Analytical Usability Evaluations of WYSIWYM

J. Masthoff (Ed.)

Autumn 2000

Table of Contents

1 Introduction

In autumn 2000, a group of third-year students attended a Usability Evaluation module at the University of Brighton. As part of the assessment, they had to perform an analytical usability evaluation. They performed their evaluation on two WYSIWYM-based systems, both of which were implemented at ITRI (the Information Technology Research Institute at the University of Brighton).

The first application, henceforth called DRAFTER, applies WYSIWYM to the authoring of documents (software user manuals, to be more precise). The students worked on a simplified version of DRAFTER that was specifically designed for evaluation purposes, and which offers its users a graphical as well as a textual mode of operation. For more documentation on DRAFTER, see R. Power and D. Scott (1998). Multilingual Authoring using Feedback Texts. In Proc. of COLING/ACL conference, Montreal.

The second application, called CLIME, allows people to use WYSIWYM to query a knowledge base of Maritime Law; WYSIWYM allows users to specify a type of ship, for example, and to ask questions about the insurance of that ship. For more documentation on CLIME, see Piwek, P., Evans, R., Cahill, L. & Tipper, N. (2000). Natural Language Generation in the MILE System. Proceedings of the IMPACTS in NLG Workshop, Schloss Dagstuhl, Germany, pp. 33-42. This report summarises their findings, and is intended mainly for the designers of the systems. The evaluations performed are *formative*, i.e., their intention is to help improve the systems. They are not summative: they do not give a usability ‘mark’.

I have used as much as possible the wording of the students. However, as English is often not their first language, I have sometimes reworded their sentences to make them easier to read. I have also put some more structure in, and removed irrelevant parts. It should be noted that it is likely that some comments of the students are incorrect: they may have overlooked functionality that is already in the systems. However, this provides information on what misconceptions a novice user may have.

The students involved: Evi Karanika, Xenofon Palamiotis, Alice Good, Julie Palmer, Nahid Miah, Sue Edwards, Ian Nicholls, Julien Fontaine, Pierre Marie Bennet, Jonas Tchetche, Inaki Paz, Almudena Jimenez, Nathalie Alexandre, Noelle Akil, Fazia Boudefoua, Ghada Fraiwat, Fortunatus Komanya, Phil Jones, Colin Adams, Paul Aquilanti, James Bonwick, and Andre Simmons.

2 Evaluation methods used

In this assignment, students were asked to perform an analytical usability evaluation, i.e., one that can be performed early in the design without needing real users (user testing is covered in their second assignment). They were given the freedom to choose which evaluation method (or combination of methods) they wanted to use. The following methods were used (and had been taught in class).

2.1 Cognitive Walkthrough

In a cognitive walkthrough, the evaluator pretends to be a novice user, someone who encounters the system for the first time.

In order to perform a cognitive walkthrough, the evaluator needs:

- a description of a typical novice user (what kind of computer experience, what kind of background knowledge, etc.)
- a typical task the user might like to perform with the system (needs to be very specific, “making a bit of user manual” is not specific enough)
- a (detailed) correct action sequence on how to perform that task with the system
- a description of the interface (or in this case, the system itself).

For each step in the correct action sequence, the evaluator will ask the following questions:

- Will the user expect to have to take this action?
- Will the user notice the control for the action?
- Once users find the control, will they recognise that it produces the desired effect?
- If the correct action is performed, will progress be apparent?

It should be noted that a Cognitive Walkthrough only looks at learnability (how easy is it for a complete novice to operate the system). It does not look at efficiency or effectiveness. As only the *correct* action sequence is followed, it will also not find problems with error messages.

2.2 Heuristic Evaluation

In a heuristic evaluation, a small set of evaluators (people with usability expertise) judge how well a system complies with recognized usability principles (the ‘heuristics’). First, each evaluator independently examines the interface. Next, the results of the individual evaluators are combined. Many sets of heuristics exist, varying in detail and specificity for a certain platform.

2.2.1 Nielsen’s heuristics

Nielsen has condensed his heuristics into a set of only ten, to make it easier to perform a heuristic evaluation:

Visibility of system status
The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.
Match between system and the real world

The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.

User control and freedom

Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

Consistency and standards

Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

Error prevention

Even better than good error messages is a careful design which prevents a problem from occurring in the first place.

Recognition rather than recall

Make objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

Flexibility and efficiency of use

Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

Aesthetic and minimalist design

Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

Help users recognize, diagnose, and recover from errors

Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

Help and documentation

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

2.2.2 Accessibility heuristics

There is a growing emphasis on making sure that systems are accessible by everyone, also by people with disabilities. The Disability Act and the Human Rights Act provide legal reasons for doing this, in addition to the obvious humanitarian reasons. Heuristics exists for making systems accessible (for instance, making sure that someone who is blind can use the system by combining it with a text-to-speech reader). See, for instance, the IBM (http://www-3.ibm.com/able/accessweb.html) and the W3C (http://www.w3.org/TR/WCAG10/) accessibility guidelines.

