Usability Engineering Project

CSCI 3160 Fall 2014/15

2014-09-03 (1a)*

1 Overview

1.1 Description

For this project you will be part of a team of students that will design and implement a prototype of the user interface for a wide-ranging computer application. You must design a software interface. You may also design hardware interfaces.

In previous years, students have built systems for reading journal articles from computer screens, recipe management systems, and tools for keeping track of music collections. Your team may choose your own project (in consultation with me, your professor) *or* you may take my suggestion of a bibliography/citation management tool.

The project will consist of a set of phases. Each phase is important. Most phases will require you to submit an assignment, and will be graded separately.

All work you submit for grading must have a professional, finished appearance. When you need to submit rough drafts and handwritten records† they should be neatly organized. Phase assignments that are too short, rife with spelling errors or grammatically embarrassing will not receive passing grades.

1.2 Phases of Project and Schedule

In a real-world project the order of phases would likely not be so rigid, but because this is an educational exercise we are subject to different constraints than in the real-world. Below are the official due dates.

Milestone	**Deadline**	**Weighting‡**
User and Needs Analyses	23 Sept. (Tues.)	8%
Task Analysis	7 Oct. (Tues.)	20%
Design Document	16 Oct. (Thurs.)	20%
complete prototype		
Testing Strategy	22 Oct. (Wed.)	17%
begin testing & data collection§		
Demo. to professor and TA	29–30 Oct. (Wed. & Thurs.)	5%
Analysis of testing	20 Nov. (Thurs.)	17%
revise prototype		
Demo. of 'final' version to class	25–27 Nov. (Tues.–Thurs.)	7%
Final Portfolio	27 Nov. (Thurs.)	3%
Group Leader Reports		3%
Fairness Evaluation	02 Dec. (Tues.)	−50%
Project Assessment	8 Dec. (Mon.)	*optional*
Maximum *overall* bonus for appropriate innovation		+25%
Bonus is available for innovation that is appropriate to users and their needs/tasks		

Additional Notes

1. Deadlines are at the beginning of class (5:35 pm Tuesdays and Thursdays, and 11:05 am on Wednesdays) on the days when the class meets, and noon on the other days. Late work will be penalized as detailed in the syllabus.
2. The class meets on Tuesdays and Thursdays between Tuesday 10 September and Tuesday 03 December. There will be some tutorials on Wednesdays in that period too.
3. A test is scheduled for Thursday 23 October.
4. The calendar (on page 19) shows the above dates.
5. The Registrar's Office will schedule the exam for sometime between 05 and 16 December.

1.3 Project Homepage

Any updates to, resources for, and announcements about the project will be available from the project homepage on the WWW at ⟨URL:http://www.cs.dal.ca/~jamie/course/CS/3160/Project/⟩. Be sure to check it frequently. It includes a copy of this project description.

[*] This project document is a revision of the January 1999 version of the Bowling Green State University Computer Science (BGSU CS) 324 course project description, which was written by J. Blustein. The BGSU CS document was based on earlier work provided by Laura Leventhal. For some details about the BGSU CS course see *Julie Barnes and Laura Leventhal; Turning the tables: introducing software engineering concepts in a user interface design course; In Proceedings of the 32nd SIGCSE Technical Symposium on Computer Science Education, (pp. 214 – 218), 2001;* DOI:10.1145/364447.364587.

[†] You will need to submit handwritten notes, etc. as part of your analysis of testing and final portfolio.

[‡] The negative weight indicates the maximum deduction; See also the grading scale in Appendix A (on page 18).

[§] You must not collect data from users without prior approval from the Ethics Authority (see p. 13).

2 Groups

'If you want to go fast, go alone. If you want to go far, go together.'
— proverb quoted by Prof. Susan Holmes

Products with significant user interface components should be developed in groups. Although many of you are not experts in UI design or the problem domain, you will benefit from group work. Being able to work in a diverse team will make you more valuable to potential employers.

2.1 Rules for Groups and Group Dynamics

1. Each group member is expected to make an equal contribution to the project. All group members will receive the same grades for the phases of the project *except* in the most exceptional circumstances.
2. The products of your group should be high-quality and I expect for the group to produce better products than would result from individual work.
3. If you submit group work for grading by e-mail then all of the group members should be sent a `Cc` of the e-mail message. None of the individual reports (below) are group work.
4. ※ weekly log ↦ You will need to keep a weekly log of group activities for the final group work report (described below in §2.2.3).

2.2 Peer Assessments

Each student must submit

1. one group leader report (see §2.2.1),
2. self- and peer-evaluations for *every* phase (see §2.2.2), and
3. a final fairness evaluation (see §2.2.3).

None of those reports are group work. I prefer to receive these reports by e-mail in PDF or text format, but hardcopies submitted to the TA are also acceptable. These reports will be confidential between their authors and me (the professor).

We need your peer-assessments because they are the only way I can know what is happening in your group and act to help you improve or intervene to resolve problems[*].

I also use self- and peer-assessments to

- help students take more responsibility for their success and to be more active and engaged learners;
- give students a way to understand user centred design and software design more deeply;
- give students (especially senior undergraduates) more control over their work;
- produce better and larger projects than would be possible without groups;
- give students a chance to develop leadership skills; and
- develop skills that are important in the workplace.

[*] Sources: Schwartz/Ryerson, Holmes/Dalhousie (see page 21).

2.2.1 Group Leader Reports

The members of your group will take turns leading the group for phases of the project. You must all work together during each phase and part of the leader's responsibility will be to ensure that the work is shared equally. The group leader will also be responsible for coördinating meetings, activities, and documentation for that phase.

Where there are three members in a team, one of you will be the leader for the task analysis, another one will be a leader for the design document, and the third member will be leader for the testing strategy. There does not need to be a formal leader for the user & needs analysis. The title page of each project phase must identify the team leader (a simple way is to put an asterisk beside the name of the leader). ↩ title page

Where there are more than three members in a team, the others will lead for either the first demonstration or testing analysis phase. No team (for this project) should have more than five members.

