# ICT Project A & B

# Project Manual

Nicole Herbert
Kristy de Salas

Version 4
January 2012

The purpose of the ICT Project Manual is to provide guidance to the students enrolled in KXX331 & KXX332 ICT Project A & B at the University of Tasmania during 2012.

Queries: Nicole.Herbert@utas.edu.au

*Introduction* ..... 2
*Learning Outcomes* ..... 3
*Generic Attributes* ..... 3
*Projects* ..... 5
*Team Management* ..... 6
*Lecturer Interaction* ..... 10
*Client Interaction* ..... 11
*Project Reports* ..... 15
*Documentation* ..... 24
*Web Page* ..... 26
*Presentation* ..... 27
*Demonstration Day* ..... 28
*Timesheets* ..... 30
*Task Schedule* ..... 32
*Assessment* ..... 33
*Unit Schedule* ..... 37
*Resources* ..... 39
*Appendix A – Timesheet* ..... 40
*Appendix B – Confidentiality agreement template* ..... 41
*Appendix C – Intellectual Property Agreement* ..... 43

# Introduction

ICT Project provides students with the experience of developing a medium-scale software project in a small team. All aspects of the development lifecycle will be considered: problem specification, requirement extraction, system design, implementation, testing, documentation and integration. The units provide students with the experience of working in a team and dealing with the associated challenges of communication and team management.

In the first two years of your degree practical experience is limited to artificial assignments. ICT Project allows you to apply your theoretical knowledge on a real-world project so that you will be able to transfer to industry easily.

ICT project is broken into two units, KXX331 Project A and KXX332 Project B, which must be completed over consecutive semesters. Students work on the same project in both units, unless circumstances prevent this.

Each student is placed into a project team of approximately 7 students; you will be involved in forming your own teams. Team size may vary due to class or project size. Each team will be given a project from the list available, the allocation will be based on student preferences. Each project has a real client. In most cases the client is not a member of the University community. All clients need to be treated professionally.

In Project A you complete release 1 (or a third of the project), you complete the remaining two thirds in Project B. Students are discouraged from working on the project over the semester break as you should be doing exams in other units and you need a rest.

In both units each student will get an individual grade. This grade will be made up of a team component and an individual component. To pass each unit a student must get at least 45% of the marks allocated for each component and greater than 50% overall. Please see the assessment section for more information.

Each team member is responsible for:

- A professional approach to the project and to the other members of the team;
- Doing the tasks allocated to them at the team meetings by the specified date;
- Keeping all appointments with clients, lecturer and team members;
- Contributing at team discussions and hence increasing team intellectual property.

**Good communication between everyone leads to success!**

# Learning Outcomes

On completion of this unit, you should be able to:

1. in development of a substantial ICT system, demonstrate foundational ICT knowledge of:
    - programming, algorithms & data structures
    - systems and applications
    - historical and current trends
2. apply knowledge of computing, business and project management principles and technical skills to guide the development and implementation of a substantial ICT system by:
    - interpreting and critically analyzing an organizations needs
    - using systems analysis techniques and tools to analyse and model business processes & information requirements
    - using abstraction and computational thinking
    - evaluating strengths and weaknesses of potential solutions
    - creating artifacts using a variety of templates, techniques and tools
    - following a recognised software development methodology
    - adapting existing and emerging computing technologies.
3. act professionally by:
    - communicating in different modes to diverse audiences
    - adhering to professional and ethical codes of conduct
    - working independently and collaborating in diverse teams
    - considering economic, social, legal, and ethical consequences

# Generic Attributes

*Knowledge*

- Students will be able to apply previous project management, systems analysis, and software development knowledge and independently learn new skills to build a software system according to client requirements and deadlines;
- Students will be able to investigate and overcome issues and challenges associated with constructing a substantial piece of software;
- Students will develop research skills to identify and use appropriate systems design and development tools and other resources;
- Students will be able to apply technical and information skills appropriate to the practice of project management in the ICT industry;
- Students will develop a broad knowledge base in the application of project management principles;

*Communication Skills*

- Students will develop the ability communicate effectively with a real world client, in particular to extract requirements from a client, analyse and organise the information and formulate ideas to provide a software solution;
- Students will demonstrate strong oral and written skills through effective teamwork situations, be able to organise and present information in well structured user and technical documents and through effective verbal communication using communication technologies as appropriate;

*Problem-solving*

- Students will develop effective problem-solving skills, be able to conceptualize problems and be able to find, acquire, evaluate and manage and use relevant information in a range of media to formulate a range of solutions to a non-trivial software project;
- Students will have ability to interact effectively with others in order to work towards a common outcome;

*Global Perspective*

- Students will be able to demonstrate mastery of skills appropriate to professional practice in preparation for the transition to an IT working environment;
- Students will be able to recognise the critical importance of the field of project management in the development of software systems;
- Students will have ability to interact with members of the Tasmanian IT industry;
- Students will be able to function in a multicultural or global context as effective project management skills are transferable;

*Social Responsibility*

- Students need to be able to acknowledge the social and ethical implications of their actions and appreciate the impact of social change on organisations and individuals where new technologies are implemented.

# Projects

Projects have different benefits to students. Students may want to take on a project that is relevant to an area they would like to work in after leaving university, as these projects should provide very relevant experience. Some projects are very client interaction based, others are graphical, and others are rather technical, while others involve displaying online content.

A complete description of available projects is in a separate document. Teams are formed based on student's preferences for a particular project or project type. Every student will have to nominate more than one project that they would like to do. Nomination forms will be handed out in the first lecture.

After you have had your initial meeting with your client you may be allowed to change project to another untaken one if a major problem exists. This will not be granted lightly. Change will only be allowed if the lecturer agrees there are insurmountable problems.

The list of projects is at http://www.cis.utas.edu.au/units/kxx331/projects2012/.

### *Intellectual Property*

It may be a condition of your project if it is sponsored by industry that you assign to the client, any intellectual property rights in relation to the project results. If this is the case and you wish to work on the project, you will need to sign an assignment agreement to give effect to this requirement. The example agreement is attached in Appendix C.

# Team Management

Lecturers will form the teams, but you can indicate some of your preferences for team members. Teams will have approximately 7 members, though this may vary due to the class size or project size. Teams will be formed so that there are people who have strengths in one or more of the following areas: analysis, design, programming, documentation and communication. The University of Tasmania encourages students to work with students of different nationalities to promote intercultural experiences.

## *Team Roles*

Each person in the team should be given a management and/or coordination role. You should rotate roles each semester so that you each get different experiences or if you find some team members are not performing well to the detriment of the team (you can rotate roles during semester if you need to).

### *Management Roles*

**Client liaison:** This person provides easy access to the team for the client. This person does the bulk of the communication with the client, eg organising the meetings, keeping them up to date on progress. This person should attend most client meetings. Note: All team members can attend meetings with the client.

**Project manager:** A project manager manages the team, sets the agenda for and controls the meetings, and liaises between all stakeholders. Their main role is managing task allocation across all aspects of the project; they need to identify all the tasks (with the help of technical manager and coordinators below) and allocate tasks at the formal team meetings (task allocation should be as fair and democratic as possible). The project manager ensures that the team submits any required forms. The project manager acts as the repository for any client documentation. The project manager maintains the task schedule and system component charts for the project. The project manager needs to be a good communicator but also have good management skills. This person should attend most meetings with the client.

**Technical manager:** The technical manager locates useful tools and software needed for the project during the analysis phase and is responsible for the technical requests document. This person coordinates the technical prototypes during the design phase. They generally implement the starting codebase. They identify implementation modules to be allocated at team meetings. They also maintain program directories, source code, and handle the duties of configuration management during the implementation phase. They are ***not*** solely responsible for developing the software, just the coordination/integration. The technical manager is normally a student with a strong Computing background (particularly in programming).

### *Coordination Roles*

**Report Coordinator(s) (only needed weeks 1-8 in semester 1, 14-17, 24-26 in semester 2)**: This coordinator is in charge of collating all the documents for the Analysis Report, the Design Report, the Review Report and the Final Project Report. Teams can have a different report coordinator for each report. The coordinator identifies sections/documents and ensures they are allocated at team meetings. They are ***not*** solely responsible for developing the reports, just the coordinating/collating. They need to have good written skills, and know how to produce professional documentation using Word or HTML.

**Manual Coordinator(s) (only needed weeks 21-26 in semester 2)**: You should have two Manual Coordinators during weeks 21-26 (one for each manual). They are in charge of ensuring that manual production is kept on track. They identify sections/ documents and ensure they are allocated at team meetings. They are ***not*** solely responsible for developing the manual, just the coordinating/collating. They need to have good written skills, and know how to produce professional documentation.

**Marketing Coordinator(s) (only needed weeks 10-13 in semester 1 and weeks 17-24 in semester 2)**: You can have up to 3 different marketing coordinators – one for presentation, one for demonstration and one for webpage. They are ***not*** solely responsible for developing these elements, just coordination. The coordinator needs to have good communication skills, both verbal and written; artistic talent is useful.

### *Team meetings*

Each team is required to have one formal team meeting each week. At this meeting you should discuss the project as a whole, check progress on individual tasks and allocate out tasks for the coming week. The task schedule should be updated to reflect work completed and work allocated. The project manager should organise times for weekly meetings.