3 DRAFTER

Five groups of students evaluated DRAFTER. The groups varied in size between two and four students. DRAFTER was made available to them on three PCs (running Linux) in the Computer Suite. They were provided with an A4 detailing how to login to the system. In addition, they were referred to the WYSIWYM website, which contains an explanation of the goal of WYSIWYM and its interface. Unfortunately, the PCs in the Computer Suite do not have a soundcard, so, most students did not manage to see the on-line demonstration of WYSIWYM. The students could ask their lecturer for further explanation (as most did), and (on their own initiative) some students spoke to Richard Powell, one of the WYSIWYM designers.

3.1 Cognitive Walkthrough

Users: English user manual developers, with standard computer experience (used to applications like Word and the internet).

Note that, as part of their assignment, students were asked to formulate a suitable task description. In some cases, they used task descriptions that do not match the system's capabilities. (For example, the task description refers to Ms Word, whereas the system is, in fact, unable to handle Ms Word.) Usually, however, this does not invalidate most of their comments.

3.1.1 Group4: Text mode

Typical task description:
Make a bit of user manual which explains how to open an Ms Word Application, namely by moving the mouse to the MsWord icon and double clicking on it.

Correct action sequence using the textual mode:

1. Click Do This Action
2. Select Open from the list
3. Click This Object and select Window
4. Click Label and select Ms Word
5. Click Using these methods
6. Select Method components
7. Click Do this Action
8. Select Move
9. Click This Object and select Icon
10. Click Label and select Ms Word
11. Click Next Step
12. Click Do This Action and select Double-Click
13. Click This Object and select Icon
14. Click Label and select MsWord

Problems found:

- Action 3. When you click on Object, you expect to have to select Application or TextEditor or Word, but that is not possible. How do you know that you have to select Window to continue working and progressing?
- Action 4. What would happen if the Application were Excel instead of MsWord? You cannot select a label that is not in the list.
- Action 5. When clicking on “Using these methods” you do not expect a list with only one item (Method components). You would expect “Next Step” or a list with different method types.
- Action 7. You are pretty lost between all the things you can click on now. The only thing that may guide you is text colour.
- Action 11. “Using this method” and “Next method” phrases may disconcert the user.

3.1.2 Group4: Graphics mode

Typical task description: Make a bit of user manual which explains how to click on the OK button, namely by moving the mouse to the MsWord icon and double clicking on it.

Correct Action Sequence using the graphical mode:

1. Click Goal: undef and select Click from the list
2. Click On: undef and select Button
3. Click Label: undef and select OK
4. Click Methods: undef and select Method components
5. Click Goal: undef and select Move
6. Click To: undef and select Button
7. Click Label: undef and select OK
8. Click Item2: undef and select Task components
9. Click Goal: undef and select Use
10. Click Actee: undef and select Mouse button

Problems found:

- Action 2. The system shows two options now: Actee and On, and it is not clear to the user which to click. The words are not very clear. The user may (correctly) think that they must click on “On” because that is red, but they may have doubts.
- Action 5: The system shows a quite complicated graphic: boxes inside other boxes, so the user is now a bit lost.
- Action 8: The user does not know where to click to specify the next step, because both steps and methods are called with the same name, itemX.

3.1.3 Group7: Text mode

Typical task description: Enter information into the knowledge base to describe the saving of a file (as in the WYSIWYM demo on the internet). The information has to specify opening the “Save As” dialog box, entering the name of the file in the “Save As” field, and clicking the OK button.

Correct action sequence:
1. Left click and hold on the text “do this action”, then select “Save” from the menu.
2. Left click and hold on the text “this data”, then select “File” from the menu.
3. Left click and hold on the text “Using these methods”, then select “Methods components” from the menu.
4. Left click and hold on the text “do this action”, then select “Enter” from the menu.
5. Left click and hold on the text “this data”, then select “Name” from the menu.
6. Left click and hold on the text “the file”, then select “Copy” from the menu.
7. Left click and hold on the text “this data”, then select “Paste” from the menu.
8. Left click and hold on the text “this object”, then select “Field” from the menu.
9. Left click and hold on the text “label”, then select “Save as” from the menu.
10. Left click and hold on the text “perform this task”, then select “Task Components” from the menu.
11. Left click and hold on the text “do this action”, then select “Open” from the menu.
12. Left click and hold on the text “this object”, then select “Dialogue Box” from the menu.
13. Left click and hold on the text “label”, then select “Save As” from the menu.
14. Left click and hold on the text “next step”, then select “Task Components” from the menu.
15. Left click and hold on the text “do this action”, then select “Click” from the menu.
16. Left click and hold on the text “this mouse”, then select “Mouse” from the menu.
17. Left click and hold on the text “this object”, then select “Button” from the menu.
18. Left click and hold on the text “label”, then select “OK” from the menu.

