# MASTER OF TECHNOLOGY IN SOFTWARE ENGINEERING

# Project Requirements 2011/2012 (Part-Time)

## 1. INTRODUCTION

As part of the Master of Technology in Software Engineering (MTech SE) course, you are required to plan and carry out a software development project. The major aim of this project is for you to apply those skills and methods that you have acquired to a real life practical situation, and to design and develop a workable application system which will operate within a business organization. This project will formally begin in late January/early February 2012 and must be completed by January 2013.

The aim of this document is to describe what is required of each project team to successfully complete the MTech SE software engineering project. **Note that this information expands upon and supersedes all other information previously given to you concerning the project**.

## 2. THE PROJECT STRUCTURE

The Project work will consist of three parts: Project Planning, Analysis and Design, and Implementation. These phases are outlined below.

### 2.1 Project Planning

The aim of the project planning phase is for you to apply the project management principles you are taught in MTech SE to effectively initiate your project. The objectives of the phase are to define and scope user requirements, determine how to manage your project risks and to fully plan-out the project. The deliverables are:

- Project Plan
- Quality Plan
- User Requirement Specification
- First Project Presentation
- First Quality Audit.

### 2.2 Analysis and Design

The aim of the analysis and design phase is for you to apply the **object-oriented analysis and design techniques** you are taught in MTech SE to create a firm basis upon which to build the software. The objectives of the phase are to analyze the requirements to produce functional specifications, specify a high-level design for

the software system, and verify the design architecture through the development of a prototype. The deliverables are:

- Functional Specification
- High-level Design Specification
- Prototyping Study Report
- Second Project Presentation
- Second Quality Audit.

### 2.3 Implementation

The aim of the implementation phase is for you to apply the programming and testing techniques you are taught in MTech SE to build a software system that meets your user's requirements. The objectives of the phase are to prepare the detailed design of software (in accordance with the verified design architecture), code the software using an **object-oriented language/tool**, plan, perform and document system testing of the software, produce user documentation, and achieved user acceptance. The deliverables are:

- Detailed Design Specification
- Source and executable code (in softcopy only)
- Test Documentation (including test plan and test results)
- User Guide
- End of Project Report
- Third Project Presentation
- Third Quality Audit.