It is important that you have other regular team meetings to work together on assessed items. You should take advantage of the tutorial slot to hold long meetings that involve major team discussion. Changes to the task allocation can only be made if ***everyone*** affected is present and the changes should be recorded in the schedule as soon as possible (otherwise task allocation changes should be made at the next formal team meeting).

Your conduct at team meetings forms part of your assessment for professionalism. This is an individual mark. You should make sure every individual has an opportunity to contribute and that the task schedule is accurate.

### *Workload*

KXX331 and KXX332 are time demanding units. You should be prepared to work steadily on your project throughout the semester. To achieve a ***passing*** grade each person should be prepared to work for at least 8 hours per week for 26 weeks (208 hours). If you want to achieve a higher grade you should expect to put in more than 8 hours a week. University guidelines suggest you should spend 10 hours a week on a unit. Previous students have spent 10-11 hours a week on average on project.

Students tend to overemphasize the importance of project, and spend too much time on it, to the detriment of other units and outside commitments. People who put in more than 20 hours a week have their priorities wrong.

The workload is spread out over the entire 26 weeks of the year, if you don't put in the hours one week, you will find you have to make it up in later weeks. Marks are allocated steadily throughout the unit so leaving all the work to the end of semester (or near a deadline) will ***not*** result in a high mark.

Your work habits form a proportion of your individual assessment for professionalism. You should make sure your work habits are satisfactory. You all get the same mark for the team component, but team members do peer assessment of each other that influences the individual component.

It is extremely important that teams try to balance the workload so that each student is making an even contribution. It may turn out that some students in a team are simply aiming to pass the unit, while others are aiming for a HD. This means the HD student needs to be doing more work of a higher quality or work of a harder nature, but not taking work off the other students. Other team members will influence your mark.

It is the team's project manager's responsibility to make sure all the tasks are allocated. The decisions about who does what should be made in a collegial manner at a team meeting. The project manager should discuss (not dictate) task allocation.

This unit provides the experience of working in a team environment. This means that if one person has commitments elsewhere, or is ill, then the rest of the team needs to cover for them. This is real world experience. It is essential that if you are going to be absent for any part of the unit that you let the rest of your team (and unit coordinator) know as soon as possible. This means you can do extra work early to make up for the time and that they can adjust their loads to cover for you while you are away.

### *Team break up procedure*

Once you are in a team you are there to stay for the year, though there is some flexibility in week 1. Unfortunately some teams do experience insurmountable problems (these are very rare) and in some cases it is necessary for an individual to leave a team. An individual could leave the team for the following reasons:

1. Failure to complete work – the student is not contributing to a level expected by the remaining members of the team.
2. Failure of team to complete work – the student feels that other members of the team are not contributing to the level expected.
3. Extreme personality clashes – the student is unable to continue associating with one or more members of the team.

An individual can ***not*** leave a team without either the student or team undertaking a three week probationary period unless the individual has found another team that they can join and ***all*** members of that team are willing to take on a new member, and (unless it is week 1) ***all*** members of the current team are willing to let that member leave. A student can, of course, choose to withdraw from the unit at anytime.

At the management meetings peer assessments and team management will be discussed, and a team or individual could be placed on three weeks probation. At the end of probation a student could be asked to leave the team. This student can then withdraw, or join another team if they know of one willing to have them, or form/join a team with other people removed from teams or if none of the previous are possible work as an individual on an in-house project for the rest of semester.

If an individual or team is put on probation, some or all of the following will occur:

- The student or team will be advised to talk to the University Counsellor;
- The team will undergo a mediation session;
- The individual (or all individuals in the team) will write weekly individual contribution reports and be required to show the work to the lecturer;
- A lecturer will attend the weekly formal team meeting.

### *Self and Peer Assessment*

Each team member will be assessing themselves and each other throughout the year. This assessment will take many forms:

- Peer Evaluation Surveys
- Individual Contribution Report
- Work Product Pay Packet

These assessments are due in weeks 5, 9, 13, 18, 21 and 27. Failure to submit an assessment form by the deadline (without a reasonable explanation) will result in a -0.5 reduction in your final grade (max -5).

It is tempting to have a pact with your team members to always give high ratings. You are advised ***not*** to do this. This encourages individuals not to do their share of the work and you will end up carrying them or submitting sub-quality work. You should respond based on ***your*** opinion of each person's contribution. You should find that if you are honest with each other you will all learn more and improve.

#### *Peer Evaluation Surveys*

These surveys ask a series of questions about a team member's performance at team meetings and about their work habits. These surveys will be used to evaluate each individual's teamwork mark for professionalism, worth 5%.

#### *Individual Contribution Report*

With each submission (reports, software, manuals), an individual will be required to write a report stating what was their contribution to the submission. It is important to provide as much detail as possible. This report must be read by all team members who must indicate agreement/disagreement. You must also write a comment about the performance of each team member.

#### *Work Product Pay Packet*

With each report, software and manual submission students will distribute $100 (virtual only, not real money) based on their opinion of the contribution by their team members, consider both quantity and quality. You do not have to give out the entire $100 (any remaining amount will be distributed by the lecturer). You can only use whole dollars (no cents). You must write a comment explaining why you have distributed the pay in the pattern you have.

For example, each student will have $100 to distribute between their team members (including self). If a person believes everyone contributed equally then they should give everyone the same amount, or if they believe that someone did more work than others, they should give that person more, and others less.

If you believe an individual has done above what was required (or asked for) then you can give them a bonus. You can only give one person a bonus, and you can't give it to yourself. You ***must*** provide an explanation of why you are giving a bonus. If you strongly believe more than one person deserves a bonus then you must choose ***one***, but in your comment state why the other person also deserves a bonus.

The lecturer will use these dollar amounts in conjunction with the Individual Contribution Reports and Timesheets to calculate an individual mark for that work product for each student.

# Lecturer Interaction

The unit coordinator is responsible for:

- Overall design and administration of the units;
- Development of assessment criteria;
- Monitoring the progress of the team project;
- Monitoring the contribution of each individual;
- Assisting the team with relationships with the client;

The lecturers are responsible for:

- Providing sufficient unit material;
- Providing support and guidance to the team;
- Ensuring students receive feedback;
- Assisting the team to develop project management strategies;
- Assisting in resolving team conflicts, which appear to be affecting the project.

### *Management Meetings*

Each team will have regular management meetings with the lecturer. Management meetings are in weeks 3 (Hbt only), 6, 10, 15, 19, 22 and 26. There will be short Management meetings in Launceston in weeks 2 and 4 (due to public holiday in week 3). The Launceston meetings are on Monday and the Hobart meetings are on Wednesday. Times for the meetings will be organised early in semester. Teams or individuals can meet with the lecturer at any time to discuss client/project/team issues (this is strongly encouraged).

The purpose of the meetings is to provide high-level management and to receive feedback on your submissions. The progress of a project will be discussed. The lecturer is particularly interested in any issues that may exist in the team, and will facilitate mediation if the team requires some attention. These meetings will also give you a chance to notify the lecturer of other resources you require for completing your project and getting specific assistance with the next submission.

At these meetings students can be put on probation or removed from the team. The students need to take the meetings seriously and see them as part of the process. The management meetings are ***very*** important and failure to attend a management meeting (without a ***prior*** reasonable explanation) will result in a loss of one (1) mark.

Email is welcome at all times, and will be answered when possible. If you want feedback on a document do **NOT** email documents, put them in your project folder and send an email explaining where to find them and what you would like feedback on. You are ***strongly*** encouraged to seek feedback on documents ***before*** submission. No feedback is given on the day of submission.

# Client Interaction

The client is responsible for:

- Supplying the team details of the user requirements for the project;
- Supplying data/content for the project;
- Attending all required and arranged client meetings with the team members;
- Providing assessment to the lecturer on the conduct of the project team and performance of the delivered system;
- Attending the presentation/demonstration at the end of semester;
- Making available any hardware or software necessary for the development of the project not freely available in the School of Computing and Information Systems.

For the project to succeed clients must be extensively involved and made aware of how the system will work and how they will use it. It is important that they are aware of progress and to give feedback during development. It is highly recommended that you are in contact with the client each week. The client liaison is the conduit for information to/from the client, other team members can contact the client but make sure that you are not all asking the same questions.

Your client interaction is assessed by the client and forms part of your professionalism team mark. Assessment sheets will be sent via email to the client throughout the year.

### *Client meetings*

Each team should have a face-to-face meeting with their client in weeks 1-6, 8, 10 and 12 in semester 1 and in at least weeks 14, 17, 20, 23 and 26 in semester 2. The lecturer is ***not*** present at the client meetings. The client liaison will contact the client to organise a date and time for each meeting – if possible you should arrange a regular day/time at the start of the semester and meet at the same time throughout the semester. The students need to create an agenda for each meeting. The agenda should be sent to the client at least 24 hours prior to the commencement of the meeting.

You should only have meetings when you have something worthwhile to discuss or progress to demonstrate or questions to ask. Clients have agreed to give you at least 20 hours of their time, you should make use of this time. It is not necessary (or sensible) for the whole team to attend each meeting, though it is desirable that all students meet the client at some time.

