although there will be no actual construction of a functioning system, the following document is written as if a system will be made.

Purpose

The purpose of this document is to define the key aspects and components of Module Timetabling System within the school of computing.

Scope

This document will form a section of the Analysis into the way modules are timetabled within the School of Computing (SoC) and the problems associated with the current methodology.

Positioning

Business Opportunity

This project will be analysing the current methodology for timetabling SoC and University modules and make recommendations for a new MT system. At the moment modules are timetabled locally with no collaboration between departments or thought for joint-honour student timetable clashes.

The business opportunity here is to make the University of Leeds a more efficient academic business by modernizing the timetabling system through effective information sharing between departments. The new system will make sure that students can have the widest selection of modules, from a range of departments without having to compromise their learning by substituting for other modules due to timetable clashes. Thus in the long term attracting more students due to an effective academic institution.
Module administrators will be able to access the system through any personal computer that has an Internet connection and look up current and up-to-date timetable information and plan appropriately the best time slot for a module.

**Problem Statement**

<table>
<thead>
<tr>
<th>The problem of</th>
<th>no collaboration between academic departments when timetabling modules that joint honours students may wish to attend.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Affects</td>
<td>students, module administrators and lecturers.</td>
</tr>
<tr>
<td>The impact of which is</td>
<td>students can’t attend all the modules they want, lecturers numbers are reduced and module administrators have to re-timetable again, a costly process in terms of time (and confused students).</td>
</tr>
<tr>
<td>A successful solution would</td>
<td>mean a reduction in the number of clashes of joint honours students and reduce the time spent by module administrators trying to compensate.</td>
</tr>
</tbody>
</table>

**Product Position Statement**

<table>
<thead>
<tr>
<th>For</th>
<th>module Administrators at the University.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Who</td>
<td>allocate modules and move modules to timetable positions.</td>
</tr>
<tr>
<td>The Module Timetabling System</td>
<td>is a web-based information-sharing tool.</td>
</tr>
<tr>
<td>That</td>
<td>enables users access to course details, and compares student enrolment data.</td>
</tr>
<tr>
<td>Unlike</td>
<td>the current system, which has NO information sharing.</td>
</tr>
<tr>
<td>Our product</td>
<td>will provide up-to-date information on module enrolment data and any popular/compulsory modules that are taken by those students. It would be access via the Internet with restricted</td>
</tr>
</tbody>
</table>
Stakeholder and User Descriptions

The stakeholders in this project are the module administrators and the University students.

Market Demographics

The users of this system will be university academic staff; therefore they are educated and likely to have some computer skills varying from basic to competent depending on department.

The market for this product is academic facilities that have no centralized control over the timetabling of modules for different academic departments. The market for this system will be localized primarily to the University of Leeds. However, it is not unlikely that other learning institutions may have a similar problem. Therefore the number of potential users primarily, will be between 0 –50 academics for Leeds University.

The product is initially for academics but it is hoped that there will be a later follow-up release for students to use to check electives and chose modules.

Stakeholder Summary

<table>
<thead>
<tr>
<th>Name</th>
<th>Represents</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>SoC Module Administrator</td>
<td>The need for successful and accurate module</td>
<td>That modules are timetabled to a slot that results in as few classes as possible for students.</td>
</tr>
<tr>
<td></td>
<td>timetabling</td>
<td></td>
</tr>
<tr>
<td>Student</td>
<td>Students</td>
<td>Only involved for testing purposes.</td>
</tr>
</tbody>
</table>

User Summary

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Stakeholder</th>
</tr>
</thead>
</table>
Module Administrators | Manages the timetabling of modules for a University department | Yes

User Environment
The environment for this system will be any Internet connected PC in an office workplace.

Stakeholder Profiles

Students

<table>
<thead>
<tr>
<th>Representatives</th>
<th>Each department has student councils, these will be the basis of the student representative groups</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Stakeholder</td>
</tr>
<tr>
<td>Type</td>
<td>Wide range of technical backgrounds, won’t actually be using the system directly</td>
</tr>
<tr>
<td>Responsibilities</td>
<td>That the system makes sure that students have the best and widest choice of University modules</td>
</tr>
<tr>
<td>Success Criteria</td>
<td>The stakeholder can define success by the number of students who have a successful timetable without clashes, and no problems if a module were to move during term time.</td>
</tr>
<tr>
<td>Involvement</td>
<td>It is envisaged that this stakeholder is involved in the project at the final stages, for testing purposes and isn’t involved in the creation or developmental stages.</td>
</tr>
<tr>
<td>Deliverables</td>
<td>None.</td>
</tr>
<tr>
<td>Comments / Issues</td>
<td>Student is a main stakeholder of the system as they stand to get as much benefit as those who actually use the system. However, it is a system that won’t need their input until the final stages of testing, if at all, as it is an academic support system that students wouldn’t usually have access to. The are involved as they need to be aware of the problem and ensure that it is correctly solved.</td>
</tr>
</tbody>
</table>