Problems found:
- Action 1. The words are written in red, which indicates that the text is an anchor, which needs to be specified. A new user may not realise that this is what red means. A possible solution may be to have some sort of key visible on the screen to indicate what the colours mean. When the program is first started up, “do this action” could be modified into “to complete the goal do this action”.
- Action 2. When the user sees the text “save this data”, they may think that “this data” refers to the data that is to be saved inside the file, and not the anchor where they specify the file itself.
- Action 3. The term “Method components” is unclear and may require a new name.
- Action 6. The user will not know the underlying object model of the system. When they are dealing with natural language text, they may not realise that they need to copy and paste objects into other parts of the text to use the object references, instead of creating new objects. For example in this action step, we need to copy the file anchor and then paste it in order to use it elsewhere. If it was not copied, the text would say “another file” referring to a new file object.
- Actions 9, 13, and 18. The anchor “Label” may be a bit confusing. It could be changed to “.. with this name”.

- Action 10. The term “Task components” is unclear and may require a new name. Alternatively, change the anchor “Next step” into a “insert new task” button.
- Action 10. The user must enter opening the dialogue box as a precondition. They may not realise they need to do this, and instead try to do it as the main task.
- Action 16. To a new user the anchor “this mouse” is confusing: it seems to imply that mouse has already been specified.

3.1.4 Group7: graphical mode

The same typical task was used as in the text mode (see 3.1.3).

Problems found (in addition to the ones already mentioned under text mode):

- The text “undef” may not be clear to new users. Perhaps it is possible to expand it to “undefined” (if this would not take up to much space).
- The term “actee” is unclear. Perhaps it could be renamed to “subject” or something similar.

3.1.5 Group6

Typical task description:
To create instructions to show a user how to save a file producing the following text:
To save the file: Choose the File option from the toolbar by moving the mouse pointer to the toolbar and clicking on the File button. Choose the Save option from the file scrolling list by clicking on the Save icon. Choose the OK button from the Save dialogue box.

Correct action sequence: (LCH = Left mouse click and hold)
1. LCH Do this action
2. Select Save
3. LCH This data
4. Select File
5. LCH Using these methods
6. Select Method components
7. LCH Do this action
8. Select Choose
9. LCH This object
10. Select Option
11. LCH Label
12. Select File
13. LCH This object
14. Select Toolbar

Etc etc. (57 action steps mentioned)

These students made it difficult for themselves by selecting such a complex task. They also made the action steps very fine grained (it should be noted that the students from

Group7 combined two of these steps into one). It is better to start with fine grained steps, but given that there is quite some repetition in how you operate DRAFTER, later steps could be combined (or omitted in the analysis). The students became disheartened when looking at the long list of action steps, and tried to be smart.

Group6 decided that on an abstract level there are two main types of actions to take in DRAFTER: changing an "action" and changing a "method". Somehow they got confused:

- they thought that changing an action always involves clicking on red text, and that red text always is "do this action".
- they thought that changing a method always involved clicking on green text.

However, they also mentioned that red was required, and green optional.

The only action steps Group6 looked at were their two abstract ones, which made it very difficult to find specific problems (though they found many general ones).
Problems found:

- One of the major problems to be found was the lack of prompting. The user is presented with the start up screen and there is no indication of where to start or what to do. We confused the drop-down menu provided on mouse clicks with commands (as in many Window programs) rather than mere words to change the text with. This is made worse by the nature of the domain (writing computer manuals) as that means that many words could be used in both instances (for example, does "save" in the dropdown menu change the text, or actually save the document?)

- A fundamental weakness of the system is that it requires the users to structure their information into goals, actions and methods before attempting to capture it. For a novice that might mean an intermediate step of writing this down, in much the same way as a knowledge engineer might do. We found that with practice, possibly because of the limited nature of the domain, this intellectual skill was soon internalised allowing the more experienced to code directly into the machine.

  The style choice of Graphical emphasises this cognitive overload, while the choice of Text, with its marked similarity to the interface provided by Word minimises the problem. The presentation selection of modality, again with its similarity to Word e.g. View/Page Outline to show controls, also helps.

- There is no indication that the user selects a fundamental concept (rather than a word) and that the action of the system may well be different depending on which action concept was chosen, i.e., how the template develops will in part depend on whether "save" or "schedule" was selected.