It is preferable if the teams make an attempt to meet the client on their premises rather than expecting the client to come to the University of Tasmania. There are some meetings where the client should expect to come to the University, particularly meetings where the students are demonstrating incomplete software or prototypes.

***Meetings during analysis phase***

These meetings are held in the first 4 weeks of semester 1 as soon as you know what project you are allocated. Often the client will give you a lot of information in the first meeting and you then need time to really think about it and come up with questions to ask in the follow up meetings.

You should be prepared to ask lots of questions, because questions can lead to further requirements being extracted. You should listen to and clarify what the client is describing. You should take notes and drawing diagrams may also be useful. If the

client agrees, tape recording the meetings can prove useful later on. Do not dictate to the client what they are going to get.

You should use the first meeting to introduce yourself to the client and make sure the client is aware of the project schedule. Make sure they understand the concept of release 1 and release 2. You should get the client to indicate mandatory features and what features they would like developed in each release. You should discuss Intellectual Property and Confidentiality. You should look at any existing software if there is any.

You may want to ask your client the following questions in the first interview.

*At the beginning*

- Exchange names
- Tell client who is the team client liaison
  - Exchange contact information, eg phone and email
- Ask for information about the client
  - Type of business in
  - Goals of business
  - Do they have a website?
- Explain the project process to the client
  - First analyse projects (4 weeks)
  - Then develop design docs (4 weeks)
  - Then implement Release 1 (4 weeks, a third of the project)
  - Release 2 next semester

*Find out about project*

- Get the client to describe what they want
  - Ask them to explain anything you do not understand
  - What they want (functionality)?
  - Why they want each function?
  - Is each function mandatory/optional?
- What do they want in release 1 or 2
- Who will use it?
  - Type of user? Eg child/adult, student/lecturer
  - Level of ICT experience?
  - Number of users?
  - How many users at one time?
- How does the project fulfil or contribute to the organisational goals?
  - How 'important' is it to the organisation?

*Data storage*

- Does it involve storing data?
- What data do they want to store?
  - Get examples of type of data
- What database system use?
- Do they already know what DB table structure they want?
- Is the data confidential?
  - Do you need a confidentiality agreement?

*Software Structure*

- How many components?
  - Eg client/server/db
- Run within a browser?
  - Webpage, program within browser (java applet, flash file)
- Work over the internet?
  - Firewall?

*Software/Hardware Requirements*

- What type of computer should it run on?
  - Mac/PC/Unix/Mobile phone/PDA
- Are there any limitations/restrictions?
  - Speed, access, size, protocols must use
- What programming language/software tools should be used?
- Does it have to communicate with any other hardware?
  - Eg printer, card reader?

*Existing System*

- Is there an existing system?
  - Can you have/see/use a copy?
- Is there a paper based system?
  - Can you get copies of forms?
- Are you just re-writing/developing existing system?
  - Or are you expanding on that system as well?
- Do they intend to expand the system that you develop in the future?
- Do they intend to use the new system for any other purpose in the future?

*At the end*

- Decide if you have enough information, or want a meeting another meeting
  - If you do, make an appointment

This is NOT a complete list of questions to ask the client, you should be able to think of some more project specific questions to ask. Do not simply read through these questions at the interview, some of them the client will answer during the discussion. Do not over PLAN the first interview; let the client give you information in the order they want to give it.

Once you have started documents such as the project brief, business case or software requirement documents you should discuss them with the client. You must make sure your client has time to read the document before the discussion.

The clients should ask for everything that they would like in the first few meetings. Generally speaking it is simply not possible for you to complete everything within 26 weeks. You have to decide what can be developed; this is a valuable experience for you. You also need to decide what will be developed in release 1 and what will be developed in the release 2 – this is written up in the release schedule. These decisions are made after consultation with the client.

***Meetings during design phase***

In week 5 you should take time to give your client a copy of the submitted Analysis Report and ask for feedback – you will write up the suggested changes as part of the Design Report.

In weeks 5 to 8 you will be developing prototypes of the GUIs (Graphical User Interfaces) for the software. You should have meetings with the client to discuss these prototypes. The GUIs are what the client will use and are the most likely thing that they will want changed. Any feedback they provide will help you fill out the GUI prototype reports.

You also need to test out anything that you are not sure will work in the client's environment. You need to find out things about the client's technological set up and make sure what you are about to do will work in their environment. This information should form part of your technical prototype report.

The client will need to hand over and explain any content required for the project – particularly important for an online content project. The content could be data, pictures, text or executable code. In the past clients have proved very unreliable about providing content, most leave it until the last few weeks, in some cases the last week. You must start applying pressure early to get content from your client.

***Meetings during the implementation phase***

You need to keep your client aware of progress, the more feedback you get early the more time you have to implement any changes that your client may ask for. It is not recommended that you do weeks of implementation without showing something to the client, or you may have to do it ***all*** again.

***Meetings at the start of semester 2***

The purpose of these meetings is to extract any changed requirements from your client and also receive feedback on release 1 and suggestions for the release 2 schedule.

***Hand over meeting***

This meeting is held in week 26 or the week after. The purpose of this meeting is to install the final release. At the meeting students will give the client an install disk and a copy of the manuals. It is important that the client knows how to install and run the software on ***their*** machine. It is best if you install it for them. Marks are allocated for installing and integrating the software.

# Project Reports

Each team must submit an Analysis Report, Design Report, Review Report and a Final Project Report. All team members must contribute to each report. Each report is made up of a number of different documents. Some examples of the different documents are available on MyLO. All documents for all reports must be electronically submitted. Each document is assessed on the basis of accuracy, usefulness, and quality.

The Analysis Report consists of:

| Document | Extent | Mark |
|---|---|---|
| Project Brief | Use template | 4 |
| Business Case | Use template | 4 |
| Software Requirement Document | Detailed description of the software requirements of the project | 6 |
| Risk Log | Use template | 4 |
| Technical Requests | Use template | 2 |

The Design Report consists of:

| Document | Short Description | Mark |
|---|---|---|
| Updated Analysis Report | Update any documents where changes were identified by client or lecturer or team | change AR mark |
| Release Schedule | Describes the 3 releases | 4 |
| Requirements Trace Matrix (RTM) | A spreadsheet itemizing each software requirement and use case | 3 |
| Software Design Document | Detailed descriptions of proposed software, include: storyboards, scenarios, UML, interface prototypes, data management, hardware/software platform, functional prototypes. | 14 |
| **One of the following:** | | |
| Business Process Modelling | Detailed description of the business process, includes diagrams | 4 |
| Market Analysis | Detailed description of the market that your product will compete in | 4 |

The Review Report consists of:

| Document | Short Description | Mark |
|---|---|---|
| Testing Summary | Describe testing of release 1 undertaken and the results of that testing | 4 |
| Exception report(s) | Update the project brief, business case, software requirement document, risk log, release schedule, RTM, and design document using Track Changes with Exception Report template as the cover. | 6 |

The Final Project Report consists of:

| Document | Short Description | Mark |
|---|---|---|
| End Of Project Report | Use template to reflect on achievements | 4 |
| Testing Summary | Describe testing of release 1 & 2 undertaken and the results of that testing | 6 |

Assessment Templates for all these documents are available on MyLO. It is strongly recommended that you read these to ensure you cover the correct material in each document.

### *Project Brief*

A template is available on MyLO along with a learning module that describes how to complete this document. The project brief captures the key aspects of the project, sets out the objectives and scope and outlines the deliverables and constraints.

### *Business Case*

A template is available on MyLO along with a learning module that describes how to complete this document. The business case describes the added value of the project to the client's business objectives. The business case should include attachments that describe the organisational strategy (possibly include modelling diagrams here) and the goals of the business (where the business is going and what they hope to achieve). The business case sets out the reasons for the project and costs and benefits of the project, details the quality expectations and acceptance criteria.

### *Software Requirement Document*

There is a learning module available on MyLO that describes how to complete this document. The Software Requirement Document should describe the software system that is required to achieve the goals of the business. Include everything that the client has asked for, even if you won't finish it all. Your document should cover everything that the client has asked for (the ***what***) and also any discussion of each point that the client may have given you (the ***why***). The reason why the ***why*** is so important, is that later on you will be implementing a feature, and you will be wondering if it is actually serving the purpose that your client needed it for. If you have the ***why*** in your requirement document you will be able to quickly check. You do not have to give a ***why*** for every ***what***, but you should be able to provide a substantial amount.

Ensure that all the requirements (functional, non-functional, and domain) for the project are elicited, and that these requirements are described in a manner that they are valid, consistent, complete, realistic and ***verifiable***.

The software requirement document is typically 6-10 pages of formatted text produced in Microsoft Word. Layout is important; it should look professional. You should put the requirement document into meaningful sections. Generally speaking, you can divide your system up into a number of components: eg Server, Client and Database or Administrator, User and Database. You could have sections like this in your document. It is suggested that you number each section. You should not have a section such as 'Requirements' or 'System Requirements' as the whole document is about requirements.