User Profiles

Module Administrators

| Representative | The representative will be from the SoC |

74
Module Timetable System Project

System Vision

<table>
<thead>
<tr>
<th>Description</th>
<th>User; usually degree or higher qualification holders with varying amount of computer expertise, depending on the individual.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>May be a sophisticated user, or relative inexperienced with computers. The system will assume some basic knowledge of computers and a windows environment.</td>
</tr>
<tr>
<td>Responsibilities</td>
<td>These include the timetabling of modules, room allocations, timetable clashes and leasing with students who have associated timetable problems.</td>
</tr>
<tr>
<td>Success Criteria</td>
<td>Success is defined by a reduction in the number of problems students have with timetable clashes.</td>
</tr>
<tr>
<td>Involvement</td>
<td>Project Supervisor: They are involved in the initial requirements gathering, development and testing phases.</td>
</tr>
<tr>
<td>Deliverables</td>
<td>None</td>
</tr>
<tr>
<td>Comments / Issues</td>
<td>This new timetabling software will take some control away from module administrators, as they will no longer be free just to choose any timeslot they like for a module. Therefore, there may be some resistance and limited help available in the undertaking of this project from these users.</td>
</tr>
</tbody>
</table>

Key Stakeholder or User Needs

The users need to be able to see if there is going to be major timetable clashes for joint-honours students before a module is allocated a new timeslot. Currently, information on timetabling is held in the student office and is accessible at www.leeds.ac.uk/timetable. However, there is no system in place for checking one modules enrolment list against another that may run at the same time.

<table>
<thead>
<tr>
<th>Need</th>
<th>Priority</th>
<th>Concerns</th>
<th>Current Solution</th>
<th>Proposed Solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Module Timetabling System</td>
<td>High</td>
<td>Current System is unacceptable</td>
<td>Currently, there is no system in place for checking to see whether students are enrolled on modules that will share the same timetable space, i.e. a student may be enrolled on MATH201 on Monday 9 and COMP3591 is moved</td>
<td>Module Administrators will be able to see major clashes before they occur using the new system as it has all information on module enrolment lists.</td>
</tr>
</tbody>
</table>
to 9 o’clock. This results in either the student changing module, if optional, or the module being re—timetabled again.*

*The existing solution to the problem is to hold information locally on what other departments are doing with regard to the timetabling of compulsory modules i.e. the Math’s department holds information on the time of the compulsory Computer modules that joint honours in Math’s and Computing students take. Thus making sure that any timetable adjustments the Maths department make, do not interfere with these compulsory modules. The problem is that this information is not updated regularly and there are no clear lines of communication of systems in place to share the data when modules change timeslots and even semesters. The problems with these clashes are therefore solved, as and when they arise, on an individual department-department basis.

Alternatives and Competition

Maintain Status Quo
The new system is an alternative to the current methodology that is employed. There may be resistance to a new system, especially as some control over timetabling will be removed from the module administrators.

Product Overview
From a high level point of view, the new system will be web-based, with restrict user and password access. It will link those timetable systems that are already in place to a new web based system.

Product Perspective
Currently there is no system in place for the successful collaboration and sharing of this information between departments. The new system will combine the existing ‘Central Teaching Space Timetable Information System’ with individual department software like the SoC ‘Student Information System’ SIS. The next context diagram shows one proposed solution:
Summary of Capabilities

The primary function of the system is to be an information tool to aid the allocation and timetabling of modules throughout the University. It will act as a student support system and allow users to look-up timetable information of other school’s modules and those students enrolled. There will be additional information on the content of the module and provided will be statistics of other popular modules chosen by those students enrolled on that module. The system will also provide additional details of staff that are responsible for that module.

The main function of the product is to be a complete tool for the allocation of modules throughout the University. It will provide reference to users when they wish to relocate a module to a different
timeslot and easily check if any students on the module they are changing are allocated to the new
timeslot. Once an appropriate timeslot has been established, module administrators would be able to
change their modules to that timeslot.