- When first compiling the correct action sequence, the fact that green text was to a certain extent optional was not clear to us. Thus it was puzzling that no matter how well we defined something, it kept wanting more! The "using these methods" was always added to an action, and while this allows for everything to be specified, it does not have a set limit and we did not know how far we should go – indeed at one point we thought we might have to describe how to make your brain move your finger, to click the mouse, to click the file menu! We soon realised that it was up to you to decide a cut off point.

- There appears to have been no effort made to design-in assistance for the novice to learn at this stage of the demo. It might help to:
  - Have a sample available on-line (for example, when we learn to program we start with the "Hello world" example)
  - Incorporate a split screen option to show feedback text and output text at the same time.
  - Provide screen level and field level help text.

3.1.6 Group3

This group tried to look at all the tasks a user may want to do with the system. Unfortunately they concentrated on lower-level tasks like "Start new file", "Change style", and "Change modality". The only real user task they looked at was much too simple, namely making a bit of manual with only "open the file" in it.

Problems found:
- Both in the style menu and the modality menu, the currently selected option is not indicated.
- The user will not expect to have to click on the text.

3.1.7 Group1

Typical task description: Producing instructions on how to save a new document in MS Office.

The group looked at both the textual and graphical interface. They did not find any problems (except for functional problems), but this is due to a very bad understanding of how to perform the walkthrough.

3.2 Heuristic Evaluation

3.2.1 Visibility of system status

The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.

- In the feedback text, the system uses a black colour indicating to the user what the final result will be. Sometimes, however, there are black words that do not belong to the output. This can lead to confusion. The system should show those words in a different colour. (Group4).

- The system only allows you to specify four methods with four steps as a maximum and the user only realises this when he reaches the end (Group4, Group1). Let the users create as many as they want. (Group4)
- In the menus for modality and style of text, it is not indicated what the current modality/style is. Placing a tick next to the currently selected option would be a simple way of signalling system status. (Group3).
- DRAFTER simply doesn't keep the user informed enough, e.g., selection of particular 'action concepts' will change the behaviour of the tool (Group6).
- DRAFTER passed this test. The feedback to a user action is instant. The text is changed to reflect the knowledge of the system. The whole state of the system can be seen at once. Apart from scrolling the client area of the window, everything is on the one window, the user does not need to navigate between several windows (Group7).

3.2.2 Match between system and the real world

The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.

- There is no "exit", you must "cancel", that has a different meaning in the real world, so the system is not using the same words as the user (Group4).
- The information doesn't always appear in a logical order (Group1).
  When you specify sub-stages they can appear like this:

Feedback Modality:

To close the MacWrite window:
¶ Before starting, perform this task
  Then proceed as follows:
    1 Move the mouse pointer to the toolbar.
    2 Click the mouse on the Cancel button.
    3 Next step.
  To quit, perform this task.
¶ Next method.

To click the mouse on the Cancel button:
¶ Before starting, perform this task
  Then proceed as follows:
    1 Use the mouse pointer by using these methods.
    2 Use the mouse button by using these methods.
    3 Next step.
  To quit, release the mouse by using these methods.
¶ Next method.

To move the mouse pointer to the toolbar:
¶ Before starting, perform this task
  Then proceed as follows:
    1 Use the mouse by using these methods.
    2 Next step.

Output Modality:

To Close the MacWrite window:
  1 Move the mouse pointer to the toolbar.
  2 Click the mouse on the Cancel button.

To click the mouse on the Cancel button:
  1 Use the mouse pointer.
  2 Use the mouse button.
To quit, release the mouse.

To move the mouse pointer to the toolbar:
Use the mouse.

To quit, perform this task.
¶ Next method.

They should appear in the same order as they are shown in the first method description. (Group4).

- The system requires the user to think in structured terms such as Action and Methods. However, the user is merely trying to write a software instruction manual, he is not a knowledge engineer (Group6).
- The system passed this test. The system will be tailored according to the user, hence it will use terminology that the specific user will understand. The system does not use words, but a set of concepts that are matched to words, so the system can work in different languages. We did, however, find some things which we considered as problems with the language. The anchor "the **label** dialogue box" might be easier for a new user to understand if it was something like "the dialogue box **with the name**". The graphical view is slightly harder to read than the textual view. (Group7).