Do not write your document as a list of dot points. Do not number every paragraph. Neither of these approaches is very professional. The document should be in paragraph form. One line paragraphs should be minimised. You should indicate what things are mandatory and optional. Remember you are only expressing what the client has told you, you can ***not*** decide what is mandatory or optional. Some indication should be given of the priority of features, if the client has given a priority.

Do not include a section on Implementation Platform, Language or Tools, unless the client has specifically told you to use those tools. You want to leave your options open. You should put all technical information in the Technical Requests Document.

### *Risk Log*

A template is available on MyLO along with a learning module that describes how to complete this document. The risk log identifies the possible risks to the completion of the project. It indicates the likelihood of the risk occurring and what impact it will have if it occurs. It also indicates some strategies to either avoid the risk, minimize the impact if it does arise or contingency plans if the risk arises.

### *Technical Requests*

A template is available on MyLO along with a learning module that describes how to complete this document. This document is meant to be a one page document. This report is primarily the responsibility of the Technical Manager, though its contents should be discussed with other team members – decisions are made as a team. It must describe your development platform, the programming language(s) you want to use, any development environments or tools you want to use. It must list any other items you will need, such as APIs or DLLs.

### *Release Schedule*

The requirement document is a description of ***everything*** the client wants. It will not be possible to for you to provide everything. You need to decide how much you can achieve over the year. You need to let the client know what you have decided, and allow them to make changes. A release is a distribution of software functionality to the customer. Release 1 should constitute approximately a third of your project. Release 2 should be the remaining two thirds of what you will do. There may be some things that you just won't have time to do, these should go into Release 3.

In first semester you indicate what you will have completed for each release. It is very rare that you complete everything you say you will by the end of release 1; students tend to overestimate how much they can do. At the start of second semester you can re-negotiate release 2 with your client based on your experience in first semester.

The information about each release is presented in a release schedule. All teams need to do a release schedule. A release schedule is a Word document; it should be about 3 pages long. Each release description should be about a page long, though release 2 can be longer and release 3 shorter. Do not simply copy the requirement document. You are now describing what the client will get combined with ***how*** you are going to do it. It is very important that you identify what a user will be able to do with your system at the end of each release. This information, apart from informing your client, is used to do your assessment.

Think carefully about what goes into Release 1, it should ***be useful*** for something. In particular, you should attempt to develop a number of interfaces in release 1, as these are things that the client generally wants to change. Generally speaking you don't do authentication of the user in release 1 as you get limited feedback from the client (it either works or it doesn't). You should generally try to get database implementation out of the way in release 1.

The release schedule is given to the client, so they know what you intend to develop this year, and what things won't be attempted. It must look professional. It should be written in paragraph form, sections are optional. Do not present a list of dot points.

### *RTM*

Typically an Excel document with the following columns: Number, Paragraph, Requirement, Type, Release and Use cases. The RTM can only be done once the software requirement document is completed. The RTM must be compatible with the Release Schedule.

### *Business Process Modelling*

There is a learning module available on MyLO that describes how to complete this document. This report is a detailed description of the business processes that your client uses, in particular in will focus on where your software will fit in the process and how it will improve/change the process.

### *Market Analysis*

There is a learning module available on MyLO that describes how to complete this document. The market analysis should identify such things as the target audience, competitors, a SWOT analysis, economic environment (eg final price, competitor prices, current relevant economic situation of market), and a marketing/promotion strategy.

### *Software Design Document*

The design document is a Word document and the body text should be at least 20 pages long but no more than 30 pages with the font set at 12pt Times New Roman. The document also has appendices that are ***NOT*** included in the page count above. The design document is a living document that is maintained throughout the development and changed frequently; you will submit the original in week 8 and a revised version (with Track Changes on) in week 17.

Before implementation can begin, the team produces the first version of a design document incorporating all or most of the material from the software requirement document but including extensive detail and prototypes. The design document should thoroughly document the following aspects: interfaces (including interface prototypes), interactions (including scenarios), functional mechanics (including functional prototypes), data management (including structure), models (including storyboards) and hardware and software platforms (including justification).

The following sections describe some of these elements in more detail.

#### *Scenarios and OO Models*

The section consists of detailed scenarios and a set of OO models.

Scenarios provide the operational concept behind a Use Case. A Scenario is a formatted description of the steps required for the completion of a Use Case. There is one scenario per use case.

The most important section is "Scenario", the action column contains all the user actions, no matter how trivial. The software reaction contains details of everything the software should take care of: how the screen is updated; any changes to file structures

such as databases; any flags that should be set; any error windows that may come up; literally everything the software should do in response to the user action.

An object oriented design should be completed for any project that requires substantial programming, particularly if the programming language is object-oriented. You can use any UML modeling program you like for this, but it must be capable of exporting the diagrams into a readable format that can be assessed, eg .pdf. You can do things like package diagram, class diagrams, sequence diagrams, activity diagrams or state diagrams.

If required, scenarios and UML diagrams should be included in an appendix. You may include some of the important scenarios in the main body of the document (or excerpts from one or two) to explain the interactions of main parts of your software. You may want to include a package diagram or a major class diagram in the body of your design document.

***Online Content***

The bulk of this content would be in the body of the design document. If you are working on an online content project you should produce a detailed textual description of the layout of your content. Online content could mean a web site, or a CD or a training package. Often the client does not provide many details about the structure or layout. These are generally provided by the team members, but need to be confirmed by the client. What is the expected content (does not include the actual textual content)? The first thing to consider is the hierarchy (layout or structure) of your content. How does the user navigate around the content? Is there a menu? What is the structure of this menu? If you are going to extract data from the user then you need to work out what you will be asking for.

***Game/Model Storyboards***

If you are working on a game or modelling project you should produce a detailed textual description of your game and model. Generally the client does not provide many details about the model structure or the game structure. The team members generally provide these but need to be confirmed by the client.

You should discuss things like the game genre and influences, the core mechanics and control scheme, and artistic style. You should cover the goals of the game and any sub-texts. You should also cover the storyline if there is one, you should develop and discuss the characters, you should discuss the pacing of the game and also the learning curve involved. The bulk of this discussion would be in the body of the design document.

If you are producing a 2D/3D model, game, video or animation then you should do storyboards. A storyboard consists of a series of diagrams and explanations of the diagrams.

The design document should include some major storyboards in the body text. The design document should also include an appendix that includes all the storyboards (by week 8 there should be the storyboards for at least release 1 and by week 17 the storyboards for release 2). This appendix is a living document and should be kept up to date.

***Data Management***

Typically data is stored in a database or is imported/exported using XML or data is simply imported/exported from a data file.

If you are using a database the report should consist of a database diagram showing tables, attributes and relationships. There is more information on MyLO on how to do database diagrams. The report should also contain a detailed description justifying the decisions about tables, attributes and types.

If you are going to use XML in your project, you need to describe your document type definition or Schema, and show some example XML. There are some examples available on MyLO.

If your software is going to read/write external text files (data files) you need to provide a detailed description of the purpose and format of these files.

A general summary of your decisions should be included in the body of your design document; the detailed descriptions should be included in an appendix (eg table structure, schema, file hierarchy).

***Interface Prototypes***

Your software will consist of a number of GUIs (screens), for example, a main screen, some different dialog boxes to get information, a screen to display the results of some analysis. You have to prototype each distinct screen (initially as a sketch with major release 1 screens prototyped by week 8, all screens should be prototyped by week 17). Typically a team will prototype 5-8 screens.

It is important to prototype your interfaces, this will mean you have something concrete to show your client to get feedback. Early feedback from your client on the interfaces can lead to greater client satisfaction. The client hardly ever likes your first design of the interface, and you need to find out the problems as soon as possible. Prototyping is also a good opportunity to learn your development language and experiment with your development environment.

Generally speaking you should do a prototype for each distinct GUI, you should group similar GUIs, e.g. enter data and edit data GUIs are generally very similar, all error GUIs have an extremely similar look and feel. The first screen you should prototype (to completion) is the main screen or equivalent. For the main screen layout, it is common to produce a number of prototypes (e.g. each team member do a version) and allow the team members and/or client to provide feedback as to which one is best (or which bits from each should be used).

If you have scenarios draw a draft of each GUI identified on paper. Every button, menu option, text field, etc referred to in the scenarios should be on the GUI. If you put a button on a GUI that is not mentioned in a scenario somewhere then there is a problem with the scenarios and vice versa. You must sketch every menu, so that you have some idea of the options, navigation and number of levels involved.

If you have storyboards have a team meeting to ensure a consistent look and feel has been used throughout. It is most likely that you will have an intro screen and/or menu page, these at least should be prototyped in the implementation language.

If you are doing online content then at least one section of your content needs to be prototyped. It is probably best to do this using the particular development software you have decided to use for implementation. Ideally you will use realistic content, but

it depends on whether your client can supply it in time. You need to develop a prototype of your home screen and menu system for your program. You are prototyping the overall layout and showing how people navigate around your content, using relatively blank pages to represent the rest of your content. You are not doing any difficult programming at this stage, just attempting to get an understanding of how your content will be displayed.

