An Integrated, Continuous Approach to Software Training Authoring, Delivery and Monitoring

April Vanessa Nixon

This thesis is submitted in fulfilment of the requirements for the degree of Master of Science in Computer Science, completed at The University of Auckland.

February 2002

Acknowledgements

I would firstly like to thank Professor John Grundy, for his supervision, ideas and guidance throughout this work, as well as efforts in securing an industry partnership with Orion Systems Ltd, and scholarship funding.

I would also like to thank Orion Systems Ltd, for allowing the use of their software products, and the staff in Orion's software support department, particularly David Brewerton and Brendon McAlevey for their participation in interviews, brainstorming and user testing. Also, thanks to the countless others who participated in the user surveys.

Thanks also to Technology New Zealand for the grant of a Technology in Industry Fellowship (TIF) and the help of the Research Office in liasing with TechNZ, as well as The University of Auckland Department of Computer Science, for the award of a tuition fees bursary, both of which have helped me to put all of my effort over the last year into this work.

Finally, special thanks to my family and friends who have supported and put up with me through the duration of this work. I owe you everything.

Abstract

Computer-assisted instruction can be defined as the use of computers as a medium for delivering instruction. Much research has been carried out in this area, which has seen the development of both research and commercial systems for authoring computer-assisted instruction, both generic and for specific domains.

In this thesis, we present our ideas for the authoring and delivery of computer-assisted instruction within the domain of software training. Our focus within this area is to test the concept of an integrated authoring and delivery environment focused specifically at software development organisations, which supports continuous service provision and continuous improvement through iteration, real time delivery and reciprocal feedback between learners and authors.

We first identify and discuss the issues related to implementing such a system, then present a specification and design of the system and describe a prototype of the system we have developed. Finally, we present an evaluation of the two main tools within the system and comment on potential future directions for this work.

Chapter 1 Introduction

1.1 Introduction

Educational software built to provide computer-assisted instruction has been an active area of research since the 1950's. The use of this educational software to specifically provide software training began with the mass popular acceptance of personal computers and with them desktop software in the late 1970's. Due partly to the amount of time that has passed since and the highly variant collection of software that exists today, all of which requires some user training, this is a very large area of research, with much work already done on tools for authoring software training, forms that training is manifested in for presentation to users and more recently, intelligence capabilities that add value and personalisation to software training. Our focus within this area is to test the concept of an integrated authoring and delivery environment focused specifically at software development organisations, which supports continuous service provision and continuous improvement through iteration, real time delivery and reciprocal feedback between learners and authors.

The rest of this chapter presents our motivation for carrying out this work, the specific goals of the work and the approach we have taken, as well as an outline of the contents of the rest of this document.

1.2 Motivation

With increased use of computers and software in more occupations and the increasingly business-critical nature of this software, it is important that users receive sufficient training. Orion Systems recently established a support team for their popular messaging product, Symphonia Messaging Toolkit. This team has assembled manual-style documentation and product usage examples, but the need for more advanced level training and assistance is recognised. This project will explore the development of task-oriented software training through a tutorial building, delivery and monitoring product. We will then attempt to abstract an improved service model, to move from the current single distribution of a paper-based training manual to a model of continuous service of dynamic training material.

1.3 Goals

By modelling the provision of software training as a continuous service of dynamic material, rather than as a one-off delivery of, for example, a static training manual, and by constructing a tool for authoring and delivering this training material, I hope to provide training authors with an environment specifically targeted at producing and maintaining software training material. The goals of the environment are to reduce the time required to author, update and distribute training material, make the process more integrated and more user-friendly and make the appearance of the material more consistent. The presence of end user usage and testing reporting, and an open communication channel between authors and end users, aims to discover if this type of information is useful input to the iterative refining of training material, and therefore warrants the continuous service of training material.

For the end users of software training, I hope the continuous service training provision model will provide a higher quality service by removing any feeling of isolation from training providers. I also hope to provide end users with a communication channel, interactivity, more frequent updates and additions of training material and therefore the ability to select from more relevant, up to date material.

The specific goals of this thesis then are:

- To assess the need for user training for Orion's Symphonia system
- To model the provision of software training as a continuous service
- To develop a prototype tutorial building and delivery system that provides task-oriented, advanced level assistance

Although the prototype tutorial system will be used to develop software training specifically for the Symphonia messaging product, the generalisation of the key principles of the product and training method and results from its evaluation are very important for this research.

1.4 Approach

This research beings with a comprehensive literature review, covering similar research into educational software and approaches to providing computer-based training and support. From this review, we discover the main issues related to our focused area of research. We then develop high-level user requirements for an ideal solution for a system to enable the authoring and provision of software training as discussed earlier in this chapter. From these requirements, we specify a conceptual object model and design for the solution and implement this as a system infrastructure, and two tool prototypes, one for authoring and the other for viewing. We then investigate the concept of the tools through a case study within Orion Systems Ltd and thoroughly evaluate the tools.

1.5 Thesis Outline

From this point on the contents of this document are as follows:

- Chapter 2: Background and Related Research – describes previous and current research in the subject area of this thesis.

- Chapter 3: Problem Statement – presents an in-depth description of the problem this thesis aims to investigate, including a discussion of the main issues to be investigated.

- Chapter 4: Specification – contains the detailed specification of the software training authoring and delivery system, including user requirements.

- Chapter 5: Design – presents the conceptual design of the system, including conceptual object models, system architecture and user interface design, according to the specification presented in the previous chapter.

- Chapter 6: Prototype Implementation – discusses the implementation of the tool prototypes, including any deviation from the conceptual design presented in the previous chapter, interesting design and implementation decisions made and the advantages and disadvantages of these.

- Chapter 7: Case Study: Using the Tools in a Software Development Environment – presents an in-depth description of the use of the prototype tools, first to construct an entire training course for Orion's Symphonia Messaging Toolkit and second, to allow users to view the completed course.

- Chapter 8: Evaluation - discusses the methods used to evaluate the prototype tools and the results of the evaluation.

- Chapter 9: Conclusion – summarises the contributions made by this thesis to the subject area and presents possibilities for future work.

1.6 Summary

This chapter outlines our research area, presents our motivation for doing this work, our general and specific goals and the approach we have taken to carry out this research. It has also outlined the contents of the remainder of this document. The next chapter contains a description of the research closely related to our work, and from there, the following chapters present our work in detail.

Chapter 2 Background and Related Research

2.1 Introduction

This chapter introduces the related research for this thesis. The first section of this chapter presents relevant background information on the general subject area of this thesis. Then the research within this area that is most similar to the focus of this thesis is presented.

2.2 Background

2.2.1 Computer Assisted Instruction

Computer assisted instruction, also known as CAI, can be defined as the use of computers as a medium for delivering instruction, and is analogous to computer-based training (CBT). A closely related field to CAI is computer-assisted learning (CAL), however this field of research is focused on the use of computers as a tool in a traditional learning environment such as a classroom, so has less relevance to this thesis than CAI.

The origins of CAI can be traced back to the 1950's, when several commercial companies, some in conjunction with researchers at universities, saw a market opportunity and attempted to adapt computers for instructional purposes [Buck 1995]. Although several products made it to market, including the Mentor system developed by Thompson, Ramo and Wooldridge Inc, and the Plato system developed by the University of Illinois, very little commercial success was experienced due to costs too high for most schools and educational institutions to afford. In spite of this, International Business Machines, Corp (IBM) did proceed to develop the IBM 1500 Computer-Assisted Instructional System [IBM 1966], an integrated multimedia computer-based instructional system. The issues the 1500 attempted to address, even at this early stage, included producing a generic educational system, capable of instructing on any topic, student registration, networking to allow multiple simultaneous students, and performance recording and analysis. These are still issues under investigation in this field of research today.

Since the field of computer-assisted instruction is very broad, it is broken down for description into several key issues that are most often the focus of research in this area.

2.2.1.1 Instructional Strategy

An instructional strategy is a technique used to teach users of a computer-assisted instructional system. This area of research is focused on identifying the possibilities available when selecting the instructional strategy or strategies for implementation within educational software, and discovering those strategies best suited to particular learning environments, user learning styles and material to be learned. A full range of instructional strategies identified by [Spalter et al 2000] includes a laboratory environment, visualisation, simulation, lecture and demonstration, case studies, role playing, mastery learning, creative project, student teaching, playground, drill, behaviour modification and incidental learning.

2.2.1.2 Content Design

Courseware content design is the study of building and structuring content that accurately represents and matches the concepts and tasks contained within a particular curriculum. The choice of content design model will depend greatly on the instructional strategy(s) chosen, as the content itself and navigation of this content are dictated by the instructional strategy. For example, a combination of the exploration and simulation instructional strategies will require the content to include rich visual images to provide realistic simulation and a highly non-linear connection between related content to allow for exploration.

2.2.1.3 Interface Design

As with content design, the choice of interface design is also closely linked to instructional strategy. With a move toward simulation and the use of virtual reality to enable exploration within a rich and complex learning environment, significant effort is required in designing and developing user interfaces. In such cases, issues including the appropriate use of multimedia to avoid learners becoming overwhelmed and accurately representing the real world need to be considered.

[van Dam 1999] believes the future of interfaces for computer-assisted education lies in multi-modal interfaces, such as those currently under investigation at Brown University that combine gesture and speech for carrying out two and three-dimensional tasks. This means a move away from desktop computing and the desktop metaphor. Also, interfaces that facilitate collaboration between learners hold much promise and should be focused on.

2.2.1.4 Intelligence

Intelligent Tutoring Systems (ITSs) are defined by [Murray 1998] to be “computer-based instructional systems that have separate data bases, or knowledge bases, for instructional content (specifying what to teach), and for teaching strategies (specifying how to teach), and attempt to use inferences about a student's mastery of topics to dynamically adapt instruction.” The addition of intelligence to CAI is researched in an attempt to move closer to the ideal in automated instruction, away from static screens and contrived situations to simulating realistic situations and adapting to the student’s learning state through the use of knowledge bases.