Within 48 hours of completing their leadership phase the designated team leader *must* submit a teamwork report that:

1. lists all the team members and how each one contributed to the phase;
2. includes a ranking of the contributions with *no ties* (someone must have made the biggest contribution, and someone else the least);
3. shows how the leader made sure that the work was fairly divided amongst the team members.

2.2.2 Per Phase Group Work Evaluations

Within 24 hours of each phase of your project being completed you should submit a peer assessment of every member of your group (including yourself). These assessments are intended to help you and your teammates to work effectively and to alert me to potential problems that I could help you with before they become serious.

The form to use for these assessments is reproduced in the website.

Your grade for these reports will depend on the quality of your rationale. Be brief and to the point. Remember that you are assessing contributions not effort. No report should have more than one page of additional notes.

2.2.3 Final Group Work Reports aka Fairness Evaluations *(Required)*

After the final project is due all team members must submit their own group evaluation reports. Each team member will write their own report.

In the report you will tell me what you did in the project, what other members of your group did and how your team made sure that the work was divided fairly between all of you. You should keep a log of all the activities your group does each week to help you in writing the report.

The deadline is shown in §1.2 (on page 1); A few more details are in §10 (on page 18).

The Leader and Fairness reports are mandatory.
You cannot receive a passing grade for the project without submitting both of these reports.

3 User and Needs Analyses

To determine the tasks that the user needs to perform with your system's interface, you might interview potential users, evaluate existing software (e.g. shareware) or both. Note in your report if you did or did not interview potential users.

Because **you are not allowed to interview anyone from outside of the class** unless you have specific permission from one of the FCS CSBREB[*] or Dalhousie's SSHREB[†] **you are allowed to interview members of other groups so long as your interviewees do not also interview you**.

Your report will include two parts and must be double-spaced. Each part should be at least one page long, but the entire document should be less than six pages long[‡].

3.1 User Analysis

It would be best if your analysis was based on *two* groups of potential users of your system. Only one of those groups should be similar to students in this class. If you are not interviewing anyone from outside of the class then you will need to combine your imagination with some easily-gathered background facts.

Your report should include the following information for each of the groups:

1. Characterize the users as a group — who are they?
2. How will they use the system?
   - Who uses the system (will they do it themselves or will someone else do it for them)?
   - What benefit will the users get from the system (why should they use it)?
   - What is the environment in which it is used (e.g. an office, home, car)?
3. What skills do they have?
   - computer skills
   - skills in the domain of the system
4. What, if any, other important characteristics do the users have?
5. What interview questions did you use?
6. How did you choose these people to be interviewed?

Recommendations for *Interview-based* User Analyses

- To be meaningful you need to interview at least six people (i.e. three from each of two populations, five from each is even better).
- You should be able to show that the people you have chosen (your sample population) is representative of your target populations.
- None of your interviewees should be friends of yours, and you should not interview someone from another group that is also interviewing someone from your group.

[*] Faculty of Computer Science's Course-based Research Ethics Board

[†] Social Science and Humanities Research Ethics Board

[‡] Copies of forms, responses, etc. that appear in an appendix to the document will *not* contribute to the page count.

3.2 Needs Analysis

Your analysis should have the following three parts:

Goal a statement of the expected use of the system

Assumptions and Constraints

1. You will need to refer to these estimates later in the project for the risk analysis (§5.3). The estimates should depend largely on the expected value of a product that will fulfill the needs. The value is not necessarily commercial: it can also be the value to the potential users, and satisfaction for the project team for example.
   - what do you need to assume about the available hardware, information, etc. for the system to work;
   - roughly how much will it cost (in effort to build, price to buy, and resources to create and use, etc.); and
   - what world-view does this system relate to (e.g. if people are more efficient then the company will make more money, or that helping people to be fully satisfied with their jobs is our most important goal).

↩ estimate of resource constraints

2. It is much better to say what is needed than to jump to conclusion of how to satisfy the need(s). For example if you think that your implementation should use a website then explain why, perhaps because it can be used from any location, rather than specifying anything about how it will be implemented or presented to the users.

List of Features

- less than one page long; and
- specific things a user could, and would want to, do with the system.

3.3 Grading Principles

⊙ Grading form is in the website

Your analyses will be graded according to the grading scale in Appendix A (on page 18). It is particularly important that you describe the characteristics and needs of people from more than one population.

3.4 Readings for User & Needs Analyses

☛ For each phase of this project you should prepare yourself by studying the assigned readings in at least *one of* Heim's *The Resonant Interface*[5] or Stone et al.'s *User Interface Design and Evaluation*[20].

In *Heim*[5] Chapter 4 until §4.3 (pp. 101 – 119)

In *Stone et al.*[20] Chapters 2 and 3

(Optional) Related Readings

- §6.4 (Soft systems methodology) of *Dix et al. (1998, 2e)*[2];
- Chapter 2 (Thinking About Users) of *Hackos and Redish*[4];
- Chapter 2 (User Profiles) especially part of the final example (pp. 64 – 65) of *Mayhew*[13]; and
- In *Preece et al. (2002)*[15]:
  - §6.3, §7.1–§7.5, §13.1 & §13.2, and
  - Box 8.2 (pp. 251 – 252).

4 Cognitive Hierarchical Task Analysis (CHTA)

For this phase you will be performing a cognitive task analysis (CHTA) for the system for which you will later develop an interface. The point of the CHTA is to determine what users need (or want) to do to achieve their goals. In this phase you will produce at least the first four levels of a detailed hierarchical description of users' goals and sub-goals. The first level is one task name that describes the reason to use the whole system. The second level is a list of the major cognitive subtasks that users perform. At levels 3 and 4, the major subtasks are decomposed.

You should **not** be be making any design decisions in your decomposition. The task analysis is an early analysis activity in which requirements are gathered. It is one of the definitional phase in your project, therefore **feasibility** of implementation **should not be a consideration** in your task analysis.

The tasks must all be actions! For example, in the case of a cookbook/recipe management system, you might call a task 'Manage ingredients' rather than 'Ingredients'. Some of the tasks will have sub-tasks and others will be categories of tasks that are grouped together.