When prototyping a screen it is normal to design a number of versions on paper first, have a team meeting to decide which ones (or bits of ones) are worth implementing (you should show the client more than one version as this gets greater feedback). The paper designs should then be implemented in the implementation language for the next stage(s) of the prototype. Once completed, have another team meeting, provide feedback and make changes, and then show them to the client to get feedback, and then create the interfaces based on client feedback. Then possibly have another round of team feedback and finally client feedback, creating new versions after each set of feedback. Going through this process is how a screen prototype goes through a number of stages.

Overall when you reach the final stages of each screen they should use the same layout principle, they should all use similar names for links or buttons, and the colours used in headings and borders should all be similar.

If you are adding to existing software, you should use this opportunity to link your new GUIs in with their old ones. This is an opportunity to understand the program you will be adding too, and an opportunity to learn the language.

In all the above you should aim for a consistent look, remember though that this is a prototype. The client won't like everything that you do; you ***will*** have to change it. The idea is to get the overall look ready so that your client can say what they do and don't like. Get feedback from client often; remember to write up their feedback in the prototype reports.

The design document should include some major interfaces (sketches or screen shots) in the body text. The design document should also include an appendix that includes all the interfaces (by week 8 there should be the prototypes for at least release 1 and by week 17 the prototypes for release 2). Each interfaces needs a diagram ***and*** a full description of what the impact of interacting with each component is. This appendix is a living document and should be kept up to date.

***Hardware/Software Decision***

Typically each project requires you to make some technical decisions – eg what programming language to use, what hardware to use, what library to use, etc. These issues need to be researched and decisions made using a logical process. It is this process that is written up in the design document.

A technical decision prototype has a definite single purpose – eg choose development language, choose a particular piece of hardware. Generally speaking there are a numbers of options available to you and you have to make some sort of choice.

To determine an outcome it is normal to have some criteria on which to make the decision. For each criteria there should be a rating system so that each possible option can be ranked against the criteria. A decision is made in stages. Each stage is investigating a different option against the specified criteria and giving it a rating.

Once each option has been investigated, a decision is made on the basis of how each option rated against each criteria.

Each technical decision should be written up in the design document as an appendix using the following sections:

1. An overall description of what decision you are trying to make.
2. A description of the criteria and rating system for each criterion
3. For each option include:
    a. A description of the option.
    b. An analysis of how this option rates for each criterion along with a discussion as to why it was given that rating.
4. A conclusion describing which option has been chosen and why (include a summary table. [This conclusion should also be summarised in the body text of the design document].

### *Functional Prototypes*

You must develop prototypes showcasing one or more crucial features (mechanics or significant screens) that could be incorporated in the final product. Developing prototypes of features allows you to experiment with different algorithms and usability scenarios for your software. A lot of prototyping is expected to take place during the design phase before the design report is complete. Prototypes act as a proof of concept. Most prototypes will be ported to the software system.

Typically each project requires you to undertake something that you are going to have to research how to do, it is not something that you have been taught in your other units. These processes need to be researched and a number of different methods investigated. It is this process that is written up in the design document.

A functional prototype has a definite single goal – eg how to create a graphical asset, how a particular feature will work, how to interact with another software package. Generally speaking there are a numbers of options available to you and you have to make some sort of choice. There should be a number of methods you could use to achieve each goal and you need to describe a number of them and then explain why you have chosen the one that you have chosen.

Each functional prototype should be written up in the design document as an appendix using the following sections:

1. An overall description of what you are working out how to do.
2. A brief description of each possible path that could be taken to achieve this goal and an analysis of why you have chosen a particular approach. [This conclusion should also be summarised in the body text of the design document, possibly including screen dumps].

## *Exception Report*

A template is available on MyLO along with a learning module that describes how to complete this document. Any significant changes to the project brief, business case, software requirement document, risk log, release schedule, RTM or design document requires the team to fill in an exception report for each change and produce an updated document as an attachment. You should use track changes on your documents, so that the changes can be identified.

### *Testing Summary*

A template is available on MyLO along with a learning module that describes how to complete this document. Use the RTM to itemize the requirements, extract the verifiable/measurable criteria for each requirement from the requirement document and test whether the requirement has been achieved. Also testing for overall usability and appearance of the software.

### *End Of Project Report*

A template is available on MyLO along with a learning module that describes how to complete this document.  This states whether the objectives have been met, deliverables achieved and benefits realized.

# Documentation

Each team is required to produce a user manual and a reference manual in KXX332. The manual coordinator is responsible for coordination and layout and formatting, in particular the Table of Contents. Every team member should participate in the text and diagram production. The manual coordinator does ***not*** have to write everything. Some examples of previous manuals are available on MyLO.

**Proof read the manuals!!! Every team member should do this!**

### *User Manual*

The user manual is written for the client. It is assessed on how well your client will understand what you are talking about. If your system is used by different types of people, eg an administrator and a user, you may consider producing two separate user manuals. The user manual must have a very professional look and feel.

This document can be a paged document (.doc or .pdf) or online (.html). The length of this document will vary considerably between teams, but as a guide a 40-50 page document is typical.

You must describe the software only, not your team, not the course. You should have descriptions on how to use every element of your program. The manual should be about your program, not primarily about installing drivers, etc. Include the following:

- Introduction
- Explain what your program can do
- Include screen dumps (but do not overly repeat the same diagram)
- Explain ***every*** input field
- Explain every menu option
- Starting and quitting
- Getting help
- Installing – this chapter may be in the reference manual.

Where relevant:

- Getting Started (a step-by-step through all initial common steps)
- A Tour (of all main features)
- Saving information
- Any networking that needs to be set up
- Printing
- Customizable aspect of your program
- Trouble shooting – (explain all error messages)

You should think carefully about the order in which you present the information. You can put similar elements into the same chapter. Put the more advanced aspects towards the end. Look at the layout of each page, where possible put text describing a diagram with that diagram. Make sure the text matches the diagram, eg close v cancel buttons. Do not make the document longer by having multiple copies of the same screen dump. Include page numbers.

### *Reference Manual*

The reference manual is written for people with programmer or system administrator ICT knowledge, it is very technical. The main purpose of the reference manual is to aid people who want to change/update your software at a later stage.

The length of this document will vary considerably between teams, but less than 50 pages would be unusual. This manual should be in an interactive html format that can be accessed via an index.html file or can be a paged document (.doc or .pdf)

The introduction should explain what your program is about from a technical point of view. This should not be the same introduction that appears in your user manual but it will be similar.

You should explain the general structure of your program, e.g. client/server, stand-alone application.

You need to explain each file, why it exists, and in general what is in it. You do not have to explain every method, so long as there are comments in your code explaining every method. You can use an electronic documentation creator, such as javadoc, to extract your comments that explain each file and method; this information should be put into an appendix.

If your program reads/generates external files or uses a database you need to explain the format of every text file or database table that accompanies your program. This will be ***very*** similar to what was in your data storage report.

You need a section that covers how to re-compile your program. In particular this must include a list of the source files needed and what versions of software you used to compile your program. You also need to describe the development tools used, including version numbers.

You need to have a section giving an example of how to change the program. Typically you won't have implemented everything the client has asked for, or you know of areas where your client is going to expand your software. You need to give a detailed description of steps a programmer should go through to change your software (including things like tools needed, important files that will need to be changed).

If it doesn’t appear in your user manual then you should have a chapter than covers installing your program. This may occur when the user of the program is not usually the person who installs it (for example, something that works over the Internet). It is very important that this chapter covers every step.

You may also want to include a trouble shooting section, if you have had problems understanding how to get it to install or compile then so will your client.

# Web Page

Each project team is required to maintain a small web page describing their project. This should be in your /public_html/web folder. Create a file called index.html, this is the base for your web page. The web pages will be made available externally via the school webpage, for this reason you can not include any executable features in your webpage, no PHP.

Every website should have a title and menu, a corporate look and feel and be professional looking. The menu must have the following options:

- Home – (index.html) has a brief description of purpose of the project
- Team – first page describes team with team photo, linking to individual pages
    - Each member has page with short resume
- Client – describes the client
- Tools – describes the languages and development tools used, including links to other relevant sites.
- Software – a clear description of the software developed
    - Include screen dumps if possible, but not executables
    - Do ***not*** include copies or links to completed documentation

The web page will be assessed at the end of second semester, but an initial assessment is undertaken on Monday of week 22 (the software page does not have to be complete at this time). Teams are advised that all the publicity for demonstration day starts in week 22 and many industry members look at the pages in week 22/23, you are advised to make sure your web page is presentable and up-to-date at this time. You can continue to change your webpage up to week 26. At the end of the year the web pages are moved to the past projects website and are available for eternity.

# Presentation

You must formally present your first release. Clients, staff and class members will be invited to listen and bring guests with them. Team members can invite other people (students, family, and employers) to listen. You should treat this process seriously and dress and act in a business-like manner. You should treat the presentation as though you were tendering/competing to do this project in the real world. You are aiming to convince your client that you are the right team to employ to develop release 2.

The presentation will occur in week 13. Launceston presentations will be held on Monday, Hobart presentations will be held on Wednesday. You will be required to attend the presentations by other teams.