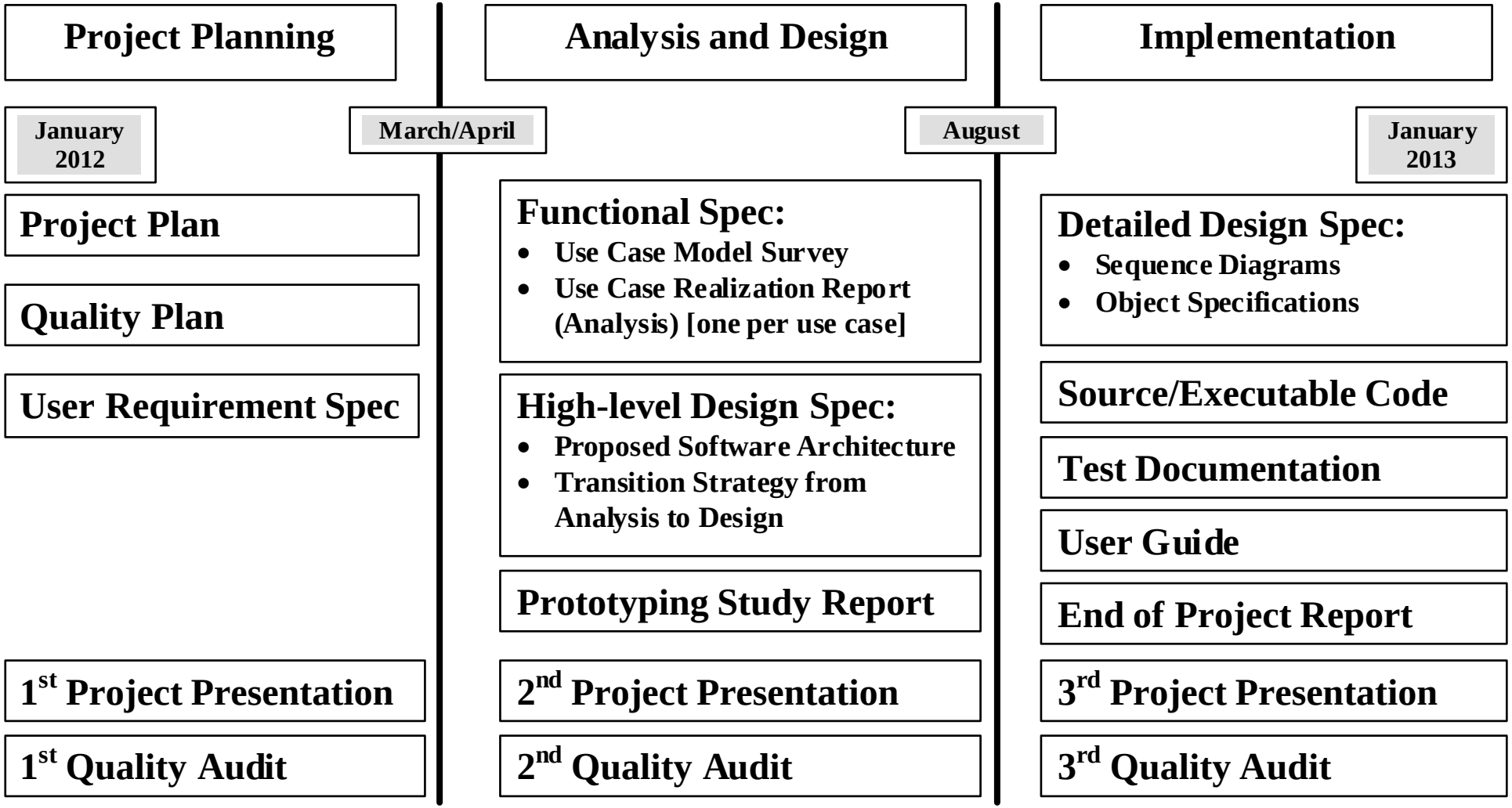
## 3. THE DELIVERABLES

The most significant characteristic of the MTech SE software engineering project is that all project teams will produce the **SAME** set of deliverables. The main requirements on each project team are as follows:

- It is mandatory for each project team to produce all the deliverables identified in section 2.
- Each deliverable **MUST** be produced to format/guideline defined in table 1 (unless special permission to use different formats is given).
- One **hardcopy** of each deliverable **MUST** be delivered to the MTech SE Internship Manager before or on the dates defined in table 1.
- One **softcopy** of each deliverable **MUST** be uploaded to the designated IVLE *workbin* before or on the dates defined in table 1.

Requirements for each deliverable are defined in more detail in table 1, and delivery schedule is summarized in figure 1.

**Figure 1: Summary Deliverables and Schedule**



## Table 1: Requirements for the Deliverables

| Deliverable | Format/Guideline | Date Required | Examiner(s) |
|---|---|---|---|
| Project Plan | Format is shown in the ***Project Planning*** module in Unit 5. A softcopy sample is provided in the **Templates** *workbin* of the **Software Engineering Project** module in the IVLE. The plan must include the following in Appendices:<br>• The justification for your estimates using ***Function Point Counting*** and ***COCOMO***, and (if necessary) other objective methods such as the other estimating techniques described in Unit 5.<br>• Management of your project's Risks (i.e.: Risk Questionnaire, Prioritized List of Risks, Risk Monitoring Techniques)<br>This plan will define your basic strategy for successfully delivering all deliverables on schedule. You will be asked to report progress against this plan during the project presentations. | 3/5 April 2012 | |
| Quality Plan | Format is shown in the ***Project Planning*** module in Unit 5. A softcopy sample is provided in the **Templates** *workbin* of the **Software Engineering Project** module in the IVLE.<br>This plan will define the methods and procedures that you MUST follow throughout the project. So be careful to define practices which you CAN follow. You will be asked to provide evidence of your compliance to this plan during the scheduled Quality Audits. | 3/5 April 2012 | Suchita |
| User Requirement Spec. | Format is shown in the ***Object Oriented Requirements and Analysis*** module of Unit 3. A softcopy sample (in slightly different but acceptable format) is provided in the **Templates** *workbin* of the **Software Engineering Project** module in the IVLE.<br>Do this as early as you can. Use it as a method to scope the project to ensure that you are not committing to an unrealistically large project. This should form the basis of all other work done on the project. | 3/5 April 2012 | |
| First Quality Audit | The project team will be expected to attend an audit meeting at which they will be expected to provide evidence that they are following the provisions of their Quality Plan. An audit checklist is given at Appendix B. **Note that the project team should prepare minutes for this meeting**. | 3/5 April 2012 | Suchita, |

**Table 1 (cont.): Requirements for the Deliverables**

| Deliverable | Format/Guideline | Date Required | Examiner(s) |
|---|---|---|---|
| First Project Presentation | The aim of the presentation is to allow the advisors to give an overall management and technical assessment of the project. The presentation should last approximately 30 minutes, followed by up to 15 minutes for question and answer. Presentation assessment guidelines are given at Appendix A.<br>The presentation should include: project background; overview of user requirements; project risks & technical challenges; plans (estimates etc) & strategies; progress report. | 10/12 April 2012 | Leonard, Yuen Kwan, Swarnalatha, Eng Lieh, Derek, Boon Kui, Suria |
| Functional Specification | As a minimum the functional specification should include the following standard RUP deliverables:<br>• Use-Case-Model Survey (including Use Case Global View)<br>• Use-Case Realization Report (Analysis) [one for each Use Case]<br>Other relevant information, such as a Data Model and the User Interface Design, may also be included in the functional specification. Note that Use-Case Realization Report (Requirements) documents **are not mandatory deliverables**.<br>The format for the above RUP deliverables is shown in the ***Object Oriented Requirements and Analysis*** module of Unit 3. These can be generated by the Rational Rose/SoDA CASE tools, or by using other suitable CASE tools in collaboration with word-processing and drawing tools. | 14/16August 2012 | Boon Kui |
| High-level Design Specification | As a minimum the high-design specification should describe the following:<br>• Proposed software architecture, describing the hardware and software platform for deployment, software distribution across the architecture (e.g.: client/server considerations), etc.<br>• Transition strategy from analysis to design, showing how language specific, platform specific and other considerations will be taken into account.<br>Other relevant information, such as a Data Base Design, may also be included in the high-level design specification.<br>No specific format is required. | 14/16August 2012 | Boon Kui |