<table>
<thead>
<tr>
<th>Customer Benefit</th>
<th>Supporting Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Up-to-date timetable and enrolment</td>
<td>The system accesses the central timetable database held by the student office for up-to-date information on modules. For each module, the module administrator can view the timeslot, the lecturer, and room allocation. From this they can access the enrolment data information lists</td>
</tr>
<tr>
<td>information</td>
<td></td>
</tr>
</tbody>
</table>
| Search for enrolment information          | All the enrolments for a particular module can be compared to selected modules in a particular timeslot, via simple search function. i.e. module administrator selects the module they want to move find a timeslot, search to see if anyone in that timeslot is enrolled on the current module. |}

<table>
<thead>
<tr>
<th>Instant Feedback on student Clashes</th>
<th>There is no guesswork involved in allocating modules a particular timeslot. The system instantly returns any clashes for any student enrolled on more than one module per timeslot.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Easy access to student and module details</td>
<td>By providing user login ID and password. Users can access student information systems across the University.</td>
</tr>
<tr>
<td>Access from any University PC</td>
<td>Users can access the information from any PC that has Internet Access.</td>
</tr>
<tr>
<td>Secure System</td>
<td>To gain access to the system a valid username and ID must be entered.</td>
</tr>
</tbody>
</table>
Assumptions and Dependencies
The following is a list is of the assumptions and dependencies of the Module Timetabling System proposed in this document.

- Current timetable and department enrolment databases can interface.
- System is dependant on integrity of enrolment and timetable information being correct.
- University will continue to operate for the next year.
- All module administrators will have to use the system to ensure functionality.
- Users have access to an Internet capable PC.

Cost and Pricing
This software will be free to all, as part of the free software foundation ethic as the source code will be readily available. Plus, it would be ‘unethical’ to charge for software produced as part of a degree program dissertation.

Licensing and Installation
There are no license requirement, however the system will only be used initially for the University of Leeds.

Product Features
The following list is the systems key feature, described at a high level of abstraction

Login
Users have to logon to the system via a valid user name and password issued by the system administrator. There will also be a change password feature.

Search/select a module
Because the system is so large with so many different University modules, there will be a facility to search for a module either by its code, or by title/keyword search. There will also be a feature to select a module from a list, under departmental headings.

View Module Details
Once a module has been selected via the search feature, module details may be viewed. These include: course title/code, lecturer details, room allocation, current time slot, which semester it runs,
and course content/assessment criteria. You will also be able to browse through the enrolment lists for the module.

Select timeslot and modules
After a module has been selected, the system will allow the user to select a new timeslot. The user will be able to select and limit the departments they wish to compare the selected module to.

View Student Details
The system will allow details of all students to be made available, i.e. what modules they are enrolled on. It will not be confidential exam information, just general enrolment and contact details.

Search for Lecture Theatre
If a module is moving timeslot the user needs to know that if they allocate a new timeslot, there will be a lecture theatre with sufficient space available for that slot. Therefore, the system will allow the user to search for a room in any given timeslot, given the requirements of the module i.e. must have wheelchair access.

Change module timeslot
Once a module has been selected and an appropriate timeslot has been found with an acceptable number of clashes, and an appropriate room found, the user can change the module to a new timeslot. Note; only a module administrator can move modules from there own school and not any other modules.

E-mail an administrator
The system will allow module administrators to e-mail other administrators to make requests for modules to be moved, or for more details given.

Quality Ranges
The performance of the system will be limited to the technical specifications of the machine it is being used. Obviously there will be a large amount of data to search through, so at times the system may be slow to use.

Because the Module Timetabling system will be web based, it will be accessible from any computer connected to the Internet. However, it will only be accessible 9am-5pm University working hours to reduce the likelihood of system integrity being compromised by those with malicious intent towards the University.
The System shall include online help for the user, contact details of first point of call for faults. Users will also be provided with a hard manual copy of how to use the system.

**Precedence and Priority**

This section defines the priority of the proposed system features outlined in section 5 of this document. It is hoped that there will be only a single release of the system, including all the features outlined above.

However, the primary function of the system is to compare modules enrolment lists of a module with other modules that occupy the same timeslot. This will take the highest priority and the other features will be secondary. The following list shows the proposed features split up into primary and secondary functions:

### Primary |
### Secondary

| 5.1  | <LOGIN>        |
| 5.2  | <SEARCH/SELECT A MODULE> |
| 5.3  | <VIEW MODULE DETAILS> |
| 5.4  | <COMPARE/VIEW ENROLMENT DETAILS> |
| 5.5  | <VIEW STUDENT DETAILS> |
| 5.6  | <SELECT TEACHING REQUIREMENTS> |
| 5.7  | <SEARCH FOR A ROOM> |
| 5.8  | <CHANGE MODULE TIMESLOT> |
| 5.9  | <E-MAIL AN ADMINISTRATOR> |

**Other Product Requirements**

**Applicable Standards**

The platform will be a windows platform; therefore will be designed to run on Internet Explorer 5.0.