The presentation should be no longer than 35 minutes (less than 20 minutes would be too short); include a demonstration and a PowerPoint presentation. All team members should participate in preparing the presentation and demonstration. You can have a subset of your team do the presentation/demonstration or you can all participate. All team members should be available to answer questions. Your presentation should contain the following:

- An introduction of your team members
- A description of the client and the client's business
- A description of the problem that the software is meant to solve
- A demonstration of the solution and software developed so far
- A description of the tools used
- A description of the difficulties/challenges you have faced
- A description of the work to be completed for release 2

You do not necessarily have to have a slide with the title difficulties. You can simply discuss the difficulties throughout your presentation. The kind of things you should talk about are:

- programming languages - are they new to you
- development tools used - is this the first time you have used them
- number of components (client/server/database, etc)
- anything that was hard or time-consuming to do
- lack of information or resources, or late receiving information or resources

A panel consisting of the lecturers and client will assess the presentation. The panel will assess how well you present and demonstrate, and they will also assess your release 1 software.

# Demonstration Day

The demonstration will be a public demonstration of your final release. Clients, staff, students and members of the Tasmanian IT community will be invited to come and look at the final products.

Demonstration days are held in week 24. The Hobart version will run from 10am - 2pm on Wednesday. The Launceston version will run from 10am - 1pm on Monday. The different duration is due to the amount of time it takes to assess all the teams.

Setting up demonstration day takes a considerable amount of time: typically one hour before it starts in the morning and one hour after it finishes. Students from each team will be required to help with the setup and pack up.

### *Demo Day Assessment*

The assessment will be based on how well you market your project and how well you demonstrate the project.

#### *Demonstration*

You need to prepare a suitable demonstration of your product and be prepared to talk to interested people. At least two people from each group should be available at all times during demo day, and everyone should participate. You should organise a roster.

Your demonstration should contain the following:

- A description before starting to use the program of what the project was all about.
- A complete demonstration of all the standard type features of the program.

The complete assessment criteria will be discussed in the lecture and are available on the MyLO site. The demonstration is worth 6% of your final mark.

#### *Marketing*

You should decorate your display area (booth) using such things as project posters, pamphlets, name tags, client merchandise/equipment. Warning: each team only has limited space, around 0.5m x 1.5m (a computer bench).

The marketing coordinator should coordinate the marketing materials, but all team members should contribute text and diagrams and provide feedback and suggestions.

Each item is assessed on the basis of appropriateness and quality. The money spent on an item has ***NO*** impact on assessment – do ***NOT*** spend too much money. Demo Day Marketing is worth 5% of the final assessment. There is an assessment of individual contribution to marketing using peer assessment tools.

*Poster*

You should prepare a poster for display on demonstration day. The poster should include the following: title, team members, client name, description of the purpose of your final product, description of what you have achieved, information about tools used, screen dumps that give a good impression of the interface.

*Pamphlet*

This is typically an A4 sheet of paper, possibly in 3-fold format that describes your project and team. You hand out these pamphlets to visitors to your stall. Printing these

pamphlets is the financial responsibility of the team – we suggest no more than 20 pamphlets unless you expect to be extremely popular. Black and white pamphlets get the same marks as colour ones!

*Name Badges*

Each student should wear a name badge. Printing these is the financial responsibility of the team. Black and white versions get the same marks as colour ones!

*Client Materials or Project Related Materials*

Some clients have materials, such as posters or artifacts that you can borrow to decorate your area. Depending on the domain of your project you can use interesting artifacts to decorate your area, e.g. sporting equipment for a sporting project.

*Giveaways*

In the past students have given away lollies or other things such as key rings. This is proving too expensive and this practice is discouraged.

*Attire*

You should treat this process seriously and dress and act in a business-like manner. A coordinated team approach is best, for example matching t-shirts or tops or similarly coloured clothes. It is ***NOT*** necessary to buy a suit for the occasion.

*Attractiveness*

Displaying your materials is just as important as having them. Think about how to use your display space, do not overcrowd, use items to emphasize displays such as boxes to raise height or tablecloths. You will also be assessed on how you attract people to your area and interact with the people around your booth. Do not have too many team members in your area, so that visitors can not see your project.

# Timesheets

Timesheets need to be filled in when you work on project to keep a record of how much each person is contributing to the project. These will show who is doing what, and that you are meeting the time requirements for this unit. Include lectures, all meetings, any reading you do, any programming, anything to do with the project.

You are required to spend at least 8 hours a week on Project on average. There will be weeks when you won't do 8 hours; this is not a problem as you should make up the hours in other weeks. The lecturer will only be concerned if you consistently work less than 8 hours. Any excessive contributions will be looked at seriously.

The timesheet system is available via the project website. There is a paper version in appendix A. The timesheets should be kept daily as this is the best way to ensure accuracy. The web server goes down frequently; if the web server goes down on Monday night there will be no extensions granted, as all the times should ***not*** be entered at the last minute.

Classify each of the tasks you do as one of the following job codes:

| **Job Code** | **Tasks** |
|---|---|
| Meeting[1] | Formal Team, Client, or Management meeting<br>Preparing for any of these meetings eg arranging a time for the meeting, phone calls, etc |
| Admin | Peer Assessment, Task Schedule, reading/writing email relevant to project, writing up minutes of meeting |
| Study | Reading project manual or online materials, reading other project management material<br>Attending lectures<br>Reviewing past material from previous years (eg design docs, manuals, marketing materials)<br>Web browsing for similar software or software tools<br>Web browsing for project content, eg pictures, text<br>Experimenting with other similar software,<br>Learning programming languages<br>Doing online tutorials about the tools or languages |
| Report | Any work on a report (analysis, design, review, final project), includes both development and proof reading<br>Meeting to work as a team (or partial team) on a report<br>Does not including prototyping, but does include time writing prototype reports<br>Does not include testing, but does include time writing testing reports |
| Implementation | Creating software (programming, graphical modelling, whatever is involved in developing system)<br>Meeting to work as team (or partial team) on implementation<br>Developing prototypes |

[1] Some teams meet to do work on a specific item, eg requirements document. This should be classed as report not meeting. If the meeting was to work on software, then it should be classed as implementation, not meeting. It should only be recorded as meeting if there was no specific work product produced, just checked or allocated.

| | |
|---|---|
| | Meeting to work as a team (or partial team) on a prototype<br>Testing software<br>Meeting to work as team (or partial team) on testing |
| Manuals | User Manual, Reference Manual,<br>Includes both development and proof reading |
| Marketing | Presentation, Demo Day, marketing materials, Webpage<br>Includes both developing material and attending |

In the comment field you must describe in detail what you did, e.g. What particular document you worked on and what you did to it; what type of meeting it was and what you discussed. A comment entry would generally be about 10-30 words. Failure to enter proper descriptions will lose marks.

Sometimes it may be hard to know what to classify a task as. Try to be as accurate as possible. For instance, if you have a team meeting to primarily work on a document then classify it as report, but if you have a team meeting to distribute out report work, then classify it as Meeting. Sometimes you may do multiple things in the one session, e.g. you may do some study, and then implementation, and then some administration. You should break the timesheet entry up into separate entries reflecting the amount of time you spent on each job code.

A timesheet spans from Monday to Sunday. A timesheet for the previous week must be completed by the following midnight Monday. A timesheet is assessed based on your level of participation and the following rules:

- Each weekly timesheet is worth 0.5%, to a maximum of 5%.
- The quality of comments may mean you do not earn the full 0.5%.
- If there are less than 5 entries in a week by Monday midnight you will get 0 for that week.
- Any entries added after midnight Monday will be considered late, and will not be counted in the above rule.
- If you average less than 8 hours over a semester (including the late entries), you will score 0 for each week you worked less than 8 hours.
- There are 13 weeks a semester; the three weeks with the lowest score will not be counted, allowing you to have some light work weeks with no penalty.

It is in everyone's best interest if you each keep honest records. The data on your timesheets (and the number of hours you spend on each work product) is used to evaluate your individual mark for a work product. It is ***incredibly*** important that your timesheets are accurate, and that you classify each job you do correctly. Team members should look at each others timesheets and ensure that they are accurate. If your team members are lying about the time they spend on project you should tell the lecturer, as this can impact on your mark.

**If it is not on your timesheet, you did not do it!**

# Task Schedule

At the weekly formal team meeting you should check the progress of each team member on the previous tasks allocated, and provide feedback to that person on their contribution. During the meeting new tasks should be allocated for the following week(s).

As soon as possible after the conclusion of the meeting (within 24 hours), the project manager should update the task schedule. This updated task schedule should reflect the progress made on the tasks allocated at the previous meeting and should also show the tasks allocated at this meeting.

The project schedule should be developed using Microsoft Project. There is a template available on MyLO, you should add subtasks as you identify them. There will be lots of subtasks to add.

The schedule must contain accurate records of who has completed which tasks, and a plan showing who is responsible for completing tasks in the coming week or further ahead if decided at the team meeting.

Only tasks relating to project development should be recorded in the schedule. These things relate to anything to do with the production of the project reports, implementing the software, preparing the manuals, or tasks related to marketing materials. These tasks typically take longer than one hour; in fact most will take weeks. Do not put meeting attendance, administrative tasks, or tasks that relate to professionalism in the schedule.