**Table 1 (cont.): Requirements for the Deliverables**

| Deliverable | Format/Guideline | Date Required | Examiner(s) |
|---|---|---|---|
| Prototyping Study Report | The project team are required to undertake prototyping of parts of the required software to explore and verify the proposed software architecture and transition strategy. A report should be written to document the finding from this study. No specific format is required. You will also be required to give a demonstration of the prototype as part of your Second Project Presentation. | 14/16August 2012 | Boon Kui |
| Second Quality Audit | The project team will be expected to attend an audit meeting at which they will be expected to provide evidence that they are following the provisions of their Quality Plan. An audit checklist is given at Appendix B. In particular, the project team will be required to demonstrate that they have carried out all required actions from the First Quality Audit. **Note that the project team should prepare minutes for this meeting**. | 14/16August 2012 | Suchita, |
| Second Project Presentation | The aim of the presentation is to allow the advisors to give an overall management and technical assessment of the project. The presentation should last approximately 30 minutes, followed by up to 15 minutes for question and answer. Presentation assessment guidelines are given at Appendix A.<br>The presentation should include: analysis of requirements; software architecture; transition from analysis to design, implementation issues; progress report against the plans *(planned versus actuals*). | 21/23 August 2012 | Leonard, Yuen Kwan, Swarnalatha, Eng Lieh, Derek, Boon Kui, Suria |
| Detailed Design Specification | As a minimum the detailed design specification should be documented by the standard RUP Design Model Report containing at least the following:<br>• Sequence Diagrams (at least one for each Use Case)<br>• Object Specifications (describing each Design Object)<br>Other relevant information, such as any changes to the high-level design, may also be included in the detailed design specification.<br>The format for the RUP Design Model Report is shown in the ***Object Oriented Requirements and Analysis*** module of Unit 3. This can be generated by the Rational Rose/SoDA CASE tools, or by using other suitable CASE tools in collaboration with word-processing and drawing tools. | 8/10 January 2013 | Boon Kui |

**Table 1 (cont.): Requirements for the Deliverables**

| Deliverable | Format/Guideline | Date Required | Examiner(s) |
|---|---|---|---|
| Source and Executable Code | The source code (written using an object-oriented language/tool) should be delivered on magnetic media. You will also be required to give a demonstration of the running system to facilitate grading. | 8/10 January 2013 | Swarnalatha, Derek, |
| Test Documentation | Format is shown in the ***Software Testing*** module in Unit 7. A softcopy sample is provided in the **Templates** *workbin* of the **Software Engineering Project** module in the IVLE.<br>The test documentation should include the test plan and the test results. | 8/10 January 2013 | Suchita |
| User Guide | No format specific format is required. However, it should be noted that an example is given in softcopy in the **Templates** *workbin* of the **Software Engineering Project** module in the IVLE. The delivery medium may be chosen as appropriate for the project (e.g.: hardcopy manual, help file, Web Page etc) | 8/10 January 2013 | |
| End of Project Report | Format and guidelines on preparing the report are given in appendix D. | 8/10 January 2013 | |
| Third Quality Audit | The project team will be expected to attend an audit meeting at which they will be expected to provide evidence that they are following the provisions of their Quality Plan. An audit checklist is given at Appendix B. In particular, the project team will be required to demonstrate that they have carried out all required actions from the Second Quality Audit. | 8/10 January 2013 | Suchita, |

| Deliverable | Format/Guideline | Date Required | Examiner(s) |
|---|---|---|---|
| Third Project Presentation | The aim of the presentation is to allow the advisors to give an overall management and technical assessment of the project. The presentation should last approximately 30 minutes followed by up to 15 minutes for question and answer. Presentation assessment guidelines are given at Appendix A.<br><br>This is the final presentation and should report on the whole project. It should include: overview of background and requirements; summary development strategy and technical issues; brief demonstration (if possible); report on the overall project progress (*planned versus actual*); recommendations for further work; lessons learnt by the project team. | 15/17 January 2013 | Leonard, Yuen Kwan, Eng Lieh Boon Kui, Suria |
| Project Demonstration | The Project team will give a thorough demonstration of their completed software system to a group of ISS lecturers, and also allow the lecturers to have a "hands-on" session with the completed software. They will also participate in a code walkthrough of their completed software with the lecturers | 15/17 January 2013 | Swarna, Derek |

# 4. RESOURCES

Each project is encouraged to use the computer hardware/software resources available at ISS. These are defined below.