**System Requirements**

The system shall be interfaced with the systems outlined in section 4 of this document.

The main component of this system will be server based and based within the SoC.

The system will require user terminals to have equivalent (or better) Pentium333 Processors, 64 MB RAM, and 10 GB hard drives.

The system requires a Windows system running Windows 95/98 or Windows NT/2000 professional. With Internet Explorer version 5.0 or better.
Performance Requirements
The system shall be able to cope with up to 50 users at any one time. However, the system will have to make sure that two departments aren’t relocating modules to the same timeslot at the same time.

Environmental Requirements
The system will run from workstation PCs therefore the environmental requirements are the same for any PC operating in an office environment.

Documentation Requirements
This section describes the documentation required by the Module Timetable system.

User Manual
The user manual would have to be distributed to every user of the system if it was built.

The user manual will be a hard paper copy describing the basic functions of the system, complete with screen shots and easy to follow Instructions. The main proposed sections include:

- Accessing the system
- Logging on/off to the site
- Searching for a module
- Select timeslot
- Compare and view Enrolment Details
- User Support Information

It is anticipated that the manual will be 50 pages long, with reference to the product features described in section 5 of this document.

Online Help
The online help function will be an on-line version of the user manual, based around the Windows help facility. It will allow topic search/selection and be updated regularly with a comprehensive troubleshooting guide.
Installation Guides, Configuration, and Read Me File

N/A as the system requires no software to be installed.

Labelling and Packaging

N/A as the system will not be marketed or packaged as it is part of Leeds University academic support network. It is expected that there be strong style and labelling conforming to University Web page design specifications.
Risk List

Introduction

Purpose
The purpose of this document is to identify and describe the major risks that are foreseen of the proposed Module Timetable System.

Scope
The document is applicable to the Module Timetable System project being developed as part of a final year project by James Halstead (IS Degree 2002).

It is a system that will integrate existing departmental SIS software with central timetable information via the World Wide Web. This new system will solve the problem of joint honour student module clashes as module administrators will be able to check whether moving or creating a module in their own department will clash with another. They will be able to select the enrolment list for a particular module and check it against other department modules from a specific timeslot.

Overview
The following pages show a table highlighting the most important risks that have been identified if the final system were to be developed.
## Risks

<table>
<thead>
<tr>
<th>Risk Ranking</th>
<th>Risk Description &amp; Impact</th>
<th>Mitigation Strategy and/or Contingency Plan</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>The interfaces between the departmental student information systems may be too complicated to construct.</td>
<td>Early exploration of departmental software and design of new sub-system if needed.</td>
</tr>
<tr>
<td>3</td>
<td>May be too few users for the system to warrant a University wide project.</td>
<td>Limit the problem to SoC and associated joint-honours in science departments, instead of University wide.</td>
</tr>
<tr>
<td>2</td>
<td>System may not run on all department computers.</td>
<td>Research specifications of computers and analyse cost of upgrading.</td>
</tr>
<tr>
<td>1</td>
<td>Resistance to change and using the new system from users. It is only effective if people use it.</td>
<td>Demonstrate and persuade the effectiveness of the new system.</td>
</tr>
<tr>
<td>0</td>
<td>Having all the University module and enrolment details on one system</td>
<td>System Security and regular backup of data would be required.</td>
</tr>
</tbody>
</table>
Business Use Case Specification: Create a new COMP Module

Brief Description
This use case describes how the business creates a SoC module.

Flow of Events
Basic Flow

1. This Use Case begins when the School Learning and Teaching Committee decide that a new module should be created.
2. They then decide on the type of module i.e. CS/IS.
3. Then they decide on title of the module, course content/syllabus.
4. Then they decide which semester it should run.
5. Then they find a lecturer to run the module.
6. Once created, the details are then passed onto the Module Administrator so the module can be timetabled the use case then ends.

Alternative Flows
None.

Special Requirements
It is assumed that the School Learning and Teaching Committee can agree upon all aspects of the module.

Pre-conditions
None

Post-conditions
None.

Extension Points
None.
Business Use Case Specification: Allocate a COMP module a new timeslot

Brief Description
This use case describes how the module administrator for the SoC allocates a module to a timeslot.

Flow of Events

Basic Flow

1. This Use Case begins when it is decided that a module should change from its current timeslot.
2. The module administrator looks at the existing timetable for the following year and fits it in where there is no clash between that module and existing COMP modules.
3. The timetable is sent to the student office for processing and the use case ends.