☛ **Do not include any tasks that are specific to any implementation.** See §4.5 for more details.

4.1 How to proceed

If you have not already identified users' goals through the user analysis and initial feature list, then you will need to do it before you can develop a complete task analysis[*]. To determine the tasks that users need to perform to fulfill their goals you may need to (re-)interview typical users and evaluate existing software. If you are working on a bibliography management tool or library catalogue then the CS librarian can show you some existing software programs and websites.

4.2 Notes

As you know, *goals* are what the users want (or need) to do and *tasks* are the steps that users need to do to achieve those goals.

Keep in mind that you can have more than one group of users (e.g. librarians, and library patrons).

4.2.1 Supporting users' goals

We use the name task analysis for historical reasons. A more apt name might be detailed description of users' goals and sub-goals. The point of the Cognitive Hierarchical Task Analysis (CHTA) is to determine what users need (or want) to do to achieve their goals. Only when you have described what users need to do are you ready to find ways to help them to achieve those goals. Tasks such as 'authenticate user with system' or 'sort records' are not related to goals but rather to a particular hypothetical system.

You should not limit your CHTA to describing tasks that you think can be carried out using software. The point of the CHTA is to determine what is needed and how the users will think about what is needed before you design something. If you limit yourself to only feasible things too early then you cannot free yourself to think of breakthrough designs, and you are likely to design a system with mediocre functionality and an interface that users must adapt themselves to, instead of a system with better functionality that supports users without forcing them to adapt themselves to software.

4.2.2 The user's cognitive and behavioural domains

Because the focus in this assignment is on cognitive tasks, the tasks that you describe will most often be decisions that the users will need to make.

The tasks that you describe *must* be what the user *wants* to do, not what the user might need to do, to work with some hypothetical system that you might create later. One of the purposes of the CHTA is to help you to think of the users' needs without consideration for what is feasible. Preparing a CHTA is a way for you to broaden your thinking, to consider possibilities that might not otherwise occur to you.

4.2.3 Task Analysis and Design

Do **not** think of the CHTA as **design**. It is an early analysis activity and, like all such activities, it is about requirements gathering to give you the information you need as source material for the design phase. Feasibility will be dealt with in phases that come after the CHTA.

[*] Your task analysis does *not* need to be for the same product you proposed in your Needs Analysis (§3.2).

4.3 Hand in

A graphical diagram (chart) and corresponding textual description are both required. Your chart should be in the style of one of (a) *Shepherd*’s[18] Figure 3.13 (on p. 54), (b) *Kirwan and Ainsworth*’s[7] Figure 3.14 (on p. 110) or (c) *Preece et al. (2002)*’s[15] Figure 7.13 (on p. 233). You may draw the diagram or chart by hand only if you are extremely neat, otherwise use a computerized drawing package.

For each task, give a written description of its function. Include the following details in tabular format:

Identification

- What is the name of this task?
- What is the goal of this task?

Location in Hierarchy

- What sub-tasks define this task?
- Is this task a subunit of a larger task?

Requirements (input) & Results (output)

- What kinds of inputs or actions does this task require from the user?
- What kind of ‘visible’[*] outputs or results occur when this task is performed?

Non-interface Aspects[†]

- What non-interface functions does this task require? or
- What automatic actions does this task expect from the system?

Other[†] • What special characteristics of this task should be recorded?

4.4 Approximate Grading Scheme[‡]

⊙ Grading form is in the website

Professional appearance *and* format	20%	
Clean and clear		
See §4.3		
Breadth of analysis	25%	≥ 15% required to pass
basic tasks and parts		≈ 15% for basics
non-essential tasks and parts		
Depth of analysis	25%	≥ 15% required to pass
accurate descriptions of tasks		
sufficient details in tasks		
at least 3 tasks need to be ≥ 4 levels deep[§]		
Organization of analysis	30%	> 20% required to pass
independent of implementation		
tasks relate to users’ goals not to system actions		
structure of tasks and subtasks reflect users’ goals		

If your task analysis is not acceptable then you will need to do it again before you can move on to the next phase[¶].

See §5.4 (on page 12)

[*] ‘Visibility’ in Don Norman’s terminology[14] means perceivable.

[†] Most tasks will have neither non-interface nor other aspects. The description for tasks that do not have such aspects should be ‘none’ or ‘N/A’.

[‡] See Appendix A, on page 18 for more details.

[§] The (unnumbered) top-level task is at level 1. Task 1.2.3, for example, would be at level 4.

[¶] The late penalty for work that must be done again may be reduced but it will not be eliminated.

4.5 Suggestions

Typical Errors to Avoid

- Lack of detail;
- Inconsistencies between the items in the chart and the written descriptions;
- Spelling and grammatical errors.
- Tasks which refer to implementation details rather than to the user's goals (e.g. selecting from a menu, sorting paper records or pop-up windows confirming data entry are all bad);

  > Every time you think of a task to include in the CHTA you should ask 'Why does this need to be a task? What need or desire of the user does it help to fulfill?' If the answer is that it will be a necessary step for the type of system that you think you will develop then you need to rethink that task. Tasks in the CHTA are steps users must take to achieve their goals *independent of any particular implementation*.
  >
  > It sometimes helps when planning tasks to think of two different implementations (notes on paper and a distributed database application for example) to decide if the task makes sense in both implementations, *but* you still need to be sure that doing the tasks in either way will support the user's goals. To carry the example further: you can sort notes on cue cards and you can sort records in a database, so sorting may seem like a sensible task. However when you ask yourself why the user will want to sort records, you will find that for most CHTAs the sorting to make something easier find — so the *real goal* is to find something (perhaps by recognizing it when they see it or perhaps by name or other attribute or datum). Similarly, users do not want to search but they might want to find; so 'find' could be a goal but 'search' would not be a goal.