## 4.1 Computer Hardware

The PCs in the discussion rooms at ISS are available for you to use to undertake your development work. In addition, each team will be provided with access to a Windows 2003 server to act as their web-server, application server and/or database server. You should contact the system administrator, Mr Jasper Tan, at 6516-8552 (e-mail: isstz @nus.edu.sg) to request for access to such a server.

## 4.2 Collaboration Facilities

To facilitate collaborative work, a directory of the MTech drive of the ISS LAN (drive S:\) should be created by each team using the following directory name:

s:\se18_t'n'\

where 'n' = the team number (e.g. '1ft' indicates team 1 of the full-time batch).

It should be noted that by default the MTech drive is accessible to all ISS students and staff. Therefore files are stored on this server at each team's own risk. Hence, if you would like to create **secure** directories on drive s:\ that can only be accessed by your team (i.e.: only your team would have ***write-access***) then you should contact the ISS IT helpdesk, at 6516-2066 (e-mail: issithelpdesk@nus.edu.sg). You should inform them of your batch (i.e.: SE18), your team number (egg: Team 1 Full-time) and the members of your team.

To further support collaboration, a Version Management System (VMS) is available for MTech students to use. The VMS can help project teams to manage versions of project documents and source code. It is basically a CVS repository which students can access through the Internet.

Softcopies of the VMS user manual and registration form are available on the ISS LAN in directory **L:\ymzeng\vms**, or they can be downloaded from http://www.iss.nus.edu.sg/iss/article_display.jsp?artid=661 *(total size of the softcopy of the manual is about 1.5MB)*.

Projects teams that are interested in using VMS should fill-in the registration form and submit it via email to the ISS IT helpdesk at issithelpdesk@nus.edu.sg. If students have any inquiries concerning VMS, then they should contact the ISS IT helpdesk, at 6516-2066 (e-mail: issithelpdesk@nus.edu.sg)

In addition, the **Software Engineering Project** module in the IVLE has been set-up to enable project teams to store and share their project files.

### 4.3 Programming Languages and Development Tools

ISS will make available, in each discussion room, several programming tools for use during the Implementation Phase or for prototyping in earlier phases. The following are provided:

1. Microsoft Visual C++.
2. Microsoft Visual C#.
3. Java Development Kit and Development Environments.

If these software are not installed on the PC you want to use in a breakout room, please contact the ISS IT helpdesk, at 6516-2066 (e-mail: issithelpdesk@nus.edu.sg) to find out the installation instructions.

Other Java programming tools and development environments may be made available for your use. To obtain more information on the tools that are available and the installation instructions, please contact the ISS IT helpdesk, at 6516-2066 (e-mail: issithelpdesk@nus.edu.sg)

To facilitate the development of client/server applications, Windows 2003 Servers are available. Software such as a Web server, an ORACLE database server and an SQL Server database server can be installed on request. To obtain more information on the Windows 2003 Servers and how you can use them in your project work, contact the ISS IT helpdesk, at 6516-2066 (e-mail: issithelpdesk@nus.edu.sg)

To facilitate the development of distributed object oriented applications, Microsoft .NET and Java 2 Enterprise Edition (J2EE) software are available. To develop J2EE applications, the following software may be installed: J2EE Reference Implementation, Orion Application Server, Apache Web Server, Apache Tomcat and Cocoon XML Engine – others will be considered on request. To obtain more information on the Microsoft .NET software and how you can use it in your project work, please contact Dr Derek Kiong at 6516-6771 (e-mail: dkiong@nus.edu.sg). To obtain more information on the J2EE software and how it can be installed, please contact the ISS IT helpdesk, at 6516-2066 (e-mail: issithelpdesk@nus.edu.sg)

### 4.4 Other Software

The ***Rational Suite Enterprise*** software is available in the discussion rooms. Its primary use would be to facilitate object-oriented analysis and design tasks. However, the suite also provides code generation (in most object-oriented languages/ environments), automated testing and reverse engineering of source code.

The ***MSOffice*** software is available in the breakout room, which should be useful for producing documents and presentations: ***Word*** (word processor), ***Excel*** (spreadsheet) and ***Powerpoint*** (presentations tool).

To facilitate the Internet software development, ***Netscape*** and ***Internet Explorer*** are available in the breakout rooms.

In addition, the ***Microsoft Project*** software is available in the breakout rooms to assist you in project planning and tracking.

If these software are not installed on the PC you are using in a discussion room, please contact the system administrator, ISS Help Desk at 6516-2066 (e-mail: issithelpdesk@nus.edu.sg) to find out the installation instructions.

### 4.5 Sample Documents

Sample documents are available in softcopy, which can be used as the basis for several of the required project deliverables. They are available in the **Templates** *workbin* of the **Software Engineering Project** module of the IVLE.