3.2.3 User control and freedom

Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

- The user cannot undo an undo: if the user cuts, there is no way back. The user can use copy and paste, but not cut and paste. This can lead to unwanted errors. (Group4) With this system, once you have cut data it is lost, there is no REDO function in the demo. The Cut function is flexible and directive, it is not however iterative. (Group6) There is no facility to redo or undo commands that were wrongly entered. The only way the user can recover from mistakes is to cut the text wrongly added. (Group7, Group1).
- If the user has different methods for one objective, they can only delete the last one, so if the user wants to delete (cut) the first one, they must delete all of them, losing all the other methods. This makes the user feel frustrated, as they must delete all the work they have done. Let the user cut any method independent of where it is. (Group4)
- When you have a feedback text larger than the window, if you make changes to the lowest part, you go to the upper part of the feedback text, losing the focus from the part you were working in. Leave the focus where the change was made, or near to it. (Group4)
- You never know how the system will make the output, sometimes the steps appear in the same line and sometimes they appear numbered. You don't know a priori what output the system will give. This means that the user loses control. The system should tell the user how the output will be or always use the same pattern. (Group4)
- The colours (red and green) used in the interface proved to be a problem for one member of the group who is colour blind. Giving the user the freedom to set the colours the text is displayed in (perhaps from a colour palette) could overcome this problem. (Group3)

- The interface does not offer any options for the user to change the size of the text or graphics that they are viewing. For example, in the graphical view the hierarchy can take up a lot of room on the screen meaning that the user has to scroll across the screen. If it were possible for the user to zoom in or out by clicking the mouse buttons, this could lead to a quicker way of viewing. In addition to this, if the user has problems reading the size of the text in text mode they could increase the size of what they are trying to read. (Group3)
- The system doesn't appear to offer a facility to focus to the area of interest, e.g. drill-down and expand. (Group6)
- Provide the option for a user to change the fonts, text size and interface colours, if they have any problems reading the text or diagrams (Group7)
- Provide the option of a split screen so the user can see both the feedback and the output at the same time. (Group7 and Group6).

3.2.4 Consistency and standards

Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

- If we click on the "File" field on the toolbar, we can find two options ("New" and "Quit"). Both do exactly the same, namely logging-out (Group4).
- The system does not give the user the opportunity to open an existing user manual, save their work or print it (Group4).
- If the user works in the textual mode, they can distinguish between steps and methods: steps are preceded by numbers, and methods by the symbol "¶". But in the graphical mode, both steps and methods are preceded by "ItemX" (where X is the step number or the method number). This can confuse the user. Use different words, perhaps 'item' for methods and 'step' for tasks. (Group4).
- DRAFTER's interface is processed in a manner similar to Word. However, when manipulating text do you expect to use the mouse? (Group6).
- Sometimes the menu options on the anchors are confusing. For example, left clicking on the anchor 'Next step' gives the option 'Task components'. Right clicking will give a menu option 'Next step', but will then give the same option as left click. (Group7).

3.2.5 Error prevention

Even better than good error messages is a careful design which prevents a problem from occurring in the first place.

- The system does not provide messages that could prevent errors, for instance, when the user cuts some text, the system doesn't inform them that the cut text cannot be recovered. (Group4, Group1)
- The colour-blindness colour could be prevented also by tagging red text with a number 1 and green text with a number 2. This would also make the differences in text type visible on a monochrome monitor. (Group3).

- With DRAFTER you can lose all your work. Whilst it is obviously a demo, there ought to be a confirm exit screen. (Group6)
- If the user wants to refer to the same object in a different anchor, the only correct operation is to copy the object, for example a file, and paste it into the second anchor. A new user of the system may instead insert a new file object, instead of copying and pasting the other file reference, because it is not an obvious operation to perform. (Group7)

3.2.6 Recognition rather than recall

Make objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

- There are no instructions for use in this system (Group4).
  Visible instruction is noticeably absent (Group6).
  Instructions are not available nor tool tips (Group7).
- Recognition rather than recall: It can be argued that DRAFTER's colour coded template is quite good in this regard. (Group6)
- There is visible recognition of text using colours. Normal text is displayed black, obligatory text anchors in red, and optional text anchors are displayed in green. One problem with the choice of colours is that red and green are not suitable for people who are colour blind. (Group7)

3.2.7 Flexibility and efficiency of use

Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

- You do not have all the options you need to conquer one objective. You can only use the options provided. We can see this problem in all these examples:
  - When you click "this object" to select an application, the only thing suitable is "window", application does not appear in the list.
  - You cannot specify as an action to search something on the screen, or to take the mouse.
  - You cannot express an if condition if an action goes wrong, to tell the user what to do then. You can only put in another method for getting to the objective.
  - You cannot express the relative position of an object on the screen or in the owner window, like upper-right.
  - You cannot add new commands or actions to use in your manual, just as with objects, As an example: select from a list, search on the screen, etc.