- Describing a hierarchical menu structure not a hierarchical task analysis (writing the description from the programmer's perspective, rather than the user's perspective is bad);

  > Not all hierarchies are menus. If tasks can appear in more than one place you can label them as repeated and describe them in only one place in the chart. Remember that although the CHTA is hierarchical it is *not* a menu.

Best Practices

The best task analyses are easy to read and refer to. I suggest that you make sure that yours include these properties:

- the pages and tasks are numbered; and
- each of the tasks is described on one page, that is the task description is not split across a page boundary. Often three tasks can fit comfortably on one page.

There are many ways to make good hierarchical task decomposition charts however the best ones tend to have these properties:

- Anything that makes it easier to find the description that corresponds to the chart entries is good. For example, labeling tasks with numbers that correspond to the hierarchy in the chart can help, e.g. task 1.2.3 is part of subtask 1.2 which is part of task 1. Task 1 is part of the top-level task, which is not numbered.
- Where special symbols are used (e.g. ⋆ for tasks that are defined elsewhere, slashed corners for optional tasks, boxes that contain tasks to indicate that they must be performed in order, dashed and solid lines) there is a legend explaining their meaning.

4.6 Readings for Your Task Analysis[*]

In *Heim*[5] §4.3 especially §4.3.1 (pp. 119 – 124)

In *Stone et al.*[20] Chapters 2 and 3

In Your Handout

- A chapter about HTA by *Annett*[1]
- *Shepherd*[18] pp. 26 – 33, 54
  about task analysis framework, example on p. 54
- *Dix et al. (1998, 2e)*[2] pp. 261 – 268
  - §7.2 (Differences between task analysis and other techniques),
  - §7.3 (Task decomposition),
  - §7.4 (Knowledge-based analysis)
- *Dix et al. (2004, 3e)*[3] pp. 532 – 538
  §15.6 (Sources of information and data collection)

(Optional) Related Readings

- *Kirwan and Ainsworth*[7] describe hierarchical task analysis on pages 104 – 118 and describe a case study on pages 324 – 339.
- Chapter 7, particularly §7.6 and §7.7 of *Preece et al. (2002)*[15]

  > Note that you are required to prepare a more detailed and formal task analysis than is described in that textbook.

- §20.2 and §20.3 (pp. 413 – 419) of *Preece et al. (1994)*[16] give examples of a hierarchical task analyses.

  > If you follow the examples in *Preece et al. (1994)* then be wary of the implementation-specific parts at the bottom left of Figure 20.1 and the lower levels of Figure 20.2.

- *Mayhew*'s[13] Chapter 3 description of procedures for generating task analyses may be helpful.

[*] You need to prepare by studying the readings in *Heim* or *Stone et al.*

5 Design Document

The design document will be a record of the decisions you made about what the users will need to accomplish their tasks. As with all parts of the project, you may change it later but you must make a detailed record of your decisions so that you can refer to them at anytime. In a commercial venture the design document would also be used to communicate with clients.

The design document will be in two parts: a conceptual design and a concrete design.

5.1 Conceptual Design

The conceptual design is about the things that the user will use and the actions the user will take with those things. We call those things *objects*. The objects are very much like data structures in program code, but do not show any syntactic details of program code yet. I do want to see descriptions of the code that you would write but I do not want you encumbered by program syntax.

Your document must answer these questions:

- What things will the user work with?[*]
- How will the user get access to, and use, those things?[†]

5.1.1 Objects

You need to specify:

- What objects to use;
- What the properties of those objects are;
- How those objects and properties will be represented conceptually;
- How users will get access to those objects.

5.1.2 Actions (aka Operations)[‡]

We call the actions that users can do with your objects 'actions' or 'operations'. You need to specify:

- What are the necessary actions/operations;
- How will users select those actions/operations (through your system);
- How will users perform those actions/operations.

Remember that actions/operations are often about how a user can get to use instances of an object.

[*] *Hix & Hartson*[6] Table 1.1, p. 7 call these 'interaction components of the interface'.

[†] *Hix & Hartson*[6] loc. cit. would call these parts of the 'interface software (to support interaction)'.

[‡] *Heim*[5] and *Stone et al.*[20] use the term 'action' for what some other authors call 'operations'.

5.1.3 Presentation

You may present these details in any way that you think is meaningful. What matters is evidence that you have given sufficient thought to the design. Your conceptual design should be broad enough to cover everything and deep enough that there is clearly some way that it could be implemented.

I have found that some of the clearest presentations have used a hierarchical format such as those shown in the following two columns[*]:

- Object: `Recipe`
  - Components
    - Ingredients
      - Amounts
      - Name
    - Instructions
    - Yield
    - Comments
    - …
- Actions:
  - create
  - edit
  - delete
  - …

- Object: `Text`
  - Attributes:
    - contents
    - …
  - Instances:
    - Recipe's Title
    - Ingredient
    - Comment
    - …
- Actions:
  - create
  - edit
  - delete
  - …

Note that in the example above some objects have *components* (subparts) and others have *attributes* (properties). Objects with components have many parts. Objects with properties have many aspects or attributes, but these are not the same as parts[†].

When you refer to other other parts of your design in the written description use numeric identifiers and names. The numeric identifiers can be page numbers or might refer to a hierarchy with which you label your objects etc. just like you should have labeled tasks in your CHTA.

5.2 Concrete Design

In the concrete design you show how you think you will put your design into practice. This exposition is typically done with drawings or sketches that give the impression of the overall interface and some important parts in more detail. If you choose to do drawings or images from screen building software (e.g. Visual Basic or OmniGraffle) then you should include some text to annotate your figures.

Your interface should be original and support users doing the tasks you specified in your task analysis, and in the user analysis. Make your concrete design detailed but don't try to make it perfect. It needs to be detailed enough for use but if it looks too much like a finished product it can be too hard to change later. Hand-drawn designs are perfectly acceptable if they are legible. Remember iteration is the key — you will need to change your design later.

↩ Note: professional appearance

5.3 Risk Analysis

You must include a brief assessment of whether or not your design is consistent with the estimate of resource constraints from your previous Needs Analysis (§3.2 on page 5). If the design is not consistent with that earlier estimate or if it is for a different product than the Needs Analysis then prepare an updated estimate and rationale for the change. The entire risk analysis section should comprise fewer than 200 words.