The contents of the workbin includes the following sample documents taken from phase 2 of the ISS FCS project:

| | |
|---|---|
| **fcs2prg1.doc** | a programmer's manual. |
| **fcs2proj.doc** | a project plan. |
| **fcs2qapl.doc** | a quality plan. |
| **fcs2spec.doc** | a system specification. |
| **fcs2tpln.doc** | a system test plan. |
| **fcs2urs.doc** | a user requirement specification. |
| **fcs2user.doc** | a user's guide. |

## 5. ASSESSMENT

The project deliverables **MUST** be delivered to ISS by the dates specified in table 1. The deliverables will be graded by the examiners listed in table 1. Assessment Guidelines are given in Appendix C. These grades will be combined together to give the final grading for the whole project. **You must obtain at least a pass grade on the project to be awarded the MTech degree**.

As part of our assessment of your project work, credit will be given for the individual contributions made by each team member. To achieve this, all students will be required to assess the contribution made to the technical and managerial work of their project by each of the members of their team. These assessments should be made by each student at the end of each phase of the project by using the Team Member Peer Assessment Form given at Appendix F. Using these assessments, each student's grade will be calculated by adding together their combined team mark (based on the combined assessment of the project deliverables) and their individual mark (based on the peer assessments).

## 6. PROGRESS REPORTING AND MONITORING

To monitor the progress of your project, each project team MUST submit a progress report by the first Monday of each month. **The first progress report is due by Monday 7 March 2012**. The report should be brief. It is recommended that you submit your reports using the progress reporting form given in Appendix E (a

softcopy is available in the **Templates** *workbin* of the **Software Engineering Project** module of the IVLE in file **prog_rep.doc**). You are required to submit your report in **softcopy only** by uploading it to your team's folder in the **Progress Reports** *workbin* of the **Software Engineering Project** module of the IVLE.

## 7. CHANGE MANAGEMENT

Each project will be required to define its own scope in a User Requirements Specification. When this specification is signed-off, it will become the agreed scope for the project.

In addition, each project is required to define procedures to control requirements change in its Quality Plan.

Changes to the requirements of the project (either increasing or decreasing) and hence to the project scope are permitted provided:

1. the changes are agreed to by the customer (if a customer exists);
2. the changes are agreed to by ISS. To achieve this, the MTech SE Internship Manager should be formally notified in writing of any such change requests; and
3. the change procedure in the Quality Plan is followed correctly.

In general, ISS will agree to changes in scope provided that a suitable reason for the change is provided and that the scope remains acceptable (i.e.: there is sufficient work for the number of team members). However, as a result of changes being agreed it should be noted that:

1. The Requirement Specification should be modified and reissued if the change effects the user requirements as defined in the approved specification.
2. The Design Specifications should be modified and reissued if the change effects the software design as defined in the approved specification.
3. The Test Plan should be modified and reissued if the change effects the test cases, test procedures, test data etc as defined in the approved plan

## 8. RULES AND REGULATIONS

The following *official* ISS rules and regulations concerning the MTech SE Project **MUST** be complied with by all candidates:

1. All MTech candidates are required to successfully complete a project. For SE students, this is always team based. The size of each team should be between 4 and 8 members (inclusive).
2. Part-time candidates are normally expected to undertake their project over the period January – December of their second year of study.
3. Full-time candidates are normally expected to undertake their project over the period September – March of their second and third semesters of study.

4. The aim of the MTech projects is to enable the students to put into practice the knowledge they acquire during the course. Hence, to ensure that this aim is fulfilled, candidates are expected to deliver a pre-defined set of deliverables according to a pre-defined schedule spread throughout the one year project duration.

5. An extension to any of the deadlines in the pre-defined schedule of up to one month can be granted provided that extenuating circumstances can be presented that would justify such an extension.

6. Due to heavy work or family commitments, part-time candidates may not have sufficient available time to commit to undertake their project during their second year of study. In such circumstances, candidates may request to defer their project until their third or fourth year. Such requests should be made in writing to the MTech Programme Manager prior to the commencement date of the project. However, it should be noted that:

   - The maximum period of candidacy is 5 years.
   - Candidate can not commence their Advanced Electives until they have successfully completed their project.
   - Candidates are required to pay full annual tuition fees during a year in which they only do project work.

7. The projects will be assessed by combining the assessment of all the deliverables. It should be noted that if individual deliverables are found to be unacceptable then candidates may be requested to undertake additional work to achieve the required standard. If necessary, an extension to the overall project duration will be granted to facilitate the required additional work. However, the maximum extension in such circumstances would normally be one year. The conditions in para (5) would also apply.

# Appendix A:
# Project Presentation Assessment Guidelines

The MTech SE Software Engineering Project requires each project team to undertake three formal project presentations. The aim of each presentation is to allow the advisors to give an overall management and technical assessment of the project. Each presentation should be marked by the Team Advisors using a ***Project Presentation Assessment Form***. The assessment should be carried out by giving marks against the four categories provided on the form. A **Maximum Mark** for each category is stated. The assessor should enter a mark less than or equal to the maximum in column titled **Evaluation**. If possible, comments should be provided to justify the assessment. Some guidance on how to interpret the eleven assessment categories is given in the following sections.