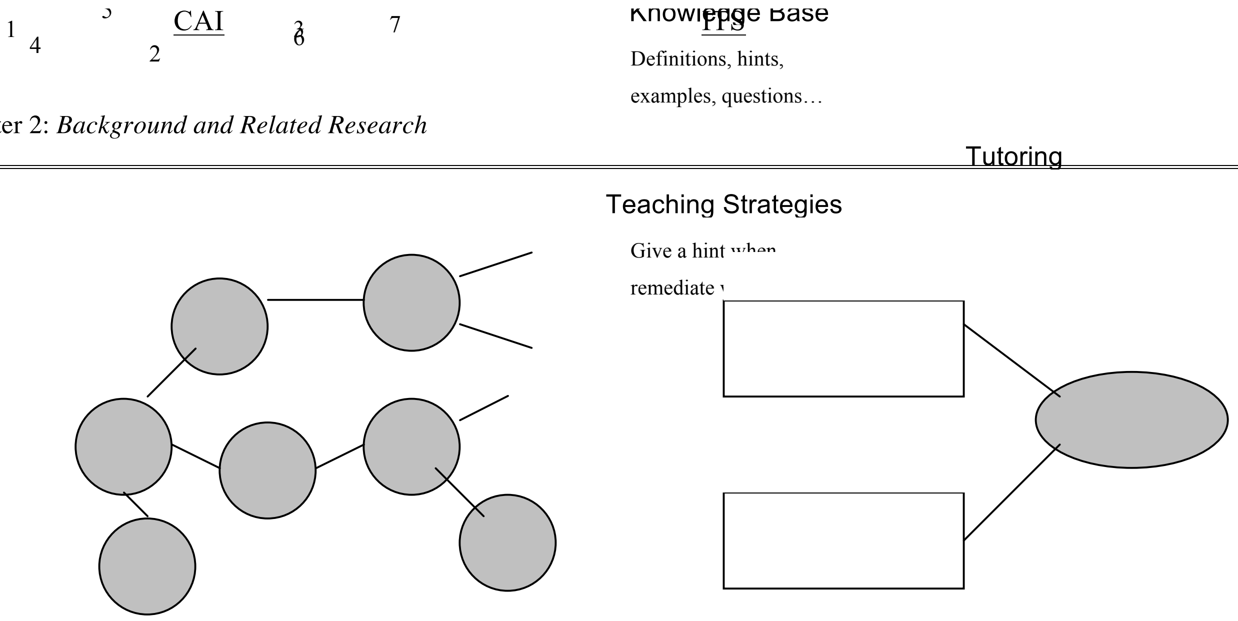


Figure 2.1: CAI storyboard versus ITS knowledge base

The main difference between traditional CAI systems and Intelligent Tutoring Systems is the representation of content, as shown in Figure 2.1. In CAI, content is generally designed using a ‘storyboard’ paradigm, where screens are designed and all possible navigations between these screens are mapped. Although navigation may be non-linear in structure, it is still pre-defined. In ITS’s, content is kept separate from the specification of how and when content is presented to the student, allowing multiple use and reuse of content.

2.3 Related Research

2.3.1 Hagler and Marcy

Marion Hagler and William M. Marcy, working out of Texas Tech University, are investigating the construction of computer simulated learning environments. To this end, they have created SIMPLE, a software environment “that makes it easy to combine existing software items (programs, simulators, word processing files, graphics, video, audio, …) to structure and deploy complex interactive learning environments for students” [Hagler et al 1995].

The main issues investigated in this work include:

A purely visual approach to constructing content and user interfaces
The authors’ goal here is to enable instructors to create complex learning environments using point-and-click methods alone. No programming or writing of control scripts should be required.

Combining existing software components to create and display complex multimedia
The strategy used by the authors here is to use OLE (Object Linking and Embedding) to combine existing software components to create learning environments containing disparate media. In this way, the SIMPLE environment acts

like "glue", linking the desired software components together in a seamless manner. SIMPLE's user interface is shown in Figure 1.1, demonstrating the concept of "gluing" screens built using external software components together.

Storing both content and visual user interface elements in a relational database

The authors have designed SIMPLE as a data driven application, where both the content and visual user interface elements of learning environments are stored in a relational database. This enables the construction of virtual graphical learner interfaces on the fly.

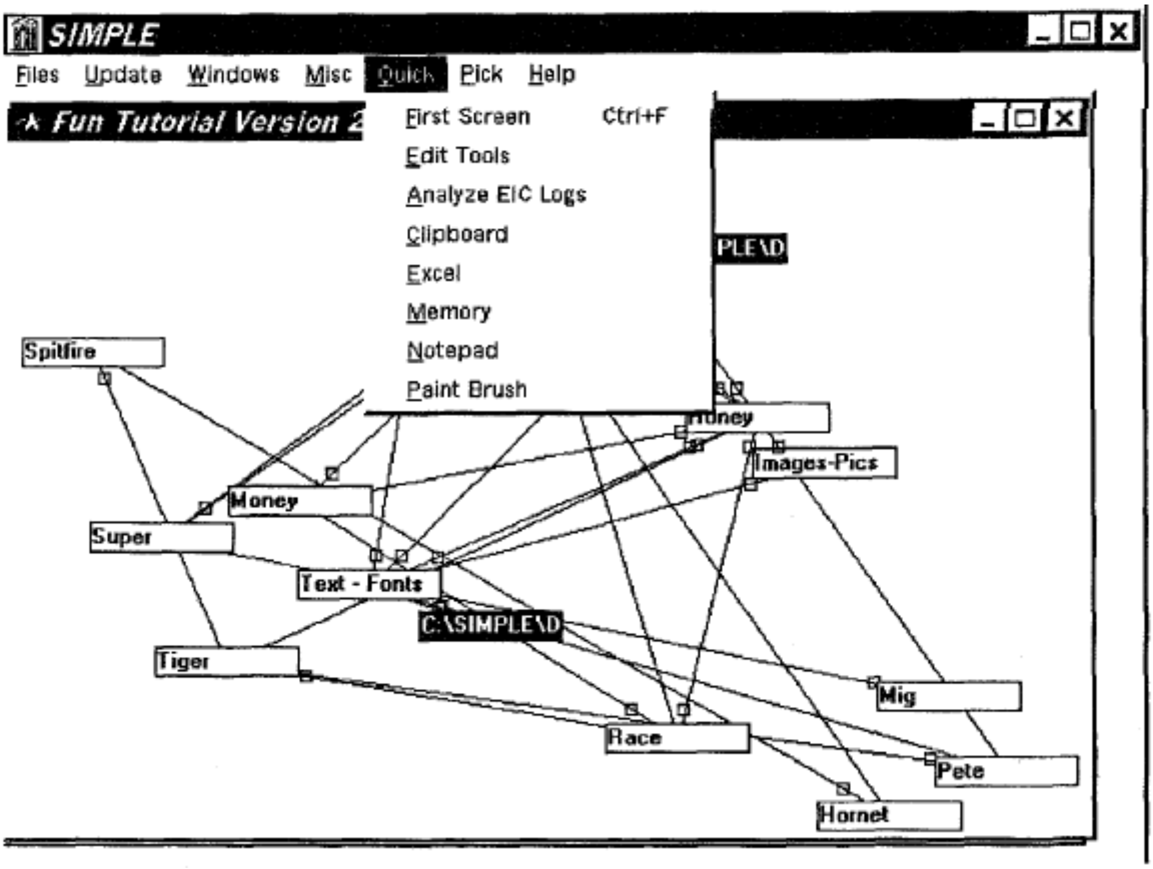


Figure 2.2: SIMPLE user interface showing the "glue" between screens created using existing software tools

The provision of feedback

Once learning environments have been distributed to students, the authors designed the SIMPLE environment to automatically log a learner's activity in an event-in-context (EIC) log for later examination by instructors to evaluate learner progress and problems. SIMPLE contains tools that allow an instructor to search the log and even play back events from a particular learning session on screen to allow some insight into the thought processes of the learner. The authors feel the graphical context of the feedback provided is very important to its successful use.

It is reported in [Hagler et al 1996] that work on tools that summarise and process the information gathered through monitoring continues.

SIMPLE learning environments, once authored, are distributed to students in a conventional manner, through the download of all of the files and database required to run the environment to the location of the student, or through the student accessing a computer where the environment is already stored. In [Hagler et al 1996], the authors conclude that the delivering of a learning environment once authored is a “formidable task” and do not claim to have resolved this issue through their research [Hagler et al 1995], [Hagler et al 1996].

2.3.2 Huston et al

Several members of the Iowa State University Engineering College’s Courseware Development Studio have been producing courseware focused on instructing engineering as part of the NSF Synthesis Coalition since 1990 [Huston et al 1994]. The authors include, Jeffrey C. Huston, Jason C. Gillette, Christoph Hiemecke, Richard M. Johnson, Mark R. Eversden and Ryan J. Pletka.

The main issues investigated in this work include:

Combining multiple curricula

Much of the work is focused on determining the benefits of creating multidisciplinary course content. The goal is to provide a greater appreciation of the interconnection of related subjects through working with courseware that has content designed to take advantage of the interconnections. The authors’ strategy for achieving this is to model the design of course content and course applicability to curricular units, by showing interconnections between instructional materials. An example of this is shown in Figure 1.2, where the interconnections between two courses on ergonomics and tire mechanics are shown, along with the applicability of these two courses to engineering curricular units.

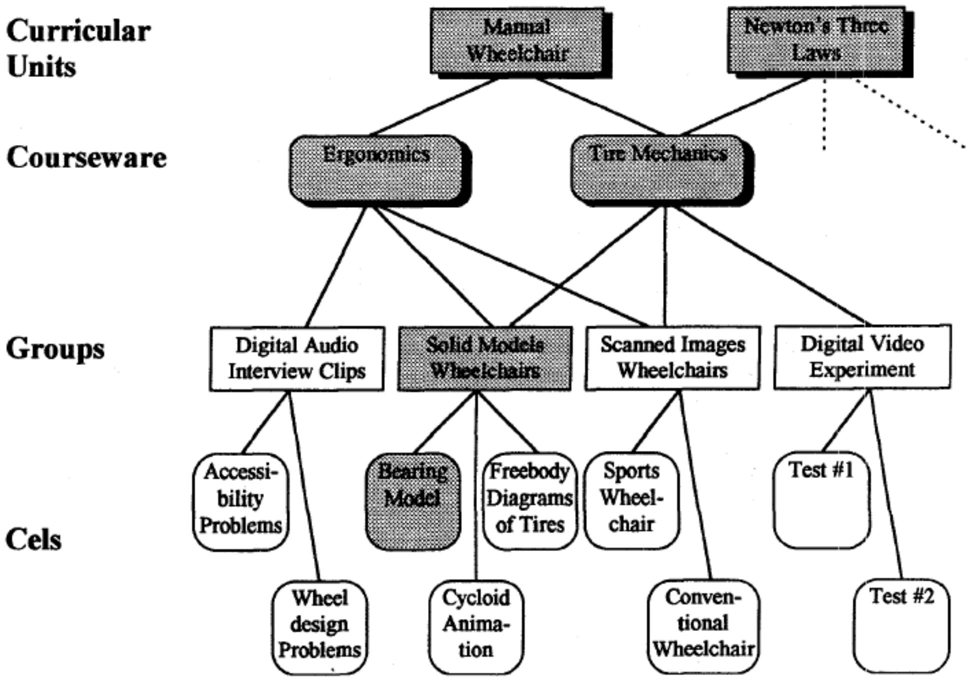


Figure 2.3: Example content model showing interconnections between instructional materials

Instructional strategy

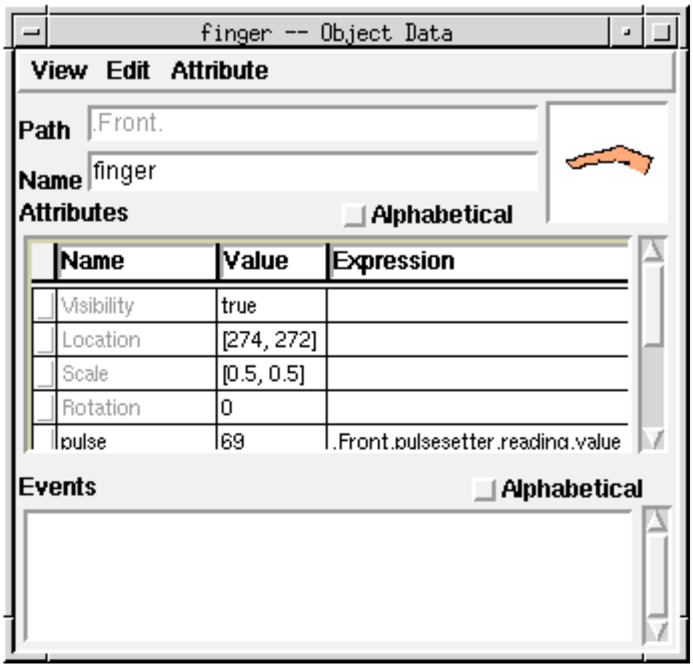
The authors have identified two main instructional strategies for use within the educational software produced as part of their research in this area. These are instructor-centred material and learner-centred material. The authors particularly identify with the advantages of learner-centred material, where content is presented in a highly non-linear and interactive way, as they feel this type of material allows learner actions to be recorded and used both to monitor learners' progress and to identify weak sections within the instructional material itself.