[*] This example is specifically for a cookbook/recipe management system.

[†] See *Stone et al.*[20] §10.4.2, pp. 147 – 149 for more details about the differences between attributes and components.

5.4 Hand in

I need to see evidence that you have considered the implications of your design, used your (modified) task analysis, and are making good progress towards the final deadline.

Your design document must be *at least* 5 pages long and include both stages and a risk analysis. Hand in your (modified) task analysis along with the design document.

※acceptable CHTA ↦

If I told you that you had to modify your task analysis before moving on to the design document, *or* if you received a grade of $\mathcal{B}^-$ (or below) for the task analysis, then you *must* redo your task analysis before I will grade your design document.

5.5 Typical Errors to Avoid

DDs without CHTAs are incomplete

- Lack of detail;
- Spelling or grammatical errors;
- Inconsistency between the design and task analysis;
- Icons or command names that are not relevant to the user.

⊙ Grading form is in the website

5.6 Approximate Grading Scheme[*]

Extent of design	65%
coherence	
completeness (breadth and depth)	
Appropriate innovation	20%
appropriateness of interface to users and task	
originality of interface	
Overall quality of design	15%
quality of proposed interface	
functionality beyond basic requirements	
Lacking necessary parts	−75%
interface type and input/output devices	
concrete design (by description, diagram/chart or both)	
conceptual design objects	
task analysis and *sensible* risk analysis	
Lacking professional appearance[†]	−10%
clarity (includes numbered cross-references)	
legibility, spelling, and grammar	

5.7 Readings for Design Document Phase[‡]

Steve Dow. How prototyping practices affect design results. *interactions* v. 18 n. 3 (May+June 2011) pp. 54–59.
⟨URL: http://doi.acm.org/10.1145/1962438.1962451⟩.
(**Optional**: recorded lecture at ⟨URL:http://youtu.be/nrNSfuGacZw⟩, 1'5.12")

In *Heim*[5]

- §2.3 and Chapter 5 are most important
- Chapters 6, 10 – 14 are less important

In *Stone et al.*[20]

- Part 3 (Ch. 8 – 19) especially Ch. 8, 9, and 14
- Terms defined in §10.4.2, on pp. 147 – 149

(Optional) Related Readings

- Chapter 3 of *Lewis & Rieman*[8]
- Pages 236 – 238, and Chapter 8 of *Preece et al. (2002)*[15]
- Chapters 21 – 23 of *Preece et al. (1994)*[16] discuss some specific activities in the development of interface design documents. Chapter 22 is of particular interest.
- The terminology from *Hix & Hartson*[6] is described on pages 7, 132 – 144.

[*] Negative numbers indicate maximum deductions; See also Appendix A (on page 18).

[†] See note on page 11 about the standards of appearance for your *concrete* design.

[‡] You need to prepare by studying the reading (or video) by Dow **and** a reading in either *Heim* or *Stone et al.*

6 Testing Strategy

You can start this phase *without* feedback about your design document

In this phase your group will develop the documents and plans needed to evaluate your prototype system with actual users. You will need to submit your plans to me and to an outside Authority. I will grade your methodology and suggest changes; the Authority will decide if you are allowed to use your methodology. If you cannot show that you used an approved methodology then you will be penalized in the next phase.

You must not collect any data ***from users who are not enrolled in this class*** **without approval of** your methodology (including the consent form) by the **the Human Research Ethics authority**[*]. (You may however perform pilot testing since no data is being collected.) Exceptions are more clearly explained in §6.4.

↩ Ethics Authority

You will need to develop your testing methodology and documents before you can get your methodology approved by the Authority. On the course website you will find a (9-page) PDF form to use for your application to the Authority and a (7-page) PDF file including instructions and a template for a consent form. You will need to prepare an application and a consent form for submission to the Authority. Your consent form should be on Dalhousie letterhead but it will be acceptable to include a copy of the university's crest instead.

Submit your application to the ethics authority as soon as possible.
The authority's evaluation of your proposal is separate from your professor's grading.

6.1 Hand in

1. scenarios of activities for users to perform (these should be based on your task analysis and should include a mixture of representative and critical tasks);
2. a list of measures you plan to use in your testing (where the reason for a measure is not obvious you should include a justification for how it will be used later); and
3. a description of the protocol you will use when interacting with the users (the instructions you will give to users, the forms you will use to record the test, how you will perform the test, a copy of the consent form you will use, etc.).

☛ **Do not delay submitting your testing strategy** because your methodology and consent form has not been approved. You must have an approved methodology and consent form to conduct the testing but **I can give you useful feedback** even if they have not yet been approved.

6.2 Suggestions

- Plan to have all of your experimental participants perform tasks in the scenarios. You will need *at least* five participants plus pilot testers[9]. Serious testing of earliest prototypes requires at least eight participants.
  Plan to take careful notes of any comments, problems, and other noteworthy events that occur during testing. Plan to record what users do with the system as well as taking notes yourselves.
- Remember that if you use a paper prototype then one of your team will not be able to take any notes, so plan accordingly.
- Use a (modified) version of one of QUIS **or** SUMI to collect post hoc impressions. (There is a copy of part of QUIS[†] in *Shneiderman*[19] on pp. 136 – 143. Information about SUMI is available online at ⟨URL: http://www.ucc.ie/hfrg/questionnaires/sumi/⟩.)[‡]
  and the *System Usability Scale*.

  ⁂ You must use two methods
- Gary Perlman's *Web-Based User Interface Evaluation with Questionnaires* has references to other questionnaires. Note that privacy laws might not permit you to collect or store questionnaire data outside of Canada. However *Dalhousie's Online Survey Service* webpage describes a service that you might be allowed to use.

6.3 Optional

- Include scripts which describe what the user should do with the interface. These can be used to test if the interface works the way you think it should and also to ensure that users understand the interfaces parts.
- Plan to use a table with headings such as those below to help you record events during the testing.