  All these examples show the lack of flexibility of the system. DRAFTER should incorporate a feature to add new objects etc to the system. It should also incorporate an if statement as an action (Group4)

- No shortcut keys are available. (Group1, Group7)
  A new user may try to cut and past previous actions if they wanted to repeat an action, but these would then refer to the same objects in the underlying object model.
  The users can not tailor frequent actions. (Group7).

3.2.8 Aesthetic and minimalist design

Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

- Hardly any information is provided by the system, so no problems with irrelevant information (Group4).
- A trivial pass. Everything is clearly visible, and there is no irrelevant information on the screen (Group7).

3.2.9 Help users recognize, diagnose, and recover from errors

Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

- There are no error messages in the system. For instance, the user can write: "Click on the name of the name of the name of the name of.." without an error message. (Group4).
  No error trapping (Group6).
  We considered this to be a failure. No warning or error messages are given to the user. (Group7).

3.2.10 Help and documentation

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

- No help or documentation provided as part of the system. (Group4, Group6, Group7).
  Possibly provide some examples or a simple tutorial. (Group7)
- Help the user by showing a help message when the mouse is over a command (anchor). (Group4)

3.2.11 Accessibility

Group 7 also performed a heuristic evaluation using IBM's accessibility heuristics http://www-3.ibm.com/able/accessweb.html)

Problems found:

- DARFTER does not provide shortcut keys, making it impossible to use for a person who (due to some disability) cannot use a mouse.
- DRAFTER uses colour not only as an enhancement, but to convey information on types of anchor. This is strongly discouraged in the accessibility guidelines. Possibly have different fonts for anchors rather than (or in addition to) different colours.

3.3 Positive comments

- We appreciate that once the user has familarised themselves with the tool it appears to provide an efficient and flexible method for recording computer manual information in a structured manner. (Group6)
- The style choice of Graphical emphasises the cognitive overload, while the choice of Text, with its marked similarity to the interface provided by Word minimises the problem. The presentation selection of modality, again with its similarity to Word e.g. View/Page Outline to show controls, also helps. (Group6)
- We found the idea of implementing knowledge using common phrases was a very good idea. It has similarities with touch screen devices that are in general use where a user inputs knowledge into a system using pictures of common phrases, e.g., information sites in travel agents. It allows the input of knowledge without needing to know how the system works underneath and reduces the need for typed input from the user, which could be input incorrectly. (Group3)

4 CLIME

Three groups evaluated CLIME: a group of one student, a group of two students, and a group of four students. CLIME (version…) was installed on a PC in room 209, a small room in the Computer Suite that can be booked by students. The students were given some information on how to start CLIME (see Appendix), known bugs in CLIME, and were referred to the CLIME tutorial (under Help in the system).

4.1 Cognitive Walkthrough

Typical users: Maritime legal experts, requiring legal information applicable to a particular ship. They are familiar with standard computer applications like Word and the internet.

4.1.1 Group2

Typical task description: Make a query on what the scope of an annual survey is for an oil tanker, and read the answer.

Correct action sequence:
1. Double click on template
2. Click on survey_question
3. Click on query menu, select edit current

4. Click on ship
5. Type in first two letters of oil tanker in text box
6. Select oil tanker
7. Click on a survey
8. Select annual survey
9. From file menu select Submit as
10. Type in filename and click save in dialogue window
11. Select your query file from the personal folder
12. Select Marpol AIP2 18.01

Problems found:

- Action 2: The user may not be sure what type of question they want to ask (so, which template to select).
- Action 2: When clicking on survey question no progress in apparent, except from it being highlighted.
- Action 3: The user will not expect to have to take this action, and will have difficulty finding the control in the menus.
- Actions 4 and 7: The user will not expect to have to take this action. The text does not appear clickable. Add an instruction advising the user that they can change the green text by clicking and selecting from a list.
- Action 5: The user will not expect to have to take this action. Add an instruction to tell the user to enter the first letters.
- Action 9: After the query is complete, the user is not advised what the next step may be. The Submit-as functionality is not easy to find, a button with Search or Go will be more noticeable.
- Action 10: The user will not expect to have to take this action. Prompt needed to advise users to type in the filename.
- Action 11: The user will not expect to have to take this action. Instruction needed to tell the user to select the query to retrieve the answer.

4.1.2 Group8

Typical task description: Construct and submit a query on the requirements for a fishing vessel with a bilge pump.