Within this work, the authors have not investigated any issues pertaining to authorship or delivery of courseware, but rather focussed on the issue of content design, using existing tools to create the prototype courseware used to evaluate their conceptual content design model.

2.3.3 Munro and Towne

Allen Munro and Douglas M. Towne, working out of the Behavioural Technology Laboratories at the University of Southern California, are the principle developers of RIDES, the Rapid ITS Development Environment. RIDES is described by [Munro et al] as "an integrated software environment for developing and delivering computer based tutorial instruction and practice in the context of graphical simulations".

Instructional content and simulations built using RIDES are based on a graphical model of the problem domain. Authors are able to build graphical models by initially creating a “scene” and then opening library objects and pasting them into scenes, or drawing objects directly onto the scenes, using a palette of drawing tools. The behaviour of objects is then specified by authoring rules that control the value of object attributes, as shown in Figure 2.4. So while a graphical model is eventually produced, the author must have some knowledge of how to author appropriate rules to govern the behaviour of the model.



Path .Front.

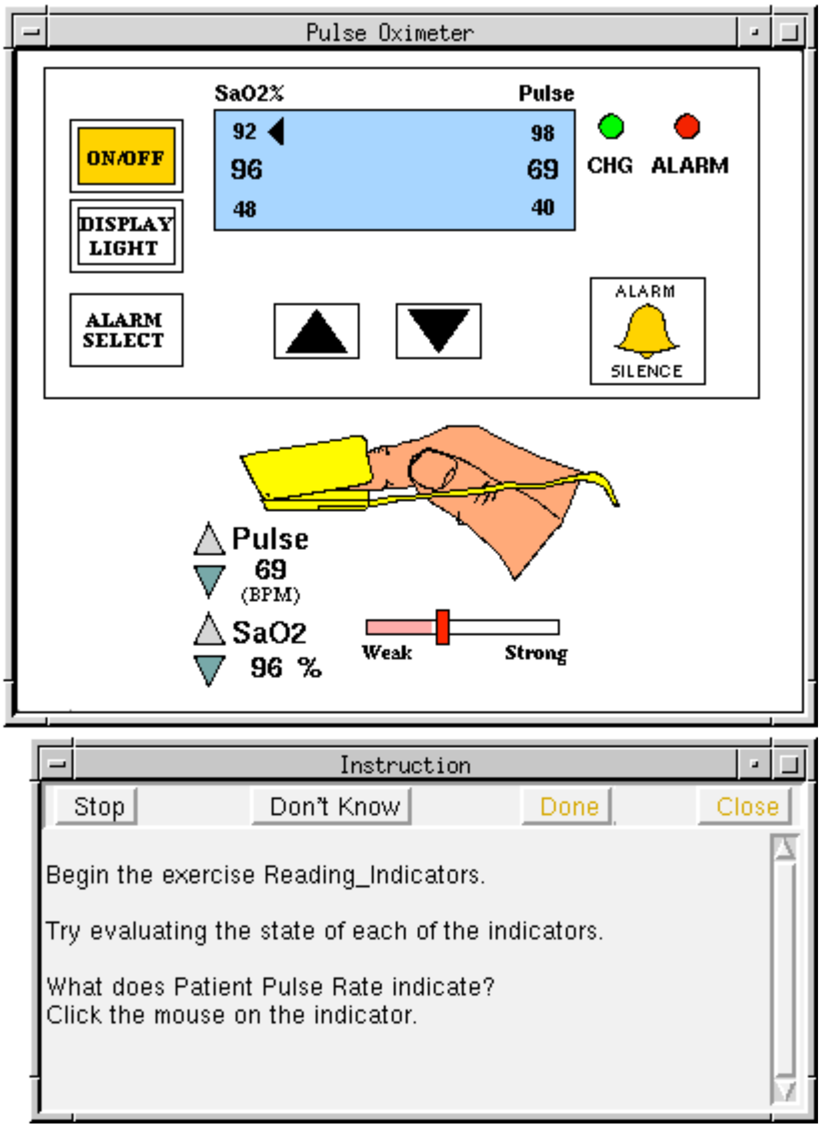
Name finger

Name	Value	Expression
Visibility	true	
Location	[274, 272]	
Scale	[0.5, 0.5]	
Rotation	0	
pulse	69	.Front.pulsesetter.reading.value

Figure 2.4: RIDES data view of a graphical object

RIDES has lesson-building tools that can then use the information already authored in the graphical model. When building a lesson using RIDES, an author can 'record' a procedure that students must learn in a way similar to recording a macro, simply by carrying out the procedure. Students may then carry out the sequence of actions that constitute the task and receive meaningful feedback on the correctness of their actions. Such a lesson constructed using RIDES is shown in Figure 2.5. In this example, the student is a nurse and how to set up and use a piece of medical equipment is the domain represented by the lesson’s graphical object model.

The main issues of CAI investigated in this work are content design through the specification of a graphical object model for simulation, and interface design of highly interactive, graphical interfaces built from the same graphical object model.



Begin the exercise Reading_Indicators.

Try evaluating the state of each of the indicators.

What does Patient Pulse Rate indicate?
Click the mouse on the indicator.

Figure 2.5: RIDES student view of instruction

2.3.4 Other Related Research

van Dam et al

Three colleagues at Brown University, Andries van Dam, Rosemary Simpson and Anne Spalter, have been investigating computer-assisted education for over three decades, with Andries van Dam beginning research in the area of educational software in 1966-67 [van Dam 1999]. While they have tried to understand all of the difficulties involved in developing educational software, the main issues under investigation by their team include pedagogical or instructional strategies, learning environment user interfaces and learner collaboration, with the goal of really improving education through the medium of the computer. Some of the major projects worked on at Brown in the field of educational software include:

- HES (Hypertext Editing System) 1967-68
- FRESS (File Retrieval and Editing System) 1968-78
- EDS (Electronic Document System) 1979-84
- BUMPS (Brown University Multiple Projection System) 1980
- MIDAS (Microprocessor Display and Simulation) 1981
- BALSA (Brown Algorithm Simulator and Animator) 1981-86
- IRIS' Intermedia 1983-89
- DynaText 1990 onwards

Although some of these research projects were carried out many years ago, the Computer Science Department at Brown University continues their research into educational software today. In particular, they are trying to generalise their methodologies and findings from research into educational software by constructing a handbook that documents the generalities of their research and constructing design patterns for applying pedagogical strategies to learning situations within educational software [van Dam 1999], [Spalter et al 2000].

Murray

Tom Murray, working as part of the Centre for Knowledge Communication at the University of Massachusetts has investigated many of the issues that arise from the addition of intelligence to CAI to create ITS. He has focused on the authoring of ITS's, identifying the key components of an ITS and developing domain independent tools for authoring each of these components: the domain model being taught, the teaching strategies to be implemented, the model of student knowledge and the learning environment itself [Murray 1998].

The Eon suite of ITS authoring tools grew out of Murray's dissertation thesis on tools for the acquisition of ITS domain and teaching knowledge [Murray 1991]. Each of the tools and how they work together to design and construct the components of an ITS are shown in Figure 2.6. The main issues investigated as part of the construction of the Eon suite of tools are content design and modelling, instructional strategy and interface design, all in the context of creating intelligent computer-assisted instruction.

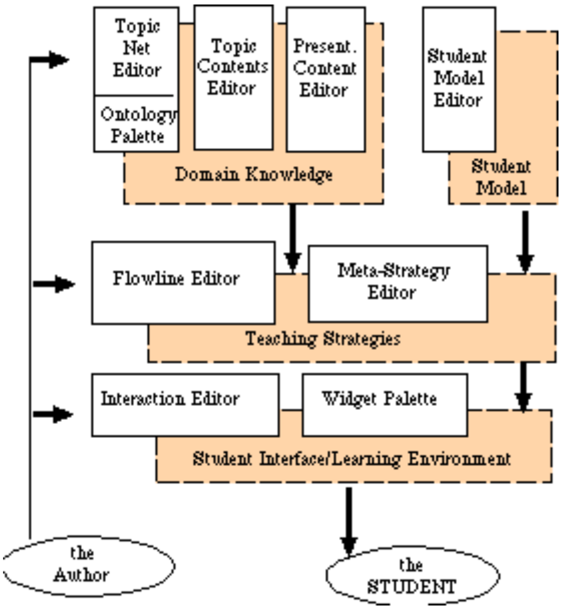


Figure 2.6: The EON authoring tools and their uses

According to [Murray 1998], "commercial authoring systems excel in giving the instructional designer tools to produce visually appealing and interactive screens, but behind the presentation screens is a shallow representation of content and pedagogy". It is for this reason, and the potential for creating much deeper and more meaningful representations of instructional content and strategy, and building the interfaces for learning environments from these representations that Murray has worked in this field of research.

2.4 Summary

The first section of this chapter presented background information on the research area of this thesis, necessary for understanding the overall placement of this work. This included a definition and description of computer-assisted instruction and learning and of educational software that is developed to deliver it. The second section presented recent work within this research area that is most closely related to the focus of this thesis.

Based on the very broad research area of computer-assisted instruction (CAI), several issues have been identified as the current main areas of research within CAI. These are instructional strategy, content design, interface design, intelligence and the provision of feedback to authors and learners. Due to the broad nature of CAI, many researchers have and are currently working in this area. Those whose work most similarly relates to the focus of this thesis are Hagler and Marcy, Huston et al and Munro and Towne.

Chapter 3 Problem Statement

3.1 Introduction

This chapter describes the problem domain this thesis aims to investigate. It presents the issues we feel are most important in integrating a software authoring and distribution environment with a service-oriented model of training provision. The scope of this thesis is then defined by those issues that will be investigated. Finally, the requirements for realising such an environment are outlined.

3.2 Issues

The issues outlined in this section are those identified through the review of related research that we feel are most important to the investigation of an integrated authoring and delivery environment focused specifically at software development organisations, which supports continuous service provision and continuous improvement through iteration, real time delivery and reciprocal feedback between learners and authors.

Feedback

The issue of feedback has been addressed by several of the authors in the previous chapter on related research. We feel that feedback to both authors and learners is very important and could be used to support an iterative process of training improvement, as well as to improve the perceived level of service received by learners. The types of learner feedback we intend to investigate are progress and achievement feedback, and the types of author feedback are usage information and explicit feedback on course quality. We also intend to allow for direct author-learner communication.