Time	Problem or Issue	Effect on Performance

[*] Our Human Research Ethics authority is the Faculty of Computer Science's Course-based Research Ethics Board. To maintain the reviewers anonymity you should send submission to your professor to pass on to the reviewers.

[†] I have licensed a copy of QUIS for use in the Faculty of Computer Science. If you want to use it please ask me.

[‡] SUMI, QUIS, *Dalhousie's Online Survey Service*, and *Web-Based Interface . . .* URLs confirmed on 28 June 2013.

6.4 Exceptions to Ethics Requirements

The only exceptions to the requirements for prior approval of the Ethics Authority are that you are allowed to:

※pilot testing ↦

1. debug your testing strategy with potential users so long as you do not collect any data; this is called 'pilot testing'. If someone helped you with pilot testing then you should not use them for actual testing.
2. if time is running short you may test with other students in this class, but you will need to note which of your testers were from the class in the analysis document that you submit.

⊙ Grading form is in the website

6.5 Grading Principles

Your strategy and supporting documents will be evaluated using three criteria: (1) completeness of your plan (type of data to be collected, how data is to be collected, range of tasks used, etc.), (2) extent of preparation, and (3) professional appearance. A perfect strategy is one which shows me that you are prepared to begin serious testing immediately. Strategies which are disorganized, untidy or do not appear professional for other reasons will be penalized as much as one grade level.

Typical Errors to Avoid

- Lack of detail, specifically:
  - pilot testing not mentioned[*],
  - number of test participants not specified,
  - cognitive walkthroughs (for baseline performance) mentioned but no evidence of planning or completion included;
- Not planning to test enough of the system[10];
- Long documents without tables of contents;
- Inconsistency between description and forms;
- Recording made by hand and only by one person;
- Texts to be read to users that are in the passive voice.

6.6 Readings for the Testing Strategy Phase[†]

Highly Recommended Readings

- ⋆ Chapter 8 in *Heim*[5]
  or Part 4 (Ch. 20 – 27), especially Ch. 21 and 23 of *Stone et al.*[20]
- ⋆ *Maner*'s notes on formative evaluation[11]

(Optional) Related Readings

- ⋆ *Dix et al. (1998, 2e)*'s[2] Chapter 11 is an excellent overview of testing and evaluation.
- ⋆ *Maner*'s notes on scenario development[12] are highly recommended.
- Chapters 10 – 12, §13.3, and Chapter 14 of *Preece et al. (2002)*[15]
- Part VI of *Preece et al. (1994)*[16] discusses evaluation and testing. If you refer to that book then pay particular attention to Chapters 30 and 31.
- *Rubin*[17] has some very straightforward, practical advice.
- For initial testing and test design see §4.1 and §4.3 of *Lewis & Rieman*[8].
- For user testing see Chapter 5 until §5.6 (and Exercise 5.1) of *Lewis & Rieman*[8].

[*] You should plan for pilot testing, but I don't require it to be done by the time this plan is due.

[†] You need to prepare by studying the readings in *Heim* or *Stone et al.*

7 Demonstrations

Demo. of first version	**in class time**[*]
Demo. to class & professor	**during class**[*]

I will be examining your work to see how much you have accomplished by these times. The first demonstration is to the professor and grading assistant only.

7.1 Private demonstration of first version to professor

I will be examining your interface to see how much you have accomplished by this time, and to offer practical advice specifically to your group. To do that I will determine how many of the features you specified in your task analysis and described in your design document you have implemented, and how well they have been implemented. You progress will be graded according to the criteria in Appendix A (on page 18).

I am expecting (1) to see partly complete prototypes, (2) to see nearly complete testing strategies, and (3) for you to have clear ideas of what you will be doing next and expected dates of completion. However I will help you with *whatever* you have. Please bring *all* of your project documents and grading forms, and most especially your prototype and testing plans to the classroom.

If you need a computer for your prototype then please bring one of your own. I need to see how your prototype will operate in the test sessions. I will examine your documents, test the prototype myself and then ask you to run me through at least one of your test scenarios.

Please arrive a few minutes early (if you don't have a class just before) so that no demonstrations are delayed.

7.2 Demonstration of 'final' version to class and professor

The demo to the class is your chance to show everyone what you have accomplished and learned through the project. I expect you to

- briefly introduce the class to your problem domain before
- showing them your prototype,
- discussing what you learned through testing, and
- what changes you have made since then.

How you do this is entirely up to you. I only require that you introduce the problem domain at the start of your presentation.

Approximate Grading Scheme

⊙ Grading form is in the website

The grading scheme for your demo of your 'final' version will be approximately as follows:

Current implementation & Planned Improvements	
quality of implementation breadth of functionality completeness	65%
Insight into testing	
lessons learned from testing lessons learned about testing methods	15%
Appropriate innovation as defined in §5.6 (on page 12)	10%
Presentation quality	10%

[*] The order of demonstrations will be posted in advance.

8 Analysis of Testing Data

Based on your users' experiences with your system, give a detailed list of the improvements that you plan to make in the next version of your user interface. For each improvement, specify why your improvement should be made. You must address each item of user feedback.

You must **show that the Ethics Authority approved your testing procedure** to get a passing grade for this phase. If you have not already turned in proof that your methodology and consent form has been approved then you must include that proof with this report.

The **sole exception** is if your testing was only inspection methods (aka discount usability testing) in which no person from outside of the team participated in the testing. Groups of three or more students who use *only* those methods *with* prior permission from the professor cannot earn a grade above $\mathcal{B}^-$ for this phase. Groups who use only those methods without prior permission should not expect a grade above $\mathcal{D}$ for this phase.

8.1 Suggestions

A summary table with headings such as in the one below *might* help to organize the parts you will need.

Problem	Effect on Performance	Importance	Frequency of Occurrence	Proposed Solution	Cost to Fix	Resolution

8.2 Lessons Learned

Reflect on what you have learned through testing. I would like you to tell me: (1) what things you did in testing that were the most helpful, and (2) if you could do the testing again, how would you change your strategy?