After each team meeting the project manager should update the task schedule to reflect the outcomes of the meeting. Once the task schedule is updated the rest of the team should review it (within 24 hours). If changes are required the changes should be made.

The schedule is a permanent document in your submission folder. It must always be up-to-date, within 24 hours of a formal team meeting or any other meetings that change the allocation (remember we know when the team meetings are because you put them in your timesheets). It can be assessed at anytime, but typically just before a management meeting, feedback will be provided at your management meetings.

***System Component Charts***

The technical manager and coordinators (in conjunction with the project manager) should develop a system component chart for their key development item (eg Analysis report, Design report, Software, Manual, webpage, demoday, etc). This chart should break the item down into sub-tasks/components. You should use this component chart to identify the critical path and critical sub-tasks. The project manager should maintain the charts. You should use this chart to do task allocation. The sub-tasks should be colour coded to indicate in-progress, completed, not-started.

# Assessment

### *Project A Assessment*

45% Reports
27% Software
10% Marketing
18% Professionalism

Project A is 100% internal. Each student will get an individual grade that is made up of an individual component worth 40% and a team component worth 60%. To pass Project A you have to get 45% of both components and more than 50% overall. So to pass Project A you would have to get at least 18/40 for the individual component and at least 27/60 for the team component and at least 50/100. The individual component is made up of: 18% Reports, 9% Software, 3% Marketing, and 10% Professionalism. The team component is made up of: 27% Reports, 18% Software, 7% Marketing and 8% Professionalism. Each person in the team will get the same mark for the team component, so it is very important that you all work as a team and contribute to the best of your ability.

Each report will be assessed on the basis of quality, accuracy and presentation by the lecturer. There are two reports (Analysis Report (20%) and Design Report(25%)). Your contribution to the reports is worth 18% of the individual mark and is assessed by the lecturer using peer assessment tools.

A panel (17%) will assess the software. The assessment will be based on what you undertook to produce for release 1. The software assessment will be performed at the presentation. Your contribution to the software is worth 9% of the individual mark and is evaluated by the lecturer using peer assessment tools.

The presentation will be assessed by a panel. All students are required to actively participate in the presentation (or its preparation) which is worth 7% of your team mark. Your contribution to the presentation (or preparation) is worth 3% of the individual mark and is assessed by the lecturer during the presentation and by using peer assessment tools.

Peers and the client and lecturer will assess professionalism. The client interaction assessment will form part of the team component (5%). For each meeting such things as sending agenda, punctuality, asking questions, clarifying points, listening, making the meetings worthwhile will be considered. Keeping the client informed of progress and problems via email will also be considered.

The student's approach to working in a team will be assessed by their team members and the lecturer and form part of the individual component of professionalism (5%). It includes such things as your attendance at team meetings, contributing to the ideas and discussion at the meetings, completing work by deadlines set at team meetings, your ability to work with team members and perform your management/coordinator role. Assessment will be completed using the peer assessment forms. Each student's level of professionalism can be discussed at management meetings with the lecturer.

The lecturer will assess timesheets and task schedule. The assessment for the timesheets is on-going throughout the semester and they are assessed using the rules under the section titled Timesheets, maximum 5% towards the individual mark. The

assessment of the task schedule is on-going throughout the semester, and the task schedule is worth 3% of your team mark.

Various penalties will be applied throughout semester. Failure to submit a peer assessment form by the deadline (without a reasonable explanation) will result in -0.5 penalty to your individual mark. Failure to attend a management meeting (or rude or abusive behaviour) will mean a -1 penalty to your individual mark, lateness to a management meeting is a -0.5 penalty. Failure to have a team representative at a lecture (without a reasonable explanation) will mean a -0.5 penalty in the team mark. Maximum penalties (-5).

| Item | Total | Team | Ind | Components | Worth |
|---|---|---|---|---|---|
| Reports | 45 | 27 | 18<br>max 24 | Analysis<br>Design | 20<br>25 |
| Software | 27 | 18 | 9<br>max 12 | Software | 27 |
| Marketing | 10 | 7 | 3<br>max 5 | Presentation | 10 |
| Professionalism | 18 | 8 | 10 | Client Interact<br>Teamwork<br>Timesheets<br>Schedule<br>Penalties | 5<br>5<br>5<br>3<br>max -5 |

Note: When peer assessment is used it is possible for a team member to get more than the amount allocated for the individual mark. In this case the excess can still be counted (up to maximum indicated) towards the student's individual mark, so long as the combined individual marks do not exceed 40 and the total mark for that item does not exceed the number in the total column.

***Project B Assessment***

20% Reports
32% Software
15% Manuals
15% Marketing
18% Professionalism

Project B is 100% internal. Each student will get an individual grade that is made up of an individual component worth 40% and a team component worth 60%. To pass Project B you have to get 45% of both components and more than 50% overall. So to pass Project B you would have to get at least 18/40 for the individual component and at least 27/60 for the team component and at least 50/100. The individual component is made up of: 8% Reports, 12% Software, 10% Professionalism, 5% Marketing and 5% Manuals. The team component is made up of: 12% Reports, 20% Software, 8% Professionalism, 10% Manuals, and 10% Marketing. Each person in the team will get the same mark for the team component, so it is important that you all work as a team and contribute to the best of your ability.

Each report will be assessed on the basis of quality, accuracy and presentation by the lecturer. There are two submissions: Review Report is worth 10% and the Final Project Report is worth 10%. Your contribution to reports is worth 8% of the individual mark and is assessed by the lecturer using peer assessment tools.

A panel of people will assess the software, worth 32%. The assessment will be based on what you have produced. The software assessment will be performed during demonstration day and at the management meeting in week 26 and at the client handover meeting. Your contribution to the software is worth 12% of the individual mark and is evaluated by the lecturer using peer assessment tools.

A panel will assess the demonstration (6%) on demonstration day. The demonstration will be assessed based on how you demonstrate your software. The lecturer will assess how you have marketed your project and how you attract people to your area (5%). The web page will be assessed by the lecturer and is worth 4% of your team mark. Layout, grammar, and having the required contents will be assessed. The marketing assessment will include a 5% individual component and is assessed by the lecturer using peer assessment tools.

Teams must produce a user manual (7.5%) and reference manual (7.5%) and is assessed by the lecturer. All items will be assessed at the end of the semester. Layout, grammar, and having the required contents will be assessed. Your contribution to the manuals is worth 5% of the individual mark and is assessed by the lecturer using peer assessment tools.

Peers and the client and lecturer will assess professionalism. The client interaction assessment will form part of the team component (5%). For each meeting such things as sending agenda, punctuality, asking questions, clarifying points, listening, making the meetings worthwhile will be considered. Keeping the client informed of progress and problems via email will also be considered.

The student's approach to working in a team will be assessed by their team members and the lecturer and form part of the individual component of professionalism (5%). Such things as your attendance at team meetings, contributing to the ideas and discussion at the meetings, completing work by deadlines set at team meetings, your

ability to work with team members and perform your management/coordinator role. Assessment will be completed using the peer assessment forms. Each student's level of professionalism can be discussed at management meetings with the lecturer.

The lecturer will assess timesheets and task schedule. The assessment for the timesheets is on-going throughout the semester and they are assessed using the rules under the section titled Timesheets, maximum 5% towards the individual mark. The assessment of the task schedule is on-going throughout the semester, and the task schedule is worth 3% of your team mark.

Various penalties will be applied throughout semester. Failure to submit a peer assessment form by the deadline (without a reasonable explanation) will result in -0.5 penalty to your individual mark. Failure to attend a management meeting (or rude or abusive behaviour) will mean a -1 penalty to your individual mark; lateness to a management meeting is a -0.5 penalty. Failure to have a team representative at a lecture (without a reasonable explanation) will mean a -0.5 penalty to the team mark. Maximum penalties (-5).

| Item | Total | Team | Ind | Components | Worth |
|---|---|---|---|---|---|
| Reports | 20 | 12 | 8<br>max 11 | Review<br>Final Project | 10<br>10 |
| Software | 32 | 20 | 12<br>max 16 | Software | 30 |
| Manuals | 15 | 10 | 5<br>max 7 | User<br>Reference | 7.5<br>7.5 |
| Marketing | 15 | 10 | 5<br>max 7 | DemoMarket<br>Demonstration<br>Webpage | 5<br>6<br>4 |
| Professionalism | 18 | 8 | 10 | Client Interact<br>Teamwork<br>Timesheets<br>Schedule<br>Penalties | 5<br>5<br>5<br>3<br>max -5 |

Note: When peer assessment is used it is possible for a team member to get more than the amount allocated for the individual mark. In this case the excess can still be counted (up to maximum indicated) towards the student's mark, so long as the combined individual marks do not exceed 40 and the total mark for that item does not exceed the number in the total column.