Alternative Flows
None.

Special Requirements
None.

Pre-conditions
If the module is a new module, the module must have been created.

Post-conditions
A module that is re-timetabled may clash with other modules external to the SoC. As there is no way of checking that those people that will be enrolled, or who are enrolled on the module will be enrolled on a joint honours scheme that has a module occupying the same timeslot.

Extension Points
None.
Business Use Case Model Survey

Introduction

Purpose
In this section of the RUP the following documentation produced will give more indication of the current practices and problem areas associated with the timetabling of COMP modules. Through clearly identify roles and describing how these business workers and entities use the business use cases.

Scope
The document is applicable to the Module Timetable System being developed as part of a final year project by James Halstead (IS Degree 2002).

Definitions, Acronyms, and Abbreviations
See Glossary.

References
See Bibliography.

Overview
The purpose of this document is to describe the business use case model in more detail.

Business Actors

SoC Learning and Teaching Committee
This actor is s group of academic staff from the SoC whose role in this system is to decide on whether a new module should be created. They keep up with what Computer graduates should be expected to do in Industry, look at what is hot and what is not and decide content for new modules.

The Central Teaching Space Timetable Information System: Student Office
This actor is actually a system that is responsible for the handling of module information for the entire University. Its main responsibility is allocating rooms based on module size and maintaining current timetable information.
Maths Department Administrator
Their main responsibility is to allocate and timetable modules for the Maths Department. This Actor is included in the Business Use Case Model to show the current situation for a different department. The problem area grows from this as some students have three joint honours departments a wide selection of electives.

SoC Department Administrator
In this system their main responsibility is to allocate and timetable modules for the SoC.

Business Use Cases
Create a new COMP module
This use case is included to show that modules are sometimes created and need timetabling, as well as the need to just re-timetable existing modules. This use case starts when the SoC Learning and Teaching Committee decide that due to Industry standards and expectations a new COMP module should be produced.

Look up current timetable information
This use case starts when a module administrator needs to know what modules are currently occupying a timeslot. Currently each department operates separately and only looks at their local Information systems or that held centrally by the student office.

Allocate a MATH module a timeslot/ Allocate a COMP module a timeslot
The problem is with these two use cases is that: If two module administrators want to allocate a module the same timeslot, they have no way of knowing whether the students who are on both those modules (or will be on that module) will both be timetabled to attend those modules at the same time.
Business Use Case Diagram

Create a new COMP module

SoC Learning and Teaching Committee

Allocate a COMP module Monday 9 o'clock

Maths Department Administrator

Allocate a MATH module Monday 9 o'clock

The Central Teaching Space Timetable Information System: Student Office

look up current timetable information

Local SIS systems

SoC Director Of Undergraduate Studies

Allocate a COMP module Monday 9 o'clock

MATH and COMP modules Monday 9 o'clock CLASH

Joint Honours Student
Business Sequence Diagram

1. Create COMP module
2. Consult timetable
3. Check for COMP clashes
4. Allocate Module a timeslot
5. Send timetable for processing
6. Allocate and book lecture theatres
7. Take COMP module
Elaboration Phase

System Sequence Diagram

- Administrators
- MT System
- Enrolment List
- Module List
- Staff/Student Details
- Lecture Theatre Booking List

- Login
- Select primary module
- View module details
- Contact a student/staff
- Search for available lecture theatres
- Get enrolment details
- Get module code
- Get module details
- Get enrolment lists
- Get list of modules in timeslots
- Get timeslots of free lecture theatres
- Compare enrolment lists
- Move module
System Activity Diagram

1. Enter username and password

2. Login

3. [logged in]

4. Select Primary Module

5. View Module Details

6. Contact students/staff

7. [before term time]

8. [no]

9. Search for appropriate Lecture Theatres

[enter username and password]

10. Select new Timeslots

11. Select Modules for comparison

12. Compare Enrolment Lists of selected modules against Primary Module

[YES]

13. Move Module

[NO]
Use Case Specification: Login

Brief Description
This use case describes how a user will log onto the web based Module Timetable System.

Flow of Events

Basic Flow

1. This use case begins when the user enters their username and password into the login screen.
2. The system validates the details and lets them progress to the main screen and the use case ends.

It is envisaged that the logon screen will look something like this:

Alternative Flows

Invalid username/password
If the user enters an invalid username or password they will not be able to log onto the system.

User forgets username/password
If the user forgets their username or password they have to contact the support department and will be reverted to their initial password.
Special Requirements
Also must have a web-based computer terminal that is capable of running the system.

Pre-conditions
The user must have been allocated a username and password and know the web address of the Module Timetable System.