Customisation

An important issue in this area of research is the ability for teacher customisation, that is the ability for teachers present in the learning environment to make custom changes to the content of a training course. In the context of this thesis, the original author of the course and the teacher may be one and the same. So while in the first case users proficient in authoring educational software would author the training course and the teachers who would apply customisation would likely be proficient in teaching, in this case, a user may not be skilled in either authoring electronic training or teaching. The issue of customisation then becomes more an issue of providing a simple to use, graphical authoring tool, which allows rapid, easy changes to training material and makes those changes immediately available and which is restrictive and guiding with respect to training structure.

Distribution of Training Material

As stated by [Hagler et al 1996], the distribution of a learning environment, once authored, to all intended students is a “formidable task”. This difficulty is at least part of the reason for the infrequent distribution of updated software training material. We intend to investigate an environment where training courses are available as a continuous service to registered users, and the most up to date training material is pulled automatically by viewing clients from a central repository.

Interface and Content Design

While not as important to the concepts of this work as the issues already mentioned, appropriate interface and content design needs to be considered in order to create training courses suitable for evaluation of the central concepts of the prototypical system used to produce them. For example, the features that achieve usage monitoring and reporting cannot be investigated without the use of adequate training courses.

While other issues identified through the review of related research are very important to the design of educational software and tools to author educational software, such as instructional strategy and intelligence, they are outside of the scope of this thesis.

3.3 Requirements

To investigate the issues identified in the previous section, we have developed these high level requirements for a prototypical software system:

Store training course designs in a central repository

Training course design, once authored, should be stored in a single, central repository, available for editing and viewing. As well as the courses themselves, user membership and access rights to courses and logged usage information should be stored in this central repository.

Provide continuous service of these designs to multiple authoring and viewing clients simultaneously

Training courses should be available to multiple authors and multiple learners simultaneously, however concurrent editing shall not be supported. Access control to prevent concurrent editing of courses should be provided.

Enable the construction of training courses from multimedia elements

Provide core desktop publishing features to authors to enable the construction of training courses suitable for evaluation of the features under investigation in this work.

Allow the viewing of a personalised subset of the training courses available
Through a membership hierarchy, allow learners access to the subset of all courses relevant to them. Relevance in this context refers to the learner or the learner's organisation having a current license for the software a training course teaches.

Monitor training course usage and collect feedback from learners for reporting to authors
In order to provide meaningful feedback on the usage of training courses, learner progress should be logged, without disturbing the learner, and made available to authors at their discretion. Learners should also be enabled to explicitly give feedback to authors on the quality of training courses.

Allow direct communication between learners and authors
Direct communication in the form of messages should be allowed between learners and authors. This should enable learners to ask direct questions of authors, for authors to make direct replies and for authors to broadcast information to all learners.

3.4 Summary

This chapter has highlighted those issues identified in the previous literature review as being most relevant to the focus of this work. The scope of this thesis is defined as the investigation of these issues. It then presents the essential high-level requirements we believe a tool that investigates these issues would need to implement.

The remainder of this document presents the detailed requirements, specification and design of an integrated software training authoring, delivery and monitoring system. Following that we describe the implementation of a prototype of the system, examples of its use through a case study and a thorough evaluation.

Chapter 4 Specification

4.1 Introduction

This chapter provides a detailed specification of the software training authoring and delivery system. This includes an overview of the system, detailed descriptions of the main components of the system, namely the viewing program and the authoring program, descriptions and drawings of the main entities that will be involved and the relationships between them. Also included are user requirements, in the form of use case diagrams and descriptions, and a description of the system architectures. Finally, all of the information in this chapter is summarised.

4.2 System Overview

In the previous section, the system was referred to as a software training authoring and delivery system. "Software training" entails all support material that users of a piece of software may expect to receive in order to learn and effectively use that piece of software. This includes the user help that would most probably be included with the software, documentation in an electronic form and training material. The main focus of this research is the authoring and delivery of training material. Using this system, the other forms of support, namely help and electronic documentation, should be able to be loaded and delivered to users but not authored.

Software vendors should be able to use the authoring program component of the system to build electronic training courses – with one course corresponding to a single piece of software. These courses could then be served to users of that software, which they would access through the viewing program. However, instead of ending the relationship between author and user, with respect to this system, when a training course is made available to users, an ongoing, bi-directional relationship should be supported. User profiles and course usage information should be gathered as users use the courses and relayed to the author(s) of the courses. Authors could then interpret this information and act on it by making any necessary changes to existing courses, developing new courses etc. The aim of this is to create a relationship of iterative improvement between vendor software support staff and software users. As a result, it is hoped that authors will find it easier to identify possible faults or places for improvement in electronic training courses and users will receive a better end product and a better level of support service.

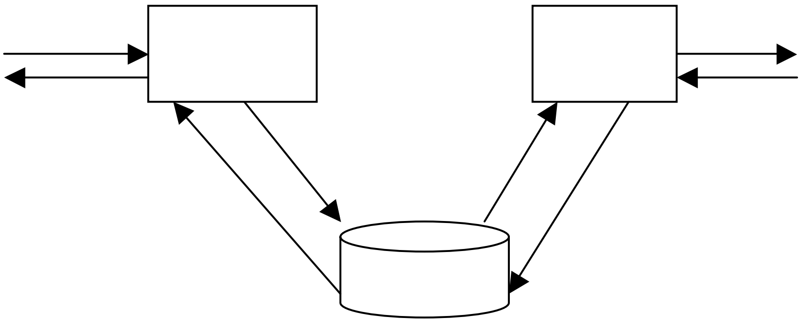

User profiles and usage feedback
designs
Author
program
User profiles and usage feedback
Support package designs
program
User profile, course reviews and usage information
Support service
User profile and course reviews
Course

Figure 4.1: System architecture

4.3 Authoring Program

4.3.1 Overview

This is the program that software vendors (most likely support staff) would use to specify and build a training manual course for a piece of software. This user of the proposed system will herein be called the "author". The author would begin by choosing either to work on an existing support package or to create a new software package. A software package relates to a single piece of software and must include a training course and optionally a help section and a product documentation section. If the author chooses to create a new support package, they would be prompted from information about the piece of software that this package is to support and would then be able to add new components to the package e.g. a training course.

Each component would need to be created in a different way. In this case of this research, help sections as found in most software applications and electronic documentation would not be created using the authoring program but would rather be loaded into the system. A component of the current course would still be created for them so that it could be seen as part of the support package structure and also so that it could be accessed through the client viewing program. A training course however, would be completely designed within the authoring program.

As well as the authoring program itself, there will need to be a set of tools that allow the author to perform specific functions such as setting up user accounts, changing passwords, creating animations and creating tests. The user account and password maintenance tools will need to be accessible in both the authoring and viewing programs, but the rest of the tools will only be needed in the authoring program.

The following types of users will be needed:

- Vendor side users - root level (absolute access), admin level (can create user accounts), author level (can create and change course designs).

- Client side users - admin level (can create user accounts), viewer level (can view courses and change their own information).

4.3.2 Content Views

When designing a training course several views should be available within the authoring program. At this stage, I can think of three views that would be suitable, namely structural view, detail view and storyboard view. Structural view would include an expandable menu that would have the support package as it root and all of its components (one or more training courses, a help section, an electronic documentation section) as its children. This view should always be present and most likely on the left hand side of the interface.

The detail view would exist on the right hand side of the interface and would reflect the details of whichever component was highlighted in the structural view. Components could be highlighted by clicking on them or would automatically be highlighted when they were first added to the structural tree. Included in the detail view should be a small image of what the age would look like to a course user.

The author should also be able to view each lesson as a whole. This would be done using storyboard view. When a lesson is highlighted in the structural view, the author should be able to select storyboard view and have the screen display all of the pages contained within the lesson in sequence. Each page would look exactly like it would to a user of the course program. To make it easier to get the full picture, the storyboard view should probably take up the entire interface, with the structural view minimised or hidden. The detail view would most likely be closed. I am not sure whether changes should be allowed to be made to pages when viewing in storyboard view, but this should be left open as a possibility.

4.3.3 Course Structure

The way that an author would work with this program is by creating and editing courses. First, a course is created, which doesn't initially consist of anything besides a name, a title and an association to the piece of software it will be teaching. The author could then build up the structure of the course by adding lessons. Each lesson should be related to a high-level task that a user might want to perform using the software, such as "opening the program" or "entering account data". Each lesson is then made up of subjects, which loosely relate to atomic tasks that the user would have to perform to achieve the overall task. Lastly, a subject is made up of one or more pages. Pages are the unit that the author actually works with, for instance text and images can be added to a page, not to a lesson or a subject. The sequence of pages will eventually be what the course user sees.

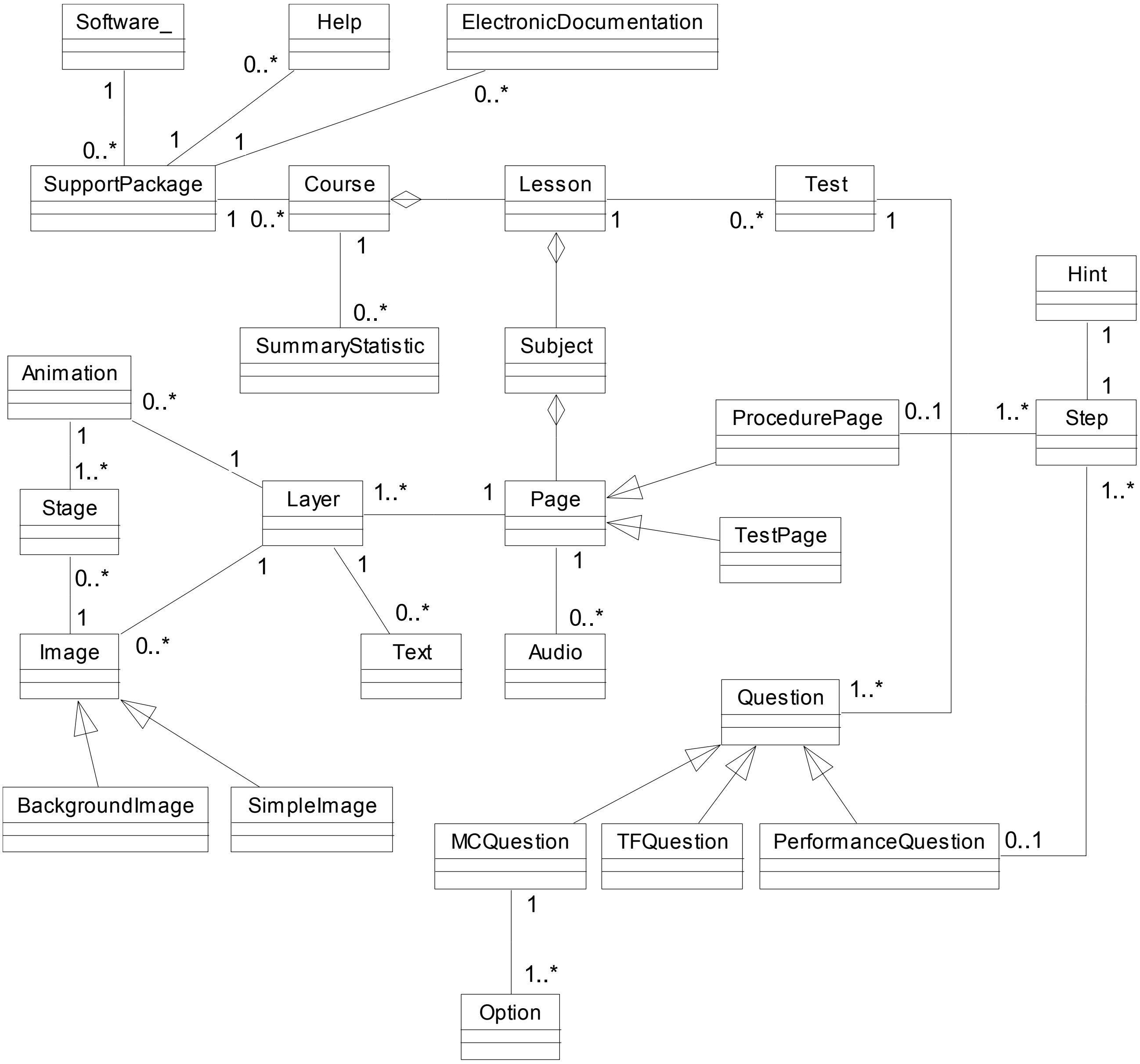


Figure 4.2 Basic course structure object model

4.3.4 Content

The items that an author should be able to add to a page includes:

- Background image – this could be any image but would be increased or decreased in size to fit exactly into the frame of a single page.
- Image – this could be added to any number of pages and could be resized and repositioned.
- Text
- Audio
- Animation

Initially, the plan is that the author will have to import everything but text and possibly simple graphics elements such as circles, rectangles, lines and arrows. However, I would also like to add a utility for creating animation, restricted to a sequence of pictures that represent actions that the user would perform with the software, for the purpose of demonstration. In this way, the user would be able to see the software in action and interact with it, without having to have a version of the software running simultaneously with the course.

The author should also be able to set basic properties for a page, such as colour, font style and font size.

4.3.5 Interface

The size of the authoring program interface should be as large as possible so that a maximum of information can be portrayed to the author. This is especially important when creating a presentation that will include graphics and text that must be easily readable. It will probably be best if the author is not able to change the size of the interface (only minimise and maximise) so that the interface elements remain in the same place and interface consistency is enforced.

When the author opens the authoring program, the initial view should be empty apart from a dialog that asks the user what they would like to do. The options should be "create a new support package" and "edit an existing support package". If the user selects create a new support package, they would be prompted for information including the name of the file, the title of the support package and some other basic properties. If however, they select to edit an existing package, this package would be opened, both structural view and detail view would be opened (with detail view containing the detail of the root node) and the usage statistics (overall and since the last change was made) would be displayed. The purpose of this is to help inform the author of what is good and what may need improving.

4.3.6 Functions

This section outlines the main user functions of the authoring program.

Administrative Functions

These functions are perhaps the most common i.e. not unique to this type of application, but are all necessary parts of the authoring program.

The first few administrative functions are related to opening and closing the program. The author must be able to open the program, login as a specific user, change their password if it has expired or simply if they choose to, and exit the program. The second set of author administrative functions are to do with setting up user accounts so that both authors and viewers have user accounts to login to the system. The author of a support package must be able to enter the details of the software the support package is for, enter in the license details for each client that purchases one, and finally create viewer and administrator user accounts for that license. The next set of functions revolves around the different views available for course content and how to navigate between them. This includes switching between the three main views, structural, detail and storyboard, as well as viewing the content catalogue and the summary usage information for a course. The last set of administrative functions includes the ability to save changes, undo and redo. It is likely that only last action undo and redo will be implemented. These last functions are seen as essential for a design program.

Course Structure Design Function

These functions relate to creating and altering the structure of a course.

The first set of functions allows the author to create structure components. This includes a parent support package, courses, lessons, subjects, pages and tests. Along with tests, the author can create component questions and provide the answers for these. The types of questions available are multi-choice, true/false and task performance questions. As well as create, the author must also be able to delete any one of these components. The next set of functions allows the user to set the properties of any of these course components. Properties of a page include the background colour, width, height, font, font style and font size. The next set of functions allows the author to alter the structure of a course. The author is able to change the order of any of the course components. This includes the order of lessons in a course, the order of subjects in a lesson and the order of pages in a subject.

Course Content Design Functions

The last set of functionality of the authoring program relates to designing the pages of the course. This mainly involves designing the content of each page and organising it on the page.

The first set of design functions relates to importing and designing media. The author is able to import images, text and audio into the content catalogue. There is a separate content catalogue for each course. The author is also able to design media, including simple shapes and text and animations. An animation will consist simply of a sequence of images and the actions that cause the sequence to progress. The next set of functions relates to adding content to a course page. Specifically, this includes adding, images, background images, text, animation and audio to a page. Along with designing, the author will also have the ability to edit media. For imported images, only the size and position will be able to be edited. When media is added to a page, each object will reside in its own layer. The author will therefore have the ability to switch between layers in detail view and to change the order of the layers. The last design function that authors will have is the ability to design the summary usage information that they would like to appear in detail view when the course node is selected. This is elaborated on in section 5.1.2, Usage Feedback.

4.4 Viewing Program

4.4.1 Overview

The viewer program is a much more simple application that the authoring program. The viewing program will reside on a viewer's desktop. Once the user opens the application and logs in, the program will display a simple list of all of the software from this vendor that they are licensed to use. By selecting a piece of software from the menu, the relevant support package is immediately opened. The viewer can then navigate through all of the components of the support package, including any training course(s), traditional help as would be found in the actual software application and electronic documentation. While using any of the courses available, the program will be monitoring the viewer's usage. This enables to program to reflect the viewer's progress back to them, through indications of which components have been completed, started, no attempted etc, as well as through test scores and even times to complete tasks. As well as feedback to the viewer, the viewing program allows feedback to the author(s). The viewer can provide explicit feedback, in the form of their user profile, so that the author(s) have some idea of the level of experience of the users of their courses, and in the form of messages sent to the authors, containing either comments or questions. Implicit feedback is extracted from the viewers in the form of usage information. The program constantly monitors what the viewer is doing and records each usage of each component of a support package. This information is relayed to the author(s) through the summary usage information displayed when they open a support package in their authoring program. The author is able to customise the type of usage information shown to them as well as the level of detail it represents.

4.4.2 Interface

The user interface for the viewing program will be of a set size so that viewer's cannot distort or rearrange the set-up of the interface. The main window allows the viewer to select from all of the support package options available. It will use large title and icon style menus for both name and image recognition. Once the viewer selects a lesson, the interface changes. The main menu window will be minimised and the lesson will be opened in a new window. This window contains both the pages of the lesson and the controls for navigating. The reason this is contained within a separate window is that the author is able to select the size of the pages in each course. This will likely not be an absolute freedom to choose, but rather then option to select between several predefined sizes. Once the viewer finishes the lesson or chooses to exit, the lesson window will disappear and the main menu window will again be maximised.

4.4.3 Functions

This section outlines the main user functions of the viewer program.

Administrative Functions

This section outlines all of those functions that are not specifically related to learning from a training course.

The first set of functionality relates to opening and closing the program. A viewer can open the program, login using a specific user account, change their password if they want and close the program. The next set of functions allows the viewer to enter and update the personal profile information. This is elaborated on in section 5.1.1, Profile Information. The last two administrative functions are viewing personal history, which includes profile information, personal details and usage and result history, and submitting a message to the author(s) of a course.

Support Viewing Functions

These functions relate specifically to using a training course. Firstly, the viewer is able to open a course or lesson. Conversely, they are also able to close either of these components. The viewer is also able to navigate through a lesson in both directions. The viewer is able to ask for a hint or a demonstration when the lesson page they are on is a task performance page. At the end of a lesson (or at any time they want to re-sit), the viewer is able to sit a test to assess their comprehension of the information they have just read. At the end of a lesson, the viewer is also able to rate the lesson, and at the completion of a course, do likewise. This information is relayed to the author(s) through their course summary information.

4.5 User Feedback

As users use the course-viewing program, user feedback will be collected and made available for authors to view at any time.

4.5.1 Types of User Feedback

The main types of user feedback that will be made available to course authors include profile information, usage information and specific feedback, including reviews of lessons, comments, mistakes found and questions.

Profile Information

The collected information for a user profile should include:

- Past experience
    - With a previous version of this software

- Use of other related products
- Experience with this piece of software and corresponding training course
  - Those parts of the training course that have already been completed.
  - Time taken to complete tasks
  - Test scores
  - Breakdown on different types of questions (e.g. performance based questions and comprehension questions).

Usage Feedback

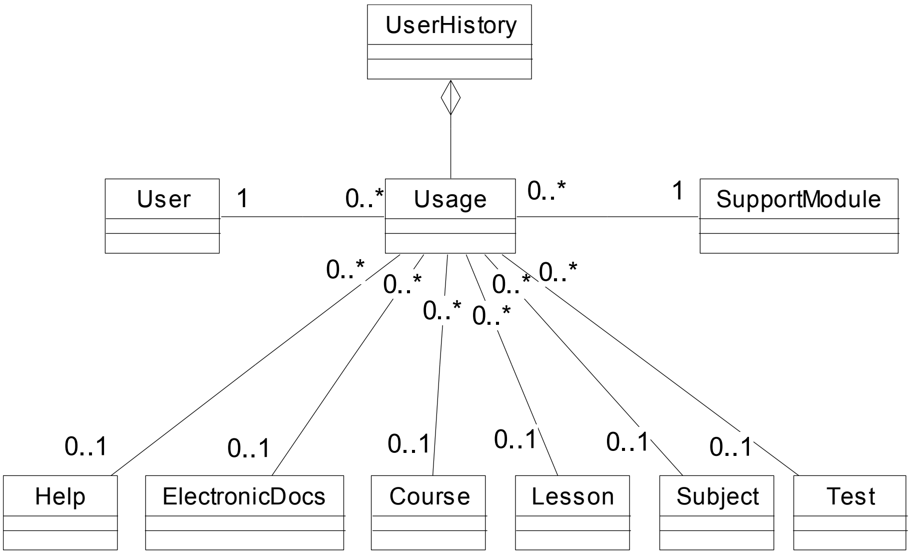


Figure 4.3: Usage data model

The authoring program should include a tool called the Summary Usage Information Designer. It should be a GUI tool that allows an author to construct the summary information that they want displayed to them in detail view when the course or lesson tree node is selected. The author should design the information by selecting options from combo boxes, and selecting radio buttons or checkboxes. This restriction of design through the GUI should make it easy enough for anyone to use and disable authors from trying to create summary info that the system cannot create.