## 1. Management Assessment

a. Has the team demonstrated that they have adequately managed the project during the last project phase?
b. Have project risks been adequately managed?
c. Has the project followed its planned schedule?
d. Have problems concerning effort and schedule over-runs been adequately resolved?
e. Have problems concerning the availability of team members been adequately resolved?
f. Has the project achieved its current objectives?
g. Has the team satisfactorily applied the management techniques taught during the course?

## 2. Technical Assessment

a. Is the scope and content of the project sufficiently technically demanding?
b. Has a wide variety of software engineering techniques been applied to the project?
c. Has the team adopted an appropriate technical strategy?
d. Has the team found adequate solutions to technical problems?
e. Has the team evaluated suitable alternatives to the technical solutions chosen?
f. Does the team appear to understand the full implications of their technical choices?
g. Has the team satisfactorily applied object oriented technology to produce their solution?

h. Is the solution technically correct (i.e.: are there obvious bugs, design errors etc)?

## 3. Presentation Assessment

a. Was there a clear introduction to the presentation?
b. Was the presentation well organized and sequenced?
c. Was the material presented relevant to the aim and objectives of the presentation?
d. Were effective visual aids used to enable the speaker to illustrate his case or make a visual statement that is significant to the presentation?
e. Did the speakers appear interested in and enthusiastic about the topic presented and eager to influence/captivate the audience?
f. Did the speakers appear composed and confident during the presentation, able to handle difficult questions or a difficult audience?
g. Were the speakers able to convince the audience that the team have done a good job during the project?

## 4. Added Value

a. Have the project team provided only the minimum required to be compliant with the requirements of the MTech course?
b. Have the project team made attempts to provide relevant additional information?
c. Are imaginative, creative or technically advanced methods, tools, practices or solutions being employed? The project team should be given credit for undertaking work which goes beyond the basic requirements.
d. Has a sufficiently flexible solution been selected so that the system is maintainable and extendible?
e. Has the team shown willingness to explore new technologies and learn new skills, in the face of increased difficulty and uncertainty?

# Appendix B:
# Project Quality Audit Checklist

## 1. Review of Actions from Previous Audit

a. Are all corrective actions from previous audit formally resolved?

b. **Note:** in the first audit, at this time feedback may be given on the QA Plan.

## 2. Project Organization

a. Is there evidence that respective team members understand their roles and responsibilities?

b. Is there evidence of delegation of work? Look for Work Instruction Forms (or similar, such as work delegation in meeting minutes).

## 3. Documentation Organization

a. Is the team able to demonstrate their project filing system? Look for the MFD and File Contents.

b. Are all documents adequately referenced and indexed? In particular, do all documents conform with the standard referencing system as stated in the project's QA Plan?

## 4. Planning and Progress Control

a. Is there adequate evidence of project planning? In particular, is there evidence that plans are updated, and that re-planning takes place when necessary?

b. Is there evidence of progress reporting and monitoring?

c. Is there evidence of progress review meetings? Look for meeting minutes.

## 5. Quality Control

a. Is there evidence that the scheduled reviews actually take place?

b. Is there adequate review evidence? (i.e.: are the review findings adequately documented)

c. Is there evidence that corrective actions raised during reviews are followed-up?

d. Is there adequate evidence of software testing? (i.e.: planning, documentation of results and follow-up)

## 6. Document and Requirements Control

a. Are all controlled documents uniquely identified, formally reviewed and signed-off?

b. Is there evidence that the change control procedure has been followed?

# Appendix C:
# Project Deliverable Assessment Guidelines

The MTech SE Software Engineering Project requires each project team to produce a mandatory set of project deliverables. Each deliverable should be marked using a ***Project Deliverable Assessment Form***. The assessment should be carried out by giving marks against the five categories provided on the form. A **Maximum Mark** for each category is stated. The assessor should enter a mark less than or equal to the maximum in column titled **Evaluation**. If possible, comments should be provided to justify the assessment. Some guidance on how to interpret the five assessment categories is given in the following sections.

## 1. Conformance to Standard

a. Does the deliverable meet the Format/Guideline stated in the document ***Master of Technology in Software Engineering: Project Requirements?***
b. Have all relevant Quality Procedures as stated in the project's quality plan (such as document approval, change management, document referencing etc) been correctly followed?

## 2. Technical Content

a. Does the deliverable serve its purpose in the context of the project? For instance, is it a good plan or specification?
b. Does the deliverable demonstrate the required level of technical capability?
c. Have the methods been correctly applied?
d. Has an appropriately elegant solution been selected?

## 3. Accuracy

a. Are the facts accurately presented?
b. Are there any faults in perception, logic or understanding?
c. Are there any technical faults, such as programming errors?
d. Is the deliverable complete, or are there significant omissions?
e. Is the English used grammatically correct?
f. Are there any spelling errors?
g. Is the deliverable internally consistent?
h. Is the deliverable consistent with other project deliverables?