Post-conditions
User is logged on
User is denied access and not logged on

Extension Points
None.
Use Case Specification: View Student/Staff Details

Brief Description
The purpose of this use case is for the user to view the details of a student/staff in case they need to be contacted. This may be to inform them of timetable clashes, or to notify students that a module is moving location to see if they have any other commitments that need to be taken into consideration before moving a selected module to a new timeslot.

Flow of Events

Basic Flow
This use case starts when the user decides that a student needs to be contacted due to timetable, or proposed timetable clashes. From the main page, the user can search for the student by Surname, First name and ID numbers for students. The system returns a list of matches, which the user can then click on the correct one to view details and then the use case ends.

Alternative Flows

System Unavailable
The system may be unable to communicate with the rest of the SIS to compare enrolment lists. Therefore an error message will be printed to the screen and the use case terminates.

Special Requirements
This use case assumes that all the student contact details for every student in the university can be obtained and are not violating any privacy policies the university may have regarding the distribution of personal information.

Pre-conditions
The user must have successfully logged onto the system via the ‘logon’ screen.

Post-conditions
The student/students can be contacted regarding changing a module to a timeslot.

Extension Points
None.
Use Case Specification: Select a Primary Module

Brief Description
The purpose of this use case is to select a University module so that its enrolment lists can be compared to that of other modules using the Module Timetable System.

Flow of Events

Basic Flow
This use case starts when the user decides that a module may need to change timeslot. They therefore need to compare the enrolment lists of a module to other modules that occupy a specific timeslot.

The user can either select the module from a list selection, grouped by department and code alphabetically. Or they can type in the University module code if known into a ‘Search’ box. The selected module is returned to the screen and this use case ends.

Alternative Flows

Module code not recognized
If the module code is not recognized, it may be that the user has mistyped the code. The code can be re-entered again and another module selected if necessary. Otherwise, if the module does not exist, see section 3 below.

System Unavailable
The system may be unable to communicate with the rest of the SIS to compare enrolment lists and the Central Teaching Space Timetable Information System. Therefore an error message will be printed to the screen and the use case terminates.

Special Requirements
The module has to be selectable from the lists and must exist in the system database; therefore if a module cannot be selected from the database, the database needs to be updated. The people who will maintain the system, most likely the student office can only do this; so the student office will have to be contacted via the ‘contact’ section of the web site, giving the module details.

Pre-conditions
The user must have successfully logged onto the system via the ‘logon’ screen.
Post-conditions

Search for Lecture Theatre
Once a module has been selected, the next use case is to search for appropriate lecture theatres and times that the module can run.

Compare Module enrolment lists
Once results have been returned by the system indicating which timeslots have available lecture theatres, the user may go on to compare enrolment lists of the selected module against the modules that occupy these timeslots to see if there are any student timetable clashes.

Extension Points
None.
Use Case Specification: View Primary Module Details

Select A Module

Brief Description
The purpose of this use case is to view the details of a selected module.

Flow of Events

Basic Flow
This use case starts when the user decides that it is necessary to view the details of a selected module.
Once a module has been selected, the user can click on the view module details button. This then brings up the next screen with the following module details:

- Module Title
- Module Code
- Timetable slot
- Place of teaching; lecture theatre or seminar room.
- Name of Lecturer
- Whether it is compulsory to any departments/subject.
- Name of Lecturer

Alternative Flows
The system may be unable to communicate with the rest of the SIS to compare enrolment lists and the Central Teaching Space Timetable Information System. Therefore an error message will be printed to the screen and the use case terminates.

Special Requirements
Also must have a web-based computer terminal that is capable of running the system.

Pre-conditions
The user must have successfully logged onto the system via the ‘logon’ screen
The user must have successfully selected a Primary module.

Post-conditions
User has successfully viewed the module details
Use Case Specification: Search for Lecture Theatres

Brief Description
This use case describes how a user can search for a timeslot that has an appropriate seminar room or lecture theatre free. This will be done going on the needs of the module and any special requirements i.e. what type of projection or computer equipment is necessary and disabilities of any enrolled students.

Flow of Events

Basic Flow
1. This use case begins when the user clicks on the search for a lecture theatre button after selecting a module.
2. The system searches for appropriate rooms based on the requirements of the module.
3. The system returns a list of lecture theatres that are free and the timeslots that they are available.

Alternative Flows

System Unavailable
The system may be unable to communicate with the rest of the SIS to compare enrolment lists and the Central Teaching Space Timetable Information System. Therefore an error message will be printed to the screen and the use case terminates.