Examples of summary information:

<u>Usage information</u>

- Number of times accessed

    - By course
        - Since last edited (default)
        - In last week
        - In last month
        - In last three months
        - In last six months
        - Since created
    - By lesson
    - By subject
- Number of different users who have accessed
    - By course
        - Since last edited (default)
        - …
    - By lesson
    - By subject
- Time spent
    - By course
        - Total
            - Since last edited (default)
            - …
        - Average
            - Since last edited (default)
            - …

Rating info

- Average rating
    - Of the course as a whole (default)
    - Of each component lesson

Testing/performance info

- Average test score
- Average score on
    - Multi-choice questions
    - True/false questions
    - Performance questions

Specific Feedback

Specific feedback is specific to a particular course or lesson. Once a viewer has finished viewing a lesson or course, they are able to rate both of these components, using a pre-defined rating scheme. In addition to this, at any time the viewer is able to send a message to the author(s). This message can contain either a comment or a question relating to the content or structure of the course.

4.5.2 Sources of User Feedback

User feedback will be sourced both explicitly and implicitly from users. When users first use the course viewing program, they will be prompted to enter information about themselves, which will constitute their user profile. This information should consist of mainly past personal experience information with existing software applications and how the users rate their own ability. The other source of explicit feedback is course reviews and specific questions. Once a user has completed a course (or perhaps a lesson) they will be able to rate the course and enter any comments they wish to make. They will also be able to submit specific questions to the course authors. I am not sure yet whether the answers to these questions will be returned through the course viewing program or sent via another route such as email, but both options are possible.

Monitoring how, when, what and for how long users use the course viewing program, will source implicit feedback. A usage history will be created for each separate user and will consist of a collection of usage records for the individual components of a course, e.g. 1 minute using Lesson 1, 30 seconds answering the test questions for Lesson 1 etc. By storing usage information at such a fine-grained level, it should be possible to reconstitute this into any summary information that the authors decide they would like. This usage information will be fed back to both the authors of the course and to the user themselves, as their interface will reflect which courses/lessons/subjects have been completed/partially completed etc. Users will also be able to look up their performance results for all of the tests they have completed.

4.5.3 Uses

There are two specific uses of all of this feedback information. The first relates to the relationship between the author(s) and the viewer. This relationship works in two directions. The first is allows the viewer to feel like they can instantly and easily access support from the software vendor by sending messages and receiving answers to their questions. The second direction is the sending of profile information from the viewer to the author. This helps to inform the author of the level of experience and skill of the users of the support packages they build.

The second use of feedback information is subtler. It enables a relationship between the viewer and the viewing program. As the viewer progresses through the course components, their level of skill and experience would hopefully change, and this should be reflected through a more customised interaction between the viewer and the program. One way this changed level of skill should be reflected is through a more appropriate level of difficulty being presented to the user. For example, the program could avoid presenting introductory lessons, introduce more advanced lessons that are not available to less skilled viewer or give the viewer more challenging test questions. Another customisation that could come through monitoring the viewer is through providing shortcuts for common actions the viewer performs or shortcuts to often-viewed course components.

4.6 Author Use Cases

4.6.1 Administrative Task Use Cases

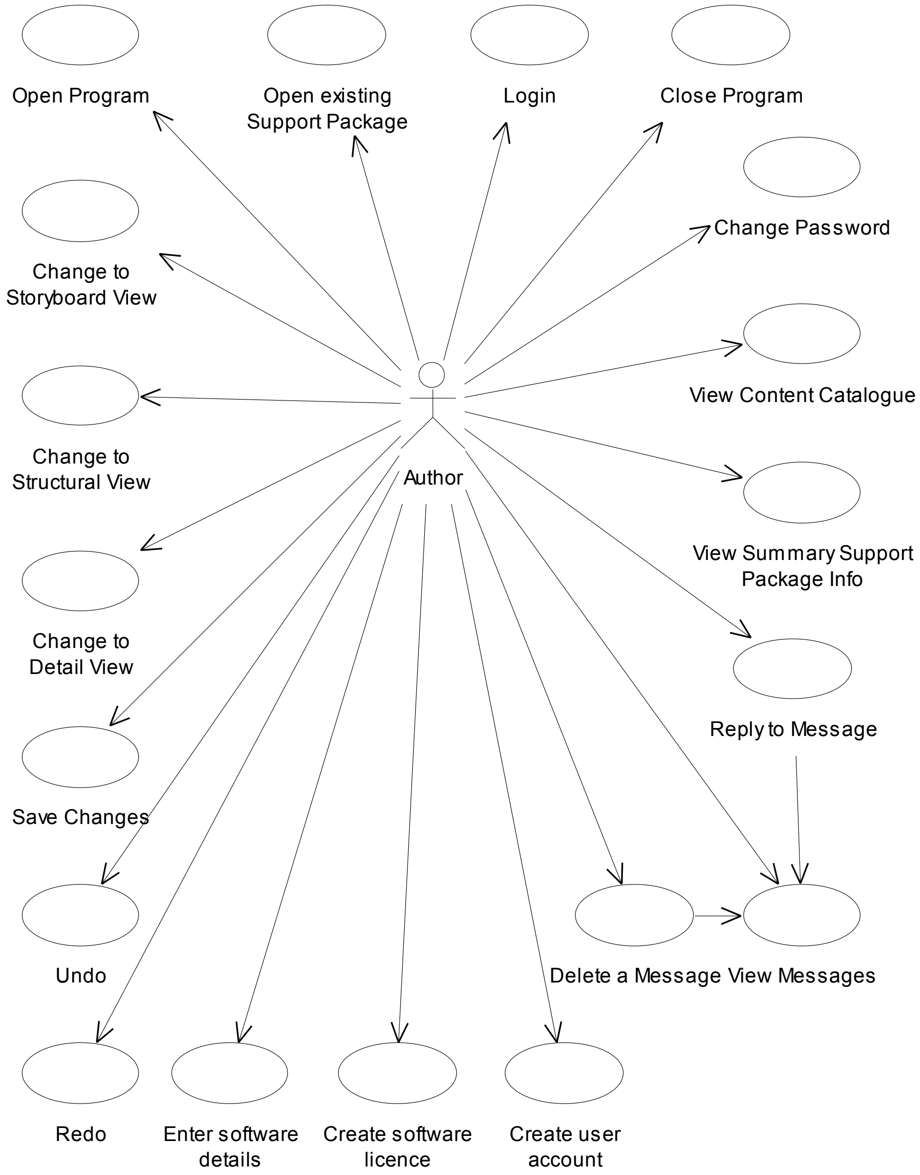


Figure 4.3: Author administrative task use cases

Figure 4.3 shows the use cases related to administrative tasks a user might wish to perform. These are tasks that do not directly involve the authoring of a training course but are required in order to make courses available to appropriate users and to effectively run the authoring tool. The details of four of these use cases are described below. The full description of all use cases can be found in Appendix B.

Open Existing Support Package

Basic Flow:

Precondition - authoring program has just been opened.

1. Select "open existing support package" from the initial dialog box.
2. Select the file from the file chooser that appears.
3. Press open.

Alternate Flow 1:

Precondition - authoring program is already open e.g. no initial dialog box open.

1. Select the file menu.
2. Choose open...
3. Select the support package file from the file chooser that appears.
4. Press open.

Alternate Flow 2:

Precondition - authoring program is already open e.g. no initial dialog box open.

1. Select the file menu.
2. Choose one of the recent support package titles from the list at the bottom of the menu.

Create User Account

Basic Flow:

1. Select Tools menu.
2. Select Create User Account. The User Account Maintenance window appears.
3. Enter the personal details of the user, including name and role, as well as their user ID and initial password.
4. Select the software that the user requires an support account for. This is from a list of already entered software.
5. Select the software licence i.e. the licence for the company the user is employed by.

6. Select the user group that this user fits into e.g. viewer, administrator.
7. Press OK or cancel.

Alternate Flow 1:

1. Select Tools menu.
2. Select Create User Account. The User Account Maintenance window appears.
3. Enter the personal details of the user, including name and role, as well as their user ID and initial password.
4. Select create new software (because this software is not on the list).This calls the Enter Software Details use case.
5. Select create new software licence (because there is not currently a licence in the system for this company to use the software). This calls the Create Software Licence use case.
6. Select the user group that this user fits into e.g. viewer, administrator.
7. Press OK or cancel.

Enter Software Details

Basic Flow:

Precondition - the user is in the process of creating a user account but the relevant software has not already been recorded in the system.

1. Enter the details of the software, including name, version, author(s), platform, language, edition etc.
2. Choose Add or cancel.

Create Software Licence

Basic Flow:

Precondition - the user is in the process of creating a user account but a software licence for the user's company has not already been recorded in the system.

1. Select the software from a pre-defined list.
2. Enter the company details and the licence terms e.g. time period valid, number of concurrent users etc.
3. Choose Add or cancel.

4.6.2 Course Structure Creation Task Use Cases

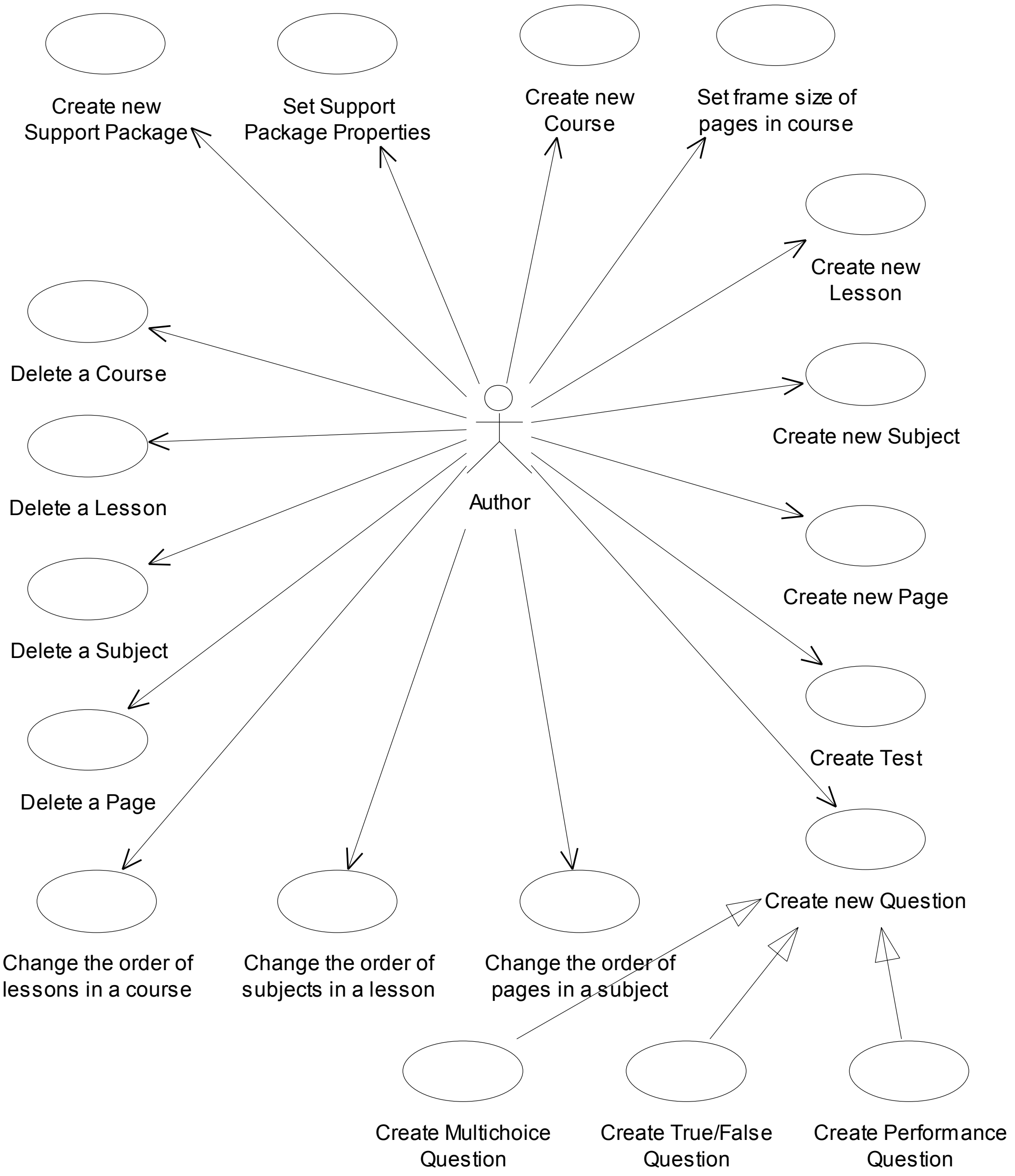

Create new Support Package
Set Support Package Properties
Create new Course
Set frame size of pages in course
Create new Lesson
Delete a Course
Create new Subject
Delete a Lesson
Author
Create new Page
Delete a Subject
Create Test
Delete a Page
Create new Question
Change the order of lessons in a course
Change the order of subjects in a lesson
Change the order of pages in a subject
Create Multichoice Question
Create True/False Question
Create Performance Question