# Unit Schedule

The amount of work can seem a little overwhelming, but you do not have to do everything at once, but you do have to work consistently.

| Week | Activities | Meetings | Major Tasks |
|---|---|---|---|
| 1 | Lecture 9- 3pm | Client | Analysis Report |
| 2 | Lecture 9-11am | Client<br>Management (Ltn) | |
| 3 | | Client<br>Management(Hbt) | |
| 4 | Lecture 11-1pm | Client<br>Management (Ltn) | |
| 5 | *Lecture 9-11am Monday (catch up lecture if required)* | Client | Design Report<br>Prototype Software |
| 6 | *Split Week for Easter* | Management<br>Client | |
| 7 | | | |
| 8 | | Client | |
| 9 | Lecture 9-11am | | Implement Release 1 |
| 10 | | Management<br>Client | |
| 11 | | | |
| 12 | | Client | Prepare Presentation |
| 13 | Presentation | | Presentation |
| 14 | Lecture 9-11am(Ltn)<br>Lecture 2-4pm (Hbt) | Client | Review Report<br>Testing Release 1<br>Implement Release 2 |
| 15 | | Management | |
| 16 | | | |
| 17 | | Client | |
| 18 | | | Implement Release 2<br>Webpage |
| 19 | | Management | |
| 20 | Lecture 9-11am (Ltn)<br>Lecture 2-4pm (Hbt)<br>*Double week* | Client | |
| 21 | | | |
| 22 | | Management | Prepare Demo Materials<br>Integrate & Test Software |
| 23 | | Client | |
| 24 | Demo Day 10am-2pm | | Manuals<br>Update Webpage<br>Final Project Report<br>Install/Integrate Software |
| 25 | | | |
| 26 | | Management<br>Client | |

You must have one team meeting each week to do work allocation and other team meetings each week to complete work products.

The proposed schedule for the lecture in week 1 is:

- 9am – Lecture – Introduction & Administration
- 10:30am – Workshop – Team Formation Exercises
- 11:30am – Workshop – Client Interviews
- 1pm – Lunch & Hand in Team nomination form
- 2pm – Workshop – Team Announcement & Meeting
- 3pm – Finish

**Hobart**

All the lectures, management meetings, presentations and Demonstration Day are on Wednesday.

**Launceston**

All the lectures, management meetings, presentations and Demonstration Day are on Monday.

### *Submission Schedule*

| Task | Date | Time |
|---|---|---|
| Timesheets | Weekly | Monday midnight |
| Schedule | Weekly | Monday midnight |
| Analysis Report | Week 5 | Monday 3pm |
| Design Report | Week 9 | Monday 3pm |
| Release 1 | Week 13 | At your presentation |
| Presentation | Week 13 | Monday/Wednesday |
| Review Report | Week 18 | Monday 3pm |
| Webpage v1 | Week 22 | Monday 3pm |
| Demonstration | Week 24 | Monday/Wednesday |
| Release 2 | Week 26 | At your MM |
| User Manual | Week 26 | Friday 3pm |
| Reference Manual | Week 26 | Friday 3pm |
| Final Report | Week 26 | Friday 3pm |
| Webpage v2 | Week 26 | Friday 3pm |

Peer assessment forms are due Thursday 3pm in weeks 5, 9, 13, 18, 21 and 27. You have an additional 72 hours to do the agree/disagree part of the ICR.

# Resources

### *Storage space*

There is storage space in \\alacritas\project_name (Hobart) or \\lawson\project_name (Launceston). The space should be available on Friday in week 1. A team can ask the lecturer for an increase to their quota, if and when needed. The technical manager should organise the folders inside your project folder as follows:

- Reports
  - Analysis Report
  - Design Report
  - Review Report
  - Final Project Report
- Manuals
  - User Manual
  - Reference Manual
- Marketing
- Downloads
- public_html
  - project (if necessary)
  - web
- Source
  - Release 1
  - Final Release
- Temp
- Submissions
  - Task Schedule
  - Week 5
  - Week 9
  - Week 13
    - Presentation
    - Release 1
      - Source
      - Executable
  - Week 18
  - Week 24
    - Marketing Materials
  - Week 26
    - Release 2
      - Source
      - Executable
    - Manuals
    - Final Project Report

If your project is a website then you will need the project folder in public_html. Source, Submissions, public_html, Reports, Manuals, Marketing, Downloads and Temp should be at the top level.

You can only place work relevant to the project inside your project folder.

### *E-mail*

E-mail is a very powerful communication tool that will be used a great deal by the lecturer. It is recommended that you check your e-mail every day.

### *LPS*

Each team can have an LPS account that you can all contribute to. You must share the cost of printing.

### *MyLO*

There are extensive resources for project on the MyLO website, including examples of all documentation.

# Appendix A – Timesheet

**Name:**
**Team:**
**Week number:**

| Date | Start | Stop | Interruption Time | Delta Time | Job Code | Comments |
|---|---|---|---|---|---|---|
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | **TOTAL TIME** | | | |

# Appendix B – Confidentiality agreement template

Some clients may require you to fill out a confidentiality agreement. If this is the case use the following template as a guide. The team and the client should change this to meet the requirements of the client. Before the team signs anything, it must have been shown to the lecturer.

This agreement dated the day of 2012 between the ICT Project Group [team name] and [client name].

[Client name] requires, and the confidants agree, that it is necessary to take the required steps to ensure that the confidential information is kept confidential.

NOW THE PARTIES AGREE AS FOLLOWS:

1. [Client name] has agreed to provide the ICT Project Group [team name] with access to information pertaining to [something].

2. That members of the project group do not hold any office, possess any property, or have an obligation by virtue of any contract that are, or might be created, in conflict with the information given under this agreement.

3. If during the duration of the agreement a risk of conflict of the nature referred to in Clause 2 arises they shall forthwith inform [client name].

DISCLOSURE OF INFORMATION/CONFIDENTIALITY

4.1 All information obtained from [client name] will be kept confidential until it is in the public domain or is deemed by [client name] not to be confidential. The team will not discuss such information outside the team without the proper written consent from [client name] with any person other than the School of Computing and Information Systems staff at the University of Tasmania, and shall keep any such information in their possession in a secure manner.

4.2 Before publishing material based on information gathered from [client name], the team agrees to consult with [client name] concerning the confidentiality of the information provided.

4.3 Confidential information includes:

Notes from interviews with [client name].

Code samples provided by [client name].

This list may be amended by an exchange of letters between the parties to the agreement. Such additions may not be made retrospectively. “Confidential Material” does not include information in the public domain, other than information that entered the public domain through a breach of this agreement and information that [client name] designates as no longer confidential.

COMMENCEMENT and CONCLUSION OF SERVICES

6.1 The parties agree that this agreement is to be taken as having commenced on the [date].

6.2 Any information provided under this agreement will be kept confidential for 3 years from the commencement of the agreement.

SIGNATURES

……
Client
TEAM MEMBERS

| | | | |
|---|---|---|---|
| …… | …… | …… | …… |
| Member 1 | Member 2 | Member 3 | Member 4 |
| …… | …… | …… | …… |
| Member 5 | Member 6 | Member 7 | Member 8 |

# Appendix C – Intellectual Property Agreement

The purpose of this agreement is to confirm the [specify the name & address of company] ("the Sponsor") commitment to sponsor the Project and to outline the conditions upon which the Sponsor has agreed to provide this support.

***1. Definitions***

**"Intellectual Property"** has the meaning given to it by the University's Intellectual Property Policy.

**"Project"** means the project outlined in the Project Brief.

**"Project Results"** means those results of the Project, which have or will be created as a result of the Project.

**"Student"** means [specify name and student number of student]

***2. Sponsor Obligations***

2.1. The Sponsor will provide in kind support of not less than 20 hours of technical supervision, guidance and support to assist the Student to undertake the Project.

2.2. [Optional if the Sponsor has agreed to pay the Student] The Sponsor will pay the Student $…. when the Sponsor is satisfied that the Project has been successfully completed.

2.3. The Sponsor agrees to permit the University to access the Project Results for the purposes of examining the Student.

***3. Intellectual Property***

3.1. The Student assigns to the Sponsor, absolutely, all of their Intellectual Property rights, existing now and in the future, in the Project Results (other than copyright in any thesis of the Student's based on the Project), throughout the world.

3.2. To the extent that the University contributes to the creation of the Project Results, the University agrees to assign to the Sponsor all Intellectual Property rights existing now or in the future in those Project Results.

3.3. The Sponsor agrees to grant the University and the Student a single non-distributable exclusive licence to use, modify or adapt the Project Results for non- commercial purposes.

***4. Publication***

The Student agrees to withhold publication of any documents relating to the Project (other than those for the purposes of examination specified in the ICT Project Manual 2012), until the written permission of the Sponsor is obtained, which shall not be unreasonably withheld.

***5. Entire Agreement***

This agreement and any documents referred to in this agreement or executed in connection with this agreement constitutes the entire

agreement of the parties in relation to its subject matter and supersedes all other representations, negotiations, arrangements, understandings or agreements and all other communications.

6. ***Governing Law***

   This agreement shall be governed by and construed in accordance with the laws of the State of Tasmania.

---

Signed on behalf of the University of Tasmania

……

Name Signature

……

Title/Position Date

---

Signed on behalf of [Client]

……

Name and Title/Position Signature

……

Business Name & Business Address

……

Home Address (if not Pty Ltd) Date

---

Signed by Student

……

Name and Student Number Signature

……

Address Date

In the presence of:

……

Name Signature Date