Special Requirements
User must have a web-based computer terminal that is capable of running the system.

Pre-conditions
The user must have successfully logged onto the system via the ‘logon’ screen
The user must have successfully selected a Primary module.

Post-conditions
Compare Module enrolment lists
Once results have been returned by the system indicating which timeslots have available lecture theatres, the user may go on to compare enrolment lists of the selected module against the modules that occupy these timeslots. (See select timeslot and select modules for comparison use cases)
Use Case Specification: Select Timeslots

Brief Description
This use case describes how a user will select timeslots for which they want to compare enrolment lists.

Flow of Events

Basic Flow
1. This use case begins once the system as searched for a lecture theatre.
2. The user is presented with a timetable format screen with blocks indicating different time periods of a day. In each of these blocks is a list of appropriate lecture theatres.
3. The user can then select the timeslots from this screen by highlighting the timetable periods on the form and the use case ends.

Alternative Flows

System Unavailable
The system may be unable to communicate with the rest of the SIS to compare enrolment lists and the Central Teaching Space Timetable Information System. Therefore an error message will be printed to the screen and the use case terminates.

Special Requirements
User must have a web-based computer terminal that is capable of running the system.

Pre-conditions
The user must have successfully logged onto the system via the ‘logon’ screen
The user must have successfully selected a Primary module.
The user must have searched for lecture theatres and system returned successfully

Post-conditions
Select Modules to compare enrolment lists
Once this use case ends the next step is to select modules to compare the enrolment list to so that clashes can be determined.
Use Case Specification: Select modules for comparison

Brief Description
This use case describes how a user selects modules for comparison against the selected Primary module enrolment list.

Flow of Events

Basic Flow
1. This use case begins once user has completed the ‘select timeslots’ use case and clicks on the select module button.
2. The system now returns a list of modules ranked by module code and title that occupy the timeslots selected in the previous use case
3. The user can select which modules they wish to compare enrolment lists to by clicking on a tick box by the side of each module then the use case ends.

Alternative Flows

System Unavailable
The system may be unable to communicate with the rest of the SIS to compare enrolment lists and the Central Teaching Space Timetable Information System. Therefore an error message will be printed to the screen and the use case terminates.

Special Requirements
User must have a web-based computer terminal that is capable of running the system.

Pre-conditions

The user must have successfully logged onto the system via the ‘logon’ screen
The user must have successfully selected a Primary module.
The user must have searched for lecture theatres.
The user must have successfully selected timeslots.

Post-conditions

Compare enrolment lists
Use Case Specification: Compare enrolment lists

Brief Description
This use case describes how the system compares the enrolment lists of selected modules against that of the selected primary module.

Basic Flow
1. This use case begins once the user has clicked on ‘compare enrolment lists’ button.
2. The system now compares the enrolment list of selected modules against that of the primary selected module.
3. The system returns the search results to the screen ranked by the timeslots that have fewer clashes. I.e. if a timeslot, say 2 o’clock on a Monday has no student clashes, this will be returned rank = 1st.
4. If there are an acceptable number of clashes the user can click on the ‘move module’ button and the use case ends.

Alternative Flows
System Unavailable
The system may be unable to communicate with the rest of the SIS to compare enrolment lists. Therefore an error message will be printed to the screen and the use case terminates.

Special Requirements
User must have a web-based computer terminal that is capable of running the system.

Pre-conditions
The user must have successfully logged onto the system via the ‘logon’ screen
The user must have successfully selected a Primary module.
The user must have searched for lecture theatres.
The user must have successfully selected timeslots.
The user must have successfully selected modules for comparison.
Post-conditions

Move Module

Once the user has determined that there are an acceptable number of clashes then they can go onto the
final stage of the system: moving the primary selected module to a new location.

Use Case Specification: Move Module

Brief Description

This use case describes how the system moves a module from one timeslot to another

Flow of Events

Basic Flow

1. This use case begins once the user clicks on the ‘move module’ button.
2. The system checks the login details to see if the user is authorised to move the primary
   selected module to the new location.
3. If authorized the system notifies the Central Teaching Space Timetable Information System
   that the module will be moved and updates accordingly, the use case ends.

Alternative Flows

User Unauthorised

If the user is has selected a module outside their department, or permission scope, then the system will
not allow them to move that module to the new timeslot. An error message is output to the screen, the
module has to be from the users own department.

System Unavailable

The system may be unable to communicate with the Central Teaching Space Timetable Information
System to update the timetable information. Therefore an error message will be printed to the screen
and the use case terminates.