8.3 Grading Principles

When grading your report I will be considering three factors: (1) the thoroughness of your analysis of the problems in your interface, (2) how specific you are about solutions, and (3) the rationality of your suggested changes.

In the best planned improvements documents every statement is justified, every comment and issue found through testing, demonstrations, etc. is addressed, and there is a clear rationale for all suggested changes to the design.

Typical Errors to Avoid

- not including page numbers, table of contents, etc. to make the structure of the document clear
- not including the (original) raw data or a copy of the the raw data

8.4 Readings for the Analysis of Testing Data Phase*

In *Heim*[5] Chapter 8

In *Stone et al.*[20] Chapters 25 and 28

(Optional) Related Readings

- *Dix et al. (1998, 2e)*'s[2] Chapter 11 has some good advice about analysing test data; and
- *Lewis & Rieman*[8] from §5.5.5 to the end of Chapter 5 is about evaluating data collected during testing.

*Additional readings may be assigned or provided by the professor. You need to prepare by studying the readings in *Heim* or *Stone et al.*

9 Final Project and Portfolio

Turn in:

- copy of your prototype
  - hardcopy of all non-code materials
  - *if* you wrote code: softcopy on CD-ROM of source and executable files
- a neat and well-organized portfolio of your project, including
  - all analyses,
  - user assessments,
  - planned improvements,
  - source code (*if applicable*),
  - signed consent forms (see the note below),
  - anything else to show how your project evolved, and
  - written list of the tasks that you identified in your task analysis and a description of which features/objects support your tasks and how the tasks were supported.

The **signed consent forms** must be included, but should be in a separate section *and* envelope so that they can be removed from the portfolio after grading.

9.1 Suggestions

I expect a detailed and well-organized document. The portfolio should be something that can be referred to repeatedly. Consider including a table of contents, indexing tabs or both.

9.2 Approximate Grading Scheme[*]

Current implementation	45%
breadth of functionality	
quality of implementation	
Planned improvements	40%
completeness	
appropriately detailed	
Appropriate[†] innovation	15%
interface	
functionality & automation	
(appropriate to users and task)	
Completeness of documents	−40%
Consistency of documents	
Professional appearance	−15%

[†] Negative weights indicate maximum deductions; See also the grading scale in Appendix A (on page 18).

[†] Innovation must be appropriate for the specified users and their specified tasks. See the earlier definition in §5.6 (on page 12), and §2.2 of *Lewis & Rieman*[8] as quoted in your lecture notes.

10 Fairness Evaluation (0% — Required, Max. deduction 50%)

Due at the start of class on Tuesday 03 December

It is essential that the group work is done fairly. Each of the group members must do this assignment on their own. Refer to your weekly log of group activities to help you complete this report. See item #4 of §2.1 (on page 2) for some details.

Every student must tell me (briefly):

1. What work each team member did to complete the project; and
2. What steps the team took to ensure that the work was divided fairly.
3. How fair they thought the overall division of project work was. In particular I need to know if anyone thought it was unfair.

11 Project Assessment (0% — Optional)

Please submit in the course mailbox or to the TA by noon on Tuesday 10 December

I will appreciate it if you would use a few minutes to write me briefly:

1. What you learned (about yourself, working in groups, and human-computer interaction) by working on this project; as well as
2. If you had do the project again, how would you do it differently?

☛ You may submit these 'lessons learned' anonymously. I want to know what you truly think about your experience with this project.

A Application of Dalhousie Grading Scale

From Dalhousie University Undergraduate Calendar Academic Regulations §17.1 (Grade Scale Definitions)[*]:

Grade	Definition	
$\mathcal{A}$-level	Excellent	Considerable evidence of original thinking; demonstrated outstanding capacity to analyze and synthesize; outstanding grasp of subject matter; evidence of extensive knowledge base.
$\mathcal{B}$-level	Good	Evidence of grasp of subject matter, some evidence of critical capacity and analytical ability; reasonable understanding of relevant issues; evidence of familiarity with the literature.
$\mathcal{C}$-level	Satisfactory	Evidence of some understanding of the subject matter; ability to develop solutions to simple problems; benefiting [sic] from his/her university experience.
$\mathcal{D}$	Marginal Pass	Evidence of minimally acceptable familiarity with subject matter, critical and analytical skills.
$\mathcal{F}$	Inadequate	Insufficient evidence of understanding of the subject matter; weakness in critical and analytical skills; limited or irrelevant use of the literature.

Grading Forms

Draft grading forms are available from the website.

[*] ⟨URL:http://www.registrar.dal.ca/calendar/ug/ACRG.htm⟩, retrieved 26 July 2009.

B Calendar (Fall 2014 – 2015)

September						
Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
		2	3	4		
		9	10	11		
		16	17	18		
		23	24	25		
		30				

October						
Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
			1	2		
		7	8	9		
		14	15	16		
		21	22	23		
		28	29	30		

November						
Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
		4	5	6		
		~~11~~	12	13		
		18	19	20		
		25	26	27		

December						
Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
		2			▨ 5	▨
▨	▨	▨	▨	▨	▨	▨
▨	▨	▨	▨ 17			

Notes

1. Dalhousie will be closed on 13 October and 11 November.
2. A test is scheduled for Thursday 23 October.
3. There will be no classes at Dalhousie on 12 November.
4. The Registrar's Office will schedule the exam for sometime between 05 and 17 December.
5. **Deadlines** are at the beginning of class (or tutorial) on Tuesdays, Wednesdays, and Thursdays, and noon otherwise.

References*

[1] John Annett. Hierarchical task analysis (HTA). In Neville Stanton, Alan Hedge, Karel Brookhuis, Eduardo Salas, and Hal Hendrick, editors, *Handbook of Human Factors and Ergonomic Methods*, Chapter 33 (pp. 33-1 – 33-7), CRC Press, 2005. ISBN 0-415-28700-6.

[2] Alan Dix, Janet Finlay, Gregory Abowd, and Russell Beale. *Human-Computer Interaction*. Prentice Hall Europe, second edition, 1998. ISBN 0-13-239864-8.