Correct action sequence:

1. Double click on Templates from the menu
2. Double click on Requirements_Query from the expanded submenu
3. Select Query->Edit->Current from the main menu
4. Click on the word Ship on the mile query window
5. Scroll down the list and select ‘fishing vessel’
6. Select the red word saying ‘Some states’ and from the popup menu select ‘fitted with’
7. Click on the text saying ‘fishing vessel’
8. Click the copy icon (middle selection)

9. Click on the ‘ship’ text
10. Click on the paste button (rightmost selection)
11. Click on the text ‘an equipment’
12. Enter ‘bi’ in the text box and then select ‘bilge pump’
13. In the browser window click on ‘Personal folder’
14. In the query window select ‘Save as’ on the file menu
15. Enter the name ‘MyQuery1’
16. Select Save in the dialogue window
17. Select submit from the file menu

Problems found (only forms are filled in for action 4, 8, and 17):

- Action 4: The user will not expect to have to take this action. They will not know that they can click it and select from a list just because the word is green.
- Action 8: While the user would be aware of copy operations by the use of other applications, they may well not understand that you can copy key words as a short cut. [Student does not seem to understand that copy is not just a shortcut, but a way to tell the knowledge base that it concerns the same ship. So, this may not be clear to novice users]
- Action 17: I do not think the user will expect that they have to save and then submit their work. Perhaps these functions could be combined. Also, the icon for ‘Submit’ is not immediately obvious (the hint does clarify the situation). Finally, no feedback is given that a submission was successful.

4.1.3 Group5

This group did not perform a structured cognitive walkthrough (in the way it is supposed to be done).

Problems found:

- To logon you have to click on ‘Submit’ after typing in the password to enter the system. It would be easier if the user could hit the ‘Enter’ key instead of having to click the Submit button. If keeping the ‘Submit’ button, it would be more explicit to call it ‘Log in’.
- In the left side of the window, there is an option ‘No answer’ which is always in grey.
- Time of loading is too long.
- When posing a French query there are huge grammar mistakes: the verb’s singularity or plurality does not match the subject.
- When we chose ‘Submit as’, a window appears asking us to save (no mention of submitting anything). This is confusing.
- When a new query is saved the answer to it overwrites some other already saved files in the same folder.
- The ‘Restart’ icon in the Query window is not very obvious to the user.
- When a request to close the Query window is sent only that window is closed, not other windows even though they were created by the Query window’s menu.
- The text of the articles is not wrapped, we have to horizontally scroll to read it all.

- Sometimes ‘Submit’ in the query menu becomes ‘Submit as’, so we end up with two ‘Submit as’ in the menu.
- When we save a query, it does not ask where we want to save it.

4.2 Heuristic Evaluation

4.2.1 Visibility of system status

The system should always keep users informed about what is going on, through appropriate feedback within reasonable time. Clearly mark links.

- Menu items do not highlight on mouse over as they do in MS Word, Internet Explorer, etc.) (Group2)
- The fact that the French flag is highlighted on the initial window and two other flags are dimmed gives the impression that we are using French. (Group2)
- The button under Help makes me think I should click on it but this is not the case. It appears to be for an entirely different purpose –that of informing the user when the answer to their query has arrived. Perhaps it should be moved to a different location in the window (Group2).
- On load of menu in left-hand pane no feedback is given. While your applet loads you should have a message saying ‘applet loading’. (Group8)
- No information is given to the user to say whether submission of a query has been successful and the user is just dumped back at the main entry point. (Group8)

4.2.2 Match between system and the real world

The system should speak the users’ language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.

- On the window where the results appear, we can see links in the answer pointing to articles where the information is. The naming of the links makes no sense to us (just some letters and numbers). (Group5)
- For the language icon, it surely would make sense to have the float over hint in that language (instead of always in English) (Group8)

4.2.3 User control and freedom

Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

- ‘Restart’ seems to take me home, but it would be clearer if there were a button called ‘Home’ which would be consistent with other Internet browser software. (Group2)

- Although there are 'Restart' and 'Exit' buttons, there is not one for 'Undo', although it is available in the Query menu. As I think it is likely to be a fairly frequent action a dedicated button alongside 'Exit' and 'Restart' would be useful. (Group2)
- The font used is ideal being sans serif. However, users may not easily be able to find the option enabling them to change the size of the font. The addition of a 'Format' menu would help. (Group2)
- You need to be careful with using colours in case the user is colour blind. (Group8)
- On leaving any of the help pages, there is no way back to the homepage, so you have to go back to it manually and re-enter your details. (Group8)