Special Requirements

User must have a web-based computer terminal that is capable of running the system.
Pre-conditions
The user must have successfully logged onto the system via the ‘logon’ screen.
The user must have successfully selected a Primary module.
The user must have searched for lecture theatres.
The user must have successfully selected timeslots.
The user must have successfully selected modules for comparison.
The user must have successfully compared enrolment lists.
Supplementary Specification

Introduction

Purpose

The purpose of this document is to capture the additional requirements that are not defined in the Requirements Use Cases for the MT System. Such requirements include legal and regulatory requirements, quality attributes like usability/reliability/performance, and other requirements like environmental and design constraints.

This document, combined with the Requirements Use Case Specification documentation and Requirements Use Case model capture all the requirements of the proposed system.

Scope

The document is applicable to the Module Timetable System being developed as part of a final year project by James Halstead (IS Degree 2002).

It is a system that will integrate existing departmental SIS software with central timetable information via the World Wide Web. This new system will solve the problem of joint honour student module clashes as module administrators will be able to check whether moving or creating a module in their own department will clash with another. They will be able to select the enrolment list for a particular module and check it against other department modules from a specific timeslot.

Overview

The rest of the supplementary specifications contain detailed functional and non-functional requirements that are part of the MT System.

Functionality

This section of the document includes the functional requirements that are common to more than one use case.

Authorisation Error

All attempts to access the MT system will be logged, and repeated failed attempts to log in will result in an authorisation error and the system will not allow access to anyone from that machine.
Usability

Training Time
The amount of time required for training the normal user will be reviewed after the system has reached the implementation and testing workflows of the RUP. It is anticipated that the time taken to train one person on the system will be only a few hours as it will be designed to be intuitive and user friendly.

Browser Compatibility
The system will be web based and will be usable from multiple browsers. It will initially be designed to run on the latest version of Microsoft Internet Explorer, however the system will be accessible and viewable from other browsers such as Netscape Navigator.

Reliability

Availability
The MT system can be accessed only during University Office hours. This is for security because if a system is available all the time, there is more likelihood of system integrity being compromised.

Mean Time Between Failures (MTBF) and Mean Time To Repair (MTTR)
This has not yet been decided and cannot until some working system has been developed and will therefore be discussed in the next iteration. The proposed target time for this is however is one hour.

Accuracy
The accuracy of the data in the system is of primary concern as the system is only 100 percent functional if the data is accurate to that moment of use. Therefore, it will be taken that the student office timetable data is correct and the local departmental information data is current. The system will be updated immediately if when changes have been made to the database when a module is moved.

Performance

Number of Users
The number of simultaneous users that the system will support will be as many as are needed by the University departments. The system will allow multiple users at one time module administrators for one department can only select and move modules from their specific department. However exact numbers shall be decided after an initial version of the system has been tested. It may be that the number of users will have to be limited if the system response is slow after testing.
Response Time: Search for a Lecture Theatre

It is expected that the response time will be only a few minutes to retrieve information on rooms that are available for the Primary selected module. This search is based on multiple fields for every timeslot in the University so the time taken to return list of appropriate lecture theatres may be more than a few minutes, depending on the number of simultaneous users at one instance.

Response Time: Compare Enrolment Lists

Once time slots have been selected with appropriate number of modules for comparison selected the system will have to compare enrolment lists. Again, this is a vast amount of data but the search should be quicker than 5.2 as it is only based on student ID number and not multiple fields. Again, the time taken will be more accurate after the next iteration.

Supportability

This section indicates any requirements that will enhance the supportability or maintainability of the system being built, including coding standards, naming conventions, class libraries,

Design Constraints

There are no design constraints at this time although it is proposed that the MT system will interface with a number of different systems. Therefore, middleware will have to be quite sophisticated and will most likely be written in Java.

Online User Documentation and Help System Requirements

The requirements state that there will have to be adequate help facilities available. So there will be a hard copy of a manual distributed to users and a help function with search facility on the main web page. Contact details in the event of problems of system administrators will also be given.

Purchased Components

None proposed so far.

Interfaces

Information on hardware and software interfaces has not yet been gathered but will be addressed in the next iteration.

Licensing Requirements

No licensing requirements as the system source code will be made freely available
Legal, Copyright, and Other Notices

Disclaimer

The system designers take no responsibility for loss of data or integrity of data. The system is used completely at the Universities own risk.

Applicable Standards

None known.

System Deployment Diagram
Module Timetable System Project

Main web page GUI

select primary module

User

Move a Module GUI

Search for free lecture theatre timeslots

module lists

Student Office Module Information

Compare Enrolment Details of selected module

Select timeslots and modules from list

Departmental SIS software

Enrolment lists

System Analysis Model