## 4. Clarity of Expression

a. Are the ideas in the written text clearly expressed?
b. Is a good, uniform and consistent writing style employed?
c. Is the tone used appropriate for a technical report?
d. Are visual aids used effectively?
e. Are diagrams well conceived and neatly drawn?
f. In programming, are appropriate naming conventions and code layout/format standards used?

## 5. Added Value

a. Have the project team provided only the minimum required to be compliant with the standard?
b. Have the project team made attempts to provide relevant additional information? For instance, in the Functional Spec or Design Spec, has the team provided a user interface design and/or a database design?
c. Are imaginative, creative or technically advanced methods, tools, practices or solutions being employed? The project team should be given credit for undertaking work which goes beyond the basic requirements.
c. Has a sufficiently flexible solution been selected so that the system is maintainable and extendable?

# Appendix D:
# Guide to Producing End of Project Reports

## 1. Introduction

Your software development project is the most important single piece of work in your degree programme. It provides the opportunity for you to plan and organize a large project over a long period, and to put into practice the techniques you have been taught throughout the course. The project report is a deliverable that will accompany the final product you have developed, and communicates what is important about your project, and how well you have accomplished the objectives. This document gives some guidelines on structuring and planning your final project report.

## 2. Project Report Objectives

The project report is an extremely important aspect of the project. It serves to show what you have achieved and should demonstrate that:

- You understand the wider context of computing by relating your choice of project, and the approach you take, to best practices, industry needs, existing products and platforms, or current trends.
- You can apply the techniques taught in the course to the problem you are addressing and that you understand their relevance to the wider world of software engineering.
- As a computing professional, you can explain your thinking and working processes clearly and concisely to third parties who may not be experts in the field in which you are working.
- You are capable of objectively criticizing your own work and making constructive suggestions for improvements or further work based on your experiences so far.

The report is intended for the project assessors, but you should think of it as an academic publication: indeed, in some cases, your project report may be published externally, and your report will be the basis for the publication. Therefore, you will not assume that the readers have followed the project throughout, and the writing should be both clear (for readers unfamiliar with your work) and concise (for busy readers).

From your point of view, it is important that the project can impress assessors, as it will form a final impression of the goodness of the project. Software engineering is also about communicating technical information; do not underestimate the importance of the report, as you might be graded below your potential because of an indifferent or poor write-up. Ideally you should

produce your report as you go along and use the last week or two to bring it together into a coherent document.

The physical layout and formatting of the report is important. A tidy, well laid-out and consistently formatted document is easier to read and suggestive of a professional attitude. There are no requirements as to the tool used to produce the report, anything you are familiar with (e.g. MS Word or LaTeX) can be used.

Report may vary in size, but generally you should produce between 4 and 10 pages of 12-point text, excluding figures, tables, and front and back matter. Quantity does not automatically guarantee quality, and a 10-page report is not necessarily twice as good as a 5-page one. Conciseness, clarity and elegance are invaluable qualities in report writing, and will be rewarded appropriately. However, you should not leave out important information just for the sake of brevity, and your report should contain the elements detailed below.

Some projects, particularly more research-oriented ones from staff, may require larger and more detailed reports. If this is the case, the customer will have to compromise on user documentation (in other words, you will produce more report and less user documentation). This trade-off will be mutually agreed between the team and the customer, and documented in project meeting minutes.

# 3. Report Content

## 3.1 Front Matter

### 3.1.1 Title page

You should include:

- The project title
- The name of the customer
- The name and matriculation number of all members of the development team
- The name of the team's team advisor
- The abstract

### 3.1.2 Abstract

The abstract is a very brief summary of the report's contents. It should be no more than half a page long. Somebody unfamiliar with your project should have a good idea of what it's about having read the abstract alone, and will know whether it is of interest to them.

### 3.1.3 Table of Contents page

This should list the main chapters and (sub) sections of your report. Choose self-explanatory chapter and section titles, with page. Avoid too many levels of subheading.

## 3.2 Body

### 3.2.1 Introduction

This is one of the most important components of the report. It should begin with a clear statement of what the project is about, so that a lay reader can understand the nature and scope of the project. The introduction should summarize everything you set out to achieve, provide a clear summary of the project's background, relevance and main contributions, including high-level statements about the approaches and technologies that were used, and any notable achievements.

Read the introduction should set the scene for the project and provide the reader with a summary of the key features of the remainder of the report. The introduction itself should be largely non-technical, and should not list low-level objectives. Concentrate on the big issues (functional, architectural, scientific or otherwise) that the project sets out to accomplish, and the key achievements.

### 3.2.2 Background

The background section of the report should set the project into context by providing

- Domain information, to support the average reader if they are likely to be unfamiliar with the problem space.
- The rationale of key choices made, such as architectural and technological choices, and any particular feature of your solution

Domain information should be sufficient to provide a context for the reader to understand the remainder of the document, and be able to understand how the needs were analyzed.