Figure 4.4: Author course structure creation task use cases

Figure 4.4 shows the task involved in authoring training course structure. This does not include authoring content. Following are the full use case flow descriptions of six of these tasks. For details of all use cases, see Appendix B.

Create New Support Package

Basic Flow:

1. Open the authoring program
2. Select "create new support package" from the options given in the initial dialog box.
3. Give the support package a file name and directory.
4. Give the support package a title.

Alternate Flow 1:

Precondition - the authoring program is already open.

1. Open the File Menu.
2. Select "new".
3. Select "support package".
4. Give the support package a file name and directory.
5. Give the support package a title.

Create New Course

Basic Flow:

Precondition - the parent support package must already exist.

1. Select the parent support package in the structural view.
2. Choose the file menu and the new submenu.
3. Select "course".
4. Enter the title of the course.

Alternate Flow 1:

1. Select the parent support package in the structural view.
2. Right click on the selected node in the structural view.
3. Select new course from the pop-up menu.
4. Enter the title of the course.

Create New Lesson

Basic Flow:

Precondition - the parent course must already exist.

1. Select the parent course in structural view.
2. Choose the file menu and the new submenu.
3. Select "lesson".
4. Enter the title of the lesson.

Alternate Flow 1:

1. Select the parent course in structural view.
2. Right click on the selected node.
3. Select new lesson from the pop-up menu.
4. Enter the title of the lesson.

Create New Subject

Basic Flow:

Precondition - the parent lesson must already exist.

1. Select the parent lesson in structural view.
2. Choose the file menu and the new submenu.
3. Select "subject".
4. Enter the title of the subject.

Alternate Flow 1:

1. Select the parent lesson in structural view.
2. Right click on the selected node.
3. Select new subject from the pop-up menu.
4. Enter the title of the subject.

Create New Page

Basic Flow:

Precondition - the parent lesson must already exist.

1. Select the parent lesson node in structural view.

2. Choose the file menu and the new submenu.
3. Select "page".
4. Enter the title of the page.

Alternate Flow 1:

1. Select the parent lesson node in structural view.
2. Right click on the highlighted lesson node.
3. Select new page from the pop-up menu.
4. Enter the title of the page.

Change the Order of Lessons in a Course

Basic Flow:

Precondition - the course must be open and visible in structural view. It must also be expanded so that all lessons are visible.

1. Click down on the lesson that you want to move.
2. Drag the lesson node to the position where it is to go.
3. Drop the lesson node.
4. The titles of all lessons are recalculated e.g. change from "Lesson 1: <lesson name>" to "Lesson 3: <lesson name>" also causes lesson 2 -> lesson 1 and lesson 3 -> lesson 2.

Create New Test

Basic Flow:

Precondition - lesson must be selected in structural view.

1. Select File menu and new submenu.
2. Select Test. The Test Designer window will open and show that the test is for the highlighted lesson.
3. Choose to add a new question. This calls the Create new Questions use case.
4. Repeat step 3 any number of times.
5. Enter the threshold for passing the test.
6. Press OK or cancel.

4.6.3 Author Content Design Task Use Cases

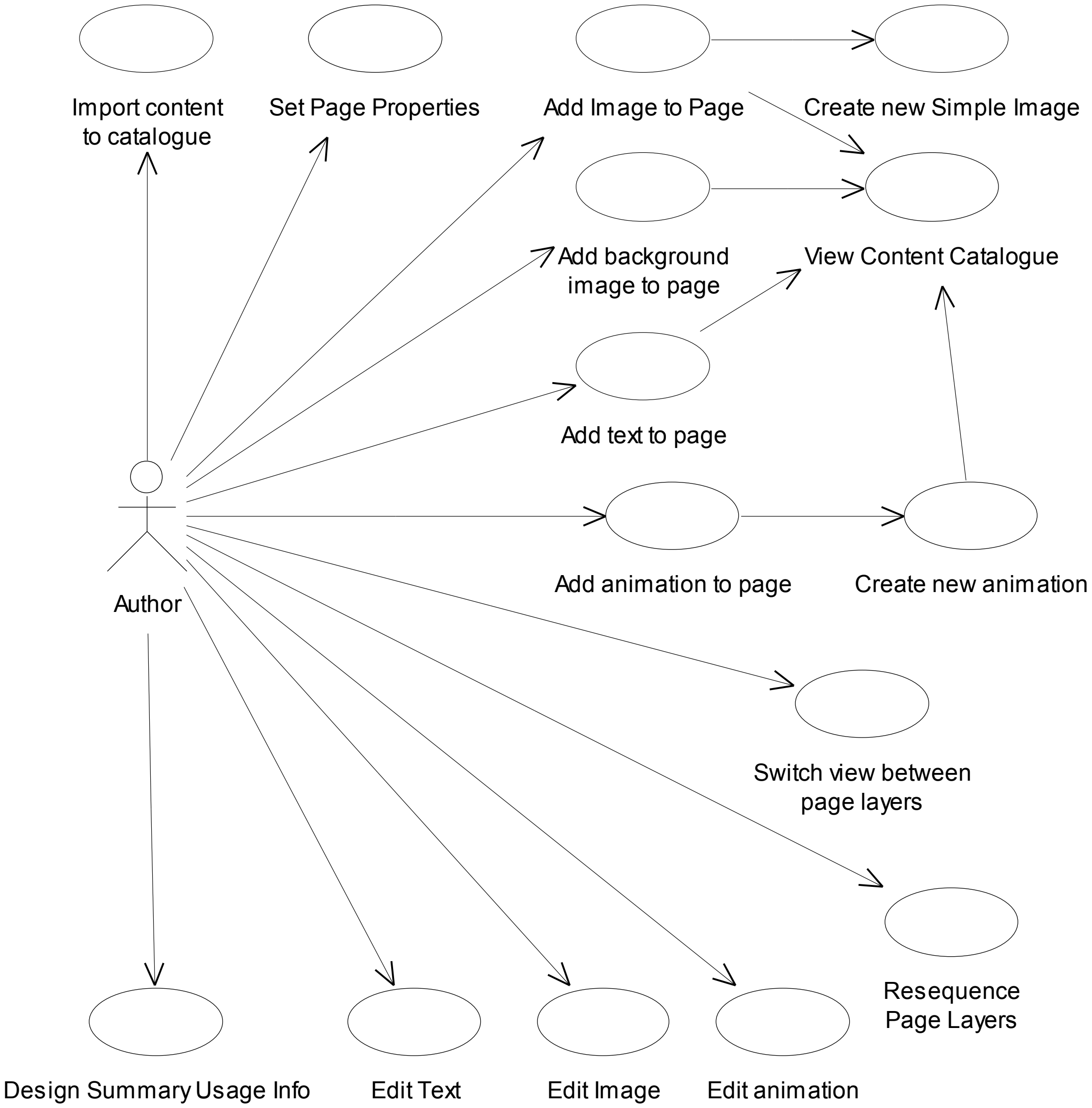


Figure 4.5 Author content design task use cases

Figure 4.5 shows the tasks related to authoring course content, the media viewable on course pages. This section describes six of the more interesting tasks in detail. For details of all tasks, see Appendix B.

Add Image to Page

Basic Flow:

Precondition - page node must be selected in structural view and therefore visible in detail view.

1. Select Insert menu.
2. Choose simple image.
3. Select "from catalogue".
4. Select the image wanted from the catalogue.
5. Press Insert.

Alternate Flow 1:

Precondition - page node must be selected in structural view and therefore visible in detail view.

1. Create new Simple Image

Create New Simple Image

Basic Flow:

Precondition - a page node is selected in structural view and therefore visible in detail view.

1. Select the Insert menu.
2. Choose simple image.
3. Choose new (as opposed to from catalogue).
4. Choose between the standard shapes available (circle, rectangle, line, arrow) and choose a colour.
5. Press Insert (the window doesn't close so that the author can easily select multiple shapes if they want).
6. The image appears on the page.
7. Author adjusts size and position and can set the order of objects on the page if they want.

Alternate Flow 1:

1. Right click on the page node in structural view.
2. Choose add simple image.
3. Choose new (as opposed to from catalogue).
4. Choose between the standard shapes available (circle, rectangle, line, arrow) and choose a colour.
5. Press Insert (the window doesn't close so that the author can easily select multiple shapes if they want).
6. The image appears on the page.

7. Author adjusts size and position and can set the order of objects on the page if they want.

Add Animation to Page

Basic Flow:

Precondition - page must be selected in structural view and therefore visible in detail view.

1. Select Insert menu.
2. Select animation. The Animation Designer window opens.
3. <Use Case> create new animation.
4. Select OK or cancel.

Create New Animation

Basic Flow:

1. Choose to add an image to the animation. This uses the View Content Catalogue use case.
2. Select an image from the catalogue.
3. Repeat steps 1 and 2 as often as needed to include all desired images.
4. Drag and drop images to achieve desired sequence. Default sequence is the order the images were added.
5. Add trigger to each image in the sequence. This could be either a timer e.g. 5 seconds after previous image, or a user action e.g. click on page in certain place, right click, double click etc.

Edit Text

Basic Flow:

(For editing the content of the text itself)

1. Select the text box in detail view of the page.
2. Edit text.
3. Click elsewhere on the interface.

Alternate Flow 1:

(For editing the properties of the text, not the text itself)

1. Select the text box in detail view of the page.
2. Change any of the values in detail view using the Text Options controls e.g. font, style, size, colour.
3. Click elsewhere on the interface.

Switch View Between Page Layers

Basic Flow:

Precondition - a page must be highlighted in structural view and visible in detail view.