[3] ———. ———. Prentice Hall, third edition, 2004. ISBN 0-13-046109-1.

[4] JoAnn T. Hackos and Janice C. Redish. *User and Task Analysis for Interface Design*. John Wiley & Sons, 1998. ISBN 0-471-17831-4.

[5] Steven G. Heim. *The Resonant Interface: HCI foundations for interaction design*. Addison-Wesley, 2008. ISBN 978-0-321-37596-4.

[6] Deborah Hix and H. Rex Hartson. *Developing User Interface Ensuring Usability Through Product & Process*. Wiley Professional Computing. John Wiley & Sons, Inc., 1993. ISBN 0-471-57813-4.

[7] B. Kirwan and L.K. Ainsworth (editors). *A Guide To Task Analysis*. Taylor & Francis Inc., 1992. ISBN 0-7484-0057-5 (cloth) / 0-7484-0058-3 (paper).

[8] Clayton Lewis and John Rieman. Task-centered user interface design: A practical introduction, 1993, 1994. At ⟨URL:ftp://ftp.cs.colorado.edu⟩ (as flat files), ⟨URL:http://hcibib.org/tcuid/⟩ (as a webpage) & ⟨URL:http://www.cs.dal.ca/~jamie/TCUID/readme.html⟩ (in PDF).

[9] James R. Lewis. Sample sizes for usability tests: Mostly math, not magic. *interactions* v. 13 n. 6 (Nov. + Dec. 2006) pp. 29–33. ⟨URL: http://doi.acm.org/10.1145/1167948.1167973⟩.

[10] Gitte Lindgaard and Jarinee Chattratichart. Usability testing: what have we overlooked? In *Proceedings of the SIGCHI Conference on Human factors in Computing Systems*, (pp. 1415 – 1424), 2007; ⟨URL: http://doi.acm.org/10.1145/1240624.1240839⟩.

[11] Walt Maner. Formative usability evaluation, 15 March 1997. ⟨URL:http://csweb.cs.bgsu.edu/maner/domains/Formeval.htm⟩. Copy at ⟨URL:http://www.cs.dal.ca/~jamie/teach/WaltManer/Formeval.htm⟩.

[12] Walt Maner. Scenario method, 15 March 1997. ⟨URL:http://csweb.cs.bgsu.edu/maner/domains/Scenario.htm⟩. Copy at ⟨URL:http://www.cs.dal.ca/~jamie/teach/WaltManer/Scenario.htm⟩.

[13] Deborah J. Mayhew. *The Usability Engineering Lifecycle: A practitioner's handbook for user interface design*. Morgan Kaufmann Publishers, Inc., 1999. ISBN 1-55860-561-4.

[14] Donald A. Norman *The Psychology of Everyday Things*. Basic Books, 1988. ISBN 0-465-06709-3.

[15] Jenny Preece, Yvonne Rogers, and Helen Sharp. *Interaction Design: Beyond Human-Computer Interaction*. John Wiley & Sons, Inc., 2002. ISBN 0-471-49278-7.

[16] Jenny Preece, Yvonne Rogers, Helen Sharp, David Benyon, Simon Holland, and Tom Carey. *Human-Computer Interaction*. Prentice-Hall Europe, 1994. ISBN 0-201-62769-8.

[17] Jeffrey Rubin. *Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests*. John Wiley & Sons, Inc., 1994. ISBN 0-471-59403-2.

[18] Andrew Shepherd. *Hierarchical Task Analysis*. Taylor & Francis, 2001. ISBN 0-7484-0838-X.

[19] Ben Shneiderman. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley, third edition, 1998. ISBN 0-201-69497-2.

[20] Debbie Stone, Caroline Jarrett, Mark Woodroffe, and Shailey Minocha. *User Interface Design and Evaluation*. Morgan Kaufmann Series in Interactive Technologies. Elsevier, 2005. ISBN 0-12-088436-4.

[21] Susan Weinschenk, Pamela Jamar, and Sarah C. Yeo. Analysis (Chapter 2; pp. 13–49). In *GUI Design Essentials for Windows 95, Windows 3.1 World Wide Web*. Wiley Computer Publishing. John Wiley & Sons, Inc., 1997. ISBN 0-471-17549-8.

*The materials referenced here were used to create this project, and may help the student. See also §B, on page 21.

Sources used for groupwork (especially peer assessment)

Susan Holmes of Dalhousie University provided excellent advice about the design of every aspect of the teamwork part of this project.

I also found the following sources particularly helpful as I developed the peer assessment of groupwork process and form. Prof. Holmes helped refine the form.

- *Improving Teaching and Learning Group Work and Group Assessment* (2004 Edition) from the University Teaching Development Centre (UTDC) at Victoria University of Wellington.

  I found Appendix D (Group Member Contribution) especially helpful so I have based parts of my form on it.

- *Peer and Self Assessment of Student Work* Prepared by Michelle Schwartz, Research Associate, for the Learning & Teaching Office at Ryerson University.

  - Parts of the rationale are drawn from lists on pages 1 and 7.
  - I found the example of the *Indiana University's School of Medicine Peer/Self Assessment Program Self Assessment* form by Carolyn Hayes (which is described as being adapted from the University of Rochester School of Medicine and Dentistry 'Peer Assessment Program') so helpful that I use the assessment scale in my form and have based much of my form on it.

- *Assessment Matters: Self-Assessment and Peer Assessment Teaching Development* by Dorothy Spiller (February 2009), produced by the Teaching Development Unit at Waikato University.

- *Methods for assessing groupwork* from the University of Waterloo's Centre for Teaching Excellence at ⟨URL:https://uwaterloo.ca/centre-for-teaching-excellence/teaching-resources/teaching-tips/developing-assignments/group-work/methods-assessing-group-work⟩ (undated; last accessed 2013-09-10).

  I adapted a few of their examples of aspects of team functioning (e.g. 'generating ideas and solutions' and 'willingly taking on unpopular jobs') for my form.