Information about the rationale of your choices is important to demonstrate the decision-making capabilities of your team, and your awareness of potential choices. There are usually many ways of solving a given problem, and you shouldn't just pick one at random. You should explain why alternative approaches were not chosen. Especially for more research-oriented reports, you may need to reference published work, such as research papers, articles and textbooks, which you based your decisions on. When referring to other pieces of work, cite the sources where they are referred to or used, rather than just listing them at the end.

### 3.2.3 Body of report

The central part of the report usually details the technical work undertaken during the project, and may be further divided into a small number of sections. The structure of these chapters is highly project dependent. They can reflect the chronological development of the project, e.g. requirements, analysis, design, implementation etc. although this is not *always* the best approach. An alternative is to cover different parts of your systems in different sections.

You should describe and justify key features, choices, design solutions, approaches used by your product at some high level, and you are advised to use figures and graphs to support your explanations. It should also document any interesting problems with your implementation, and challenges that required significant work to resolve. Integration and testing are also important to discuss in some cases.

Management highlights may also be included, but only where they add value to your report. This is not a status report: detailed issues, timescales, slippages, manpower issues and so on are assessed through other cannels. However, a summary of project metrics, and any interesting SE management approaches adopted by the project, should be provided.

### 3.2.4 Evaluation

You should include a section that evaluates the outcome of your project. Your testing outcomes are part of the evaluation (again, no tedious details should be provided, although informative summaries are good), but assessors will also be looking for an evaluation of the strengths and weaknesses of what you have done. You may provide support for your evaluation, such as quantitative results (e.g. numerical results, performance measurements) or a more qualitative. You should be concise and honest, focusing on important and relevant aspects of your system. Do not make hollow or unsubstantiated claims; provide a critical appraisal of what you have done, showing that you also understand the limitations of your.

### 3.2.5 Conclusions and Future Work

The project's conclusions should summarize the outcome of your project, and describe what you have demonstrated. Statements on the final status of the system, its fulfillment of requirements, and important lessons learnt or discoveries can be included. Avoid tedious personal reflections like "I learned a lot about Java programming...", or "Part time studying puts a strain on teamwork…". You should finish the report by listing realistic ways in which the project can be taken further, particularly if you can show that you understand the true potential of the product you have created.

## 3.3 Back Matter

### 3.3.1 Acknowledgements

You may briefly thank those individuals who have provided particularly useful assistance, technical or otherwise, during your project.

### 3.3.2 Bibliography

This consists of a list of books, articles, manuals etc. used in the project and referred to in the report. It need not necessarily be extensive, just list the resources that make sense. You should provide enough information to allow the reader to find the source, for example:

[1] Bennett, A.J., Field, A.J. and Harrison, P.G., "Modeling and Validation of Shared Memory Coherency Protocols", Performance Evaluation, 1996, Vol. 27 & 28, 1996, pp. 541-562.

### 3.3.3 Appendix

Appendices are used to attach information that is peripheral to the main body of the report. Information typically included are things like parts of the code, tables, proofs, test cases or any other material which would break up the theme of the text if it appeared in situ. You are not required to provide appendices, and you should not overdo it.

You definitely should NOT include extensive code listings, user guides, detailed analysis and design documentation, and so on.

# Appendix E : Project Progress Report Form

| **Team Number:** | **Date:** | |
|---|---|---|
| **Summary of work done since last report:** | | |
| **Deliverables Started since last report:** | **Deliverables Complete since last report:** | |
| **Effort Expended** | | |
| **Name of each Team Member** | **Hours** | **Brief Summary of Work** |
| | | |
| | | |
| | | |
| | | |
| | | |

**Problems Encountered** *(highlight any assistance you might need from your ISS advisor or other staff)*:

**Plans for the Next Month:**

# Appendix F : Team Member Peer Assessment Form

| | |
|---|---|
| **Your Name:** | |
| **Your Matriculation Number:** | |
| **Your Project Team Number:** | |
| **Saturday or Evening Team?:** | |
| **Project Phase:** | |

| Rating | Performance |
|---|---|
| 0-3 | Poor |
| 4-6 | Average |
| 7-9 | Good |
| 10 | Excellent |

Assess the contribution of each of the members of your team (not including yourself) in terms of the contribution they made to the technical and managerial tasks of the project. Give a rating out of 10 for each of: **Technical Work** and **Management Work**, for each team member, using the rating scale above

| S/N | **Name of Team Member** *(enter the name of each team member)* | **Contribution to Technical Work** *(give rating out of 10)* | **Contribution to Management Work** *(give rating out of 10)* | **If you rated a Team Member above 7 or below 5 in either category then you MUST explain your rating below** |
|---|---|---|---|---|
| 1. | | | | |
| 2. | | | | |
| 3. | | | | |
| 4. | | | | |
| 5. | | | | |
| 6. | | | | |
| 7. | | | | |