4.2.4 Consistency and standards

Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

- Menu structures quite confusing. Without reading the documentation I would not know that I should select 'Query', 'Edit', 'Current', in order to base my query on a template. I think it would be better to use 'File', 'Edit', 'View' as titles in the menu-bar as in MS Word and Internet Explorer. 'Query' could come under the 'Edit' menu with a sub option of either 'New' or 'Edit Current'. The 'View' menu could list options as it is in popular Office applications. 'Help' looks as if it is in the right place. (Group2)
- Links in Help documentation are not underlined (hyphen with astrix above is used) and are therefore not consistent with standard blue underlined links. (Group2)
- 'Restart' and 'Exit' are normally associated with restarting the computer and exiting an application, whereas they have been used to mean something slightly different here. It would be preferable to express these as 'Restart query' and 'Exit query'. (Group2)
- We would like to use the right mouse button to delete, copy, paste, etc., like in Office applications. (Group5)
- The colour schemes used for the different pages do not have any relation to each other. Nobody can tell that the pages are related. There is no consistency in colours. (Group5)
- We cannot override a query. We have to save it under another name and delete the original one. (Group5)
- No menu alternative has been provided to the icon for the language. (Group8)
- There are no navigation links in the pages http://localhost:8000/CLIME/help/bugs.htm, http://localhost:8000:/CLIME/help/index.htm , http://localhost:8000/CLIME/help/UserManual/index.htm ,and http://localhost:8000/CLIME/help/Tutorial/index.htm . (Group8)

4.2.5 Error prevention

Even better than good error messages is a careful design which prevents a problem from occurring in the first place.

- There do not appear to be any alerts. When submitting a query it would be useful to have a message asking the user if they want to proceed or amend their query. (Group2)
- When we delete a file, we are not asked for confirmation (Group5)
- When we close the query window by using the x icon (on the top right of the title bar), the system does not ask us if we want to save the query (Group5)
- When attempting to save a query under a name with spaces in it, the system accepts it, and then stops functioning (freezes) (Group5)

4.2.6 Recognition rather than recall

Make objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

- The use of colour coding means that the user has to remember what the colours stand for. Perhaps you could use a drop down box for changeable variables, with no default selection for items that must be changed. (Group8)
- Green and red links in the query windows are not obviously selectable. A raised box and a message on mouse over would help. (Group2)
- It would be clearer if there were a 'Submit' or 'Proceed with query' button on the bottom of the query window. (Group2)
- The icon in the menu options next to Templates is not descriptive. Replace the icon with something the users will recognise straight away, 'Folder' icon maybe. (Group8)

4.2.7 Flexibility and efficiency of use

Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

- There are no accelerators in the main html page and query window. Put in non-conflicting (with browser) accelerators in the menus. (Group8)

4.2.8 Aesthetic and minimalist design

Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

- In the http://localhost:8000/servlet/Auth window: The left part should be narrower than the right one, it wastes a lot of space. The foreground/background colours in this part make the text very difficult to read. (Group5)

- The colour scheme between the authentication page, main CLIME page, the manuals/help page and error pages is awful. There is a need to implement a side wide approach. (Group8)

4.2.9 Help users recognize, diagnose, and recover from errors

Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

- There do not appear to be any error messages within the system. When the system appears to be doing nothing the user may assume it has crashed (perhaps it did). (Group2)
- When attempting to access a stored illegally named (with spaces in it) query in order to delete it the system freezes and the computer has to be restarted. (Group5)
- There is no link back to the authentication page on entering an incorrect username or password. (Group8)
- On selecting Query->Edit->Current without any selected nothing happens, there is no error message to the user. Put in a pop-up dialogue box saying “You must select a query to edit”. (Group8)
- On entering a long username or password, no feedback is given when you are unable to enter any more characters. Tell the user the maximum length of the field. (Group8)
- On entering no data input type in the following URL (http://localhost:8000/CLIME/CLIMEDevelopmentTool.htm) and pressing ‘Submit’ you are just given an error message saying ‘Error –no data input type was specified’ with no links back. (Group8)
- On bookmarking CLIME.html (http://localhost:8000/servlet/Auth) you get a HTTP 500 error. Override default browser error codes for your own and put links in to appropriate page. (Group8)
- On following one of the links to this URL: http://localhost:8000/CLIME/help/UserManual.htm you get a HTTP 404 error. Put sensible error message and back links in. (Group8)

4.2.10 Help and documentation

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user’s task, list concrete steps to be carried out, and not be too large.

- http://localhost:8000/CLIME/help/Questionnaire.doc The text is not readable, it is in machine language. (Group5)
- Misplaced information. On selecting Help->About, I would expect to typically find version number, who created the application and maybe date. I would not expect to find the following: Home page, Summary, Papers, Consortium, Address Book. While I think these things are needed, they do not belong in the about section. If you told

your typical user to find papers about CLIME most would definitely not think about the About box. (Group8)

4.2.11 Accessibility

Group 8 also performed a heuristic evaluation using W3C's accessibility heuristics (see http://www.w3.org/TR/WCAG10/).

Problems found:

- CLIME does not provide any keyboard alternatives to mouse actions. The guideline requires that all functionality is accessible via the keyboard.
- CLIME uses colour as the only means to convey information about anchors.
- The colours cannot be changed by the user.