1. From the list of layers in the Page Properties section of the detail view, select the layer to view.

4.7 Viewer Use Cases

4.7.1 Viewer Administrative Task Use Cases

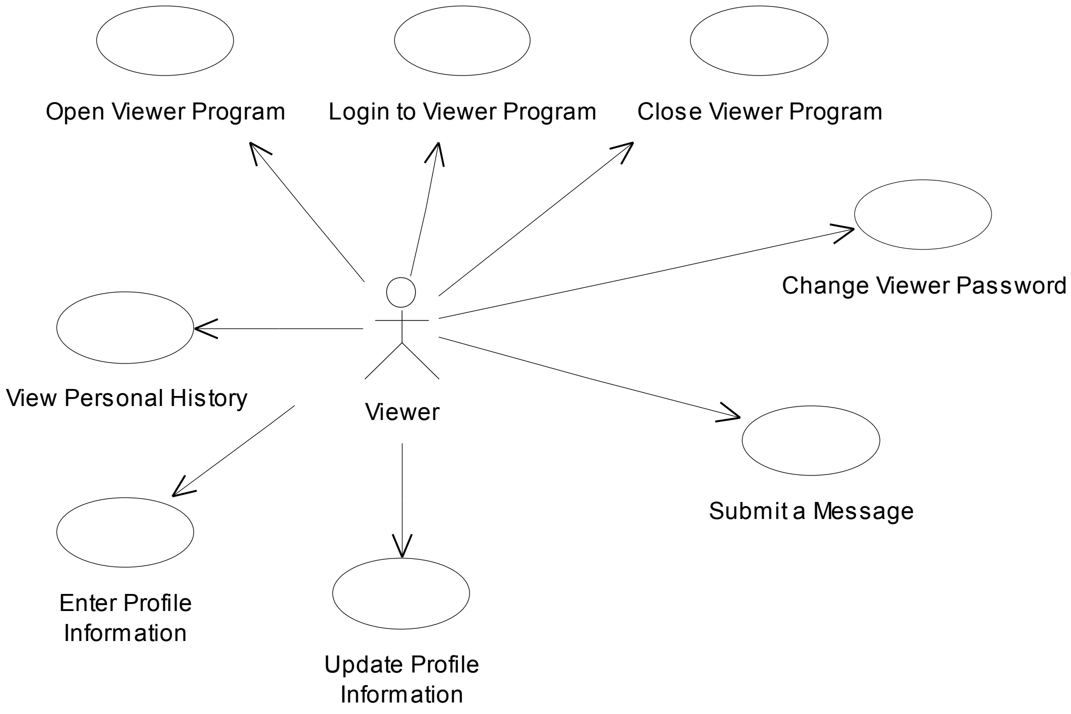


Figure 4.6: Viewer administrative task use cases

Figure 4.6 shows the administrative tasks a user of the viewing tool would perform. These are tasks that are peripheral to the actual viewing of a course.

Login to Viewer Program

Basic Flow:

Precondition - program has just been opened.

1. Enter user name.
2. Enter password.
3. Press enter or click login.

Close Viewer Program

Basic Flow:

1. Press exit
2. Answer yes to prompt to exit.
3. The application and the currently open support package closes.

Alternate Flow 1:

1. Press exit
2. Answer yes to prompt to exit.
3. The application remains open.

Change Viewer Password

Basic Flow:

Precondition - user must already be logged into the program.

1. Go to the personal information section.

3. Enter current password.

4. Enter new password twice.

5. Press OK (or cancel).

View Personal History

Basic Flow:

Precondition - already logged into the viewing program.

1. Open the course that you want the history for. This calls the Open a Course use case.
2. Select Personal History. This would replace the lesson menu of the course as the content of the main window. Personal History contains both lessons completed and test results.

Submit a Message

Basic Flow:

Precondition - user is on the last page of a lesson.

1. Select "submit a comment or question about this lesson".
2. Enter the message text.
3. Press submit or cancel.

Alternate Flow 1:

Precondition - user has just completed the last component of a course (not necessarily in order).

1. Choose to "submit a comment or question about this course" when prompted.
2. Enter the message text.
3. Press submit or cancel.

Alternate Flow 2:

Precondition - user is on the main course selection window or the main lesson selection window of a course.

1. Select "contact support staff".
2. Enter the message text.
3. Press submit or cancel.

Enter Profile Information

Basic Flow:

Precondition - it is the user's first time using the application.

1. Agree with the dialog prompting for profile information.
2. Enter values for all of the variables given.
3. Press OK or cancel.

Alternate Flow 1:

Precondition - not first time using the application (no dialog prompt is open).

1. See Update Profile Information use case.

Update Profile Information

Basic Flow:

1. Select Personal Information.
2. Choose to enter profile information.
3. Enter values for all of the variables given.
4. Press OK or cancel.

4.7.2 Viewer Support Viewing Task Use Cases

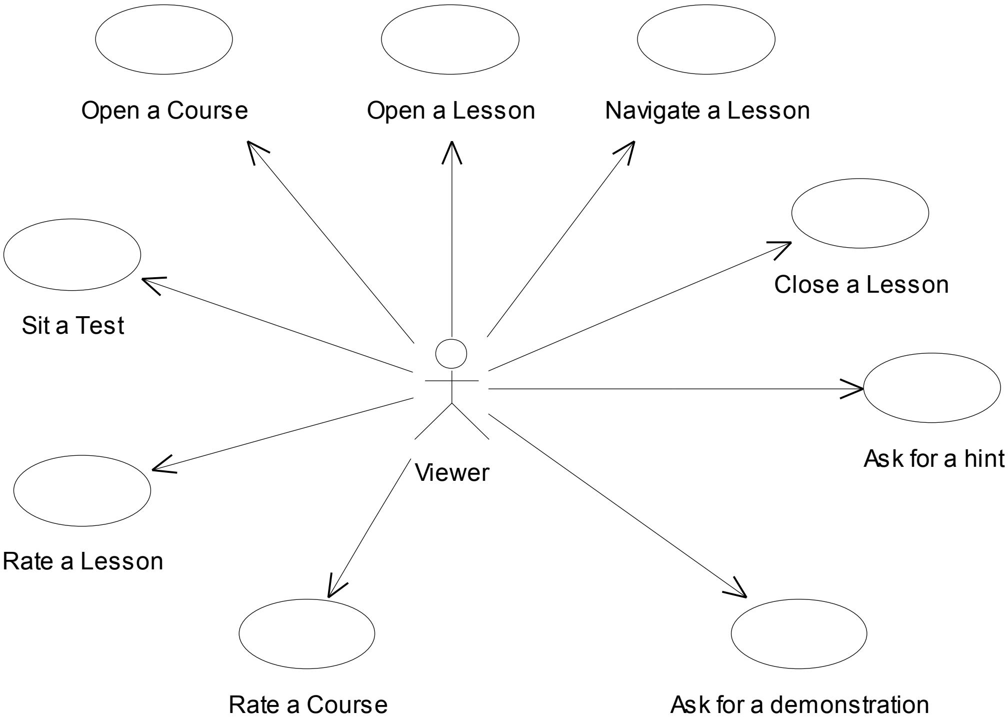


Figure 4.7: Viewer training course viewing task use cases

Figure 4.7 shows the tasks a user would perform in viewing a training course. This section describes the flow of these tasks in detail.

Open a Course

Basic Flow:

Precondition - user is on the initial page - the software support package selection page.

1. From the list of software that the user is registered for, select the desired title.
2. The list of software will be replaced by the menu for that support package e.g. course, help, docs etc.
3. Select the course component of the package from the menu.
4. The package menu will be replaced by the course menu e.g. lessons.

Open a Lesson

Basic Flow:

Precondition - the user has already opened a course.

1. Select a lesson. Lessons do not have to be completed in order and can be run any number of times. The lesson will appear in a separate window and the main window will be minimised.

Navigate a Lesson

Basic Flow:

Precondition - a lesson is open.

1. Read the content on the current page.
2. Choose "next page".

Alternate Flow 1:

Precondition - the current page contains an animated demonstration or an audio component.

1. Read the content on the current page.
2. Choose "replay".

Alternate Flow 2:

Precondition - the current page is the last in the lesson.

1. Read the content on the current page.
2. Choose "exit lesson".

Close a Lesson

Basic Flow:

Precondition - a lesson is currently open.

1. Press the close window button or select Close Lesson.
2. Answer yes to prompt to exit lesson.
3. If the lesson was completed, this is reflected on the main window by a change in colour of the lesson title.

Sit a Test

Basic Flow:

Precondition - the course is open and the lesson menu is visible.

1. Select a test from the menu.
2. Answer each question.
3. Submit the test.

Alternate Flow 1:

Precondition - the user is at the personal history section.

1. Select the test from user's personal history (e.g. one that has already been sat).
2. Choose "re-sit".
3. Answer each question.
4. Submit the test.

Alternate Flow 2:

Precondition - user reaches a test page at the end of a lesson.

1. Answer each question.
2. Submit the test.

Rate a Lesson

Basic Flow:

Precondition - the user has just finished a lesson.

1. Choose "rate this lesson".
2. Choose between the offered ratings.
3. Choose submit or cancel.

Rate a Course

Basic Flow:

Precondition - user has just completed the last component of a course (not necessarily in order).

1. Choose to "rate this course" when prompted.
2. Choose between the offered ratings.
3. Choose submit or cancel.

Ask for a Hint

Basic Flow:

Precondition - user is on a demonstration page.

1. Select "give me a hint".
2. Press OK to close the hint.

Ask for a Demonstration

Basic Flow:

Precondition - the user must be on a demonstration page.

1. Choose "show me".
2. Press "play".
3. Close the demonstration.

4.8 Summary

This chapter presented a detailed specification of the training course authoring, delivery and monitoring system. It began by presenting an overview of the goals and general functionality of the system. It then described in detail the specification of the authoring and viewing tools, including a general overview, descriptions of necessary interface features and key functionality. The types of reciprocal user feedback and intended uses of this information, necessary to support iterative authoring, were then described. Finally, use case diagrams and explanations of these for the functionality of both the authoring and viewing tools were given.

Chapter 5 Design

5.1 Introduction

This chapter presents a complete conceptual design of the system prototype. First, the conceptual object models used to describe training courses and all information stored about them and relevant to them, such as the course membership hierarchy, are described. Then the intended architecture of the system is outlined and the communication between components during a typical interaction is demonstrated using sequence diagrams. Finally, the user interfaces of the authoring and viewing tools are described.

5.2 Conceptual Object Models

5.2.1. Person and Software Information

This section details the relationships between, and the information associated with, the people who will be using the system and the software that will be taught/learned.

5.2.1.1 Classes

Anybody
This is more of a database design class than an object oriented one. It is an abstraction of a person and an organisation and is useful when there are relationships between Anybody and another class.

Person
A person can be either an employee of a software vendor organisation or an employee of a client organisation. Person inherits some of its properties from Anybody.

Organisation
An organisation can be either a software vendor organisation, in which case its related people will be authors, or a client organisation, whose people will be software users.

Role
A role is the specific role that a person performs for an organisation. This is only included to give uniqueness to each instance of the following class, Employment Contract. This is because it is possible for a single person to have

more than one employment contract to a single organisation, the difference being the role they were hired to perform. However, this is not likely to be included in any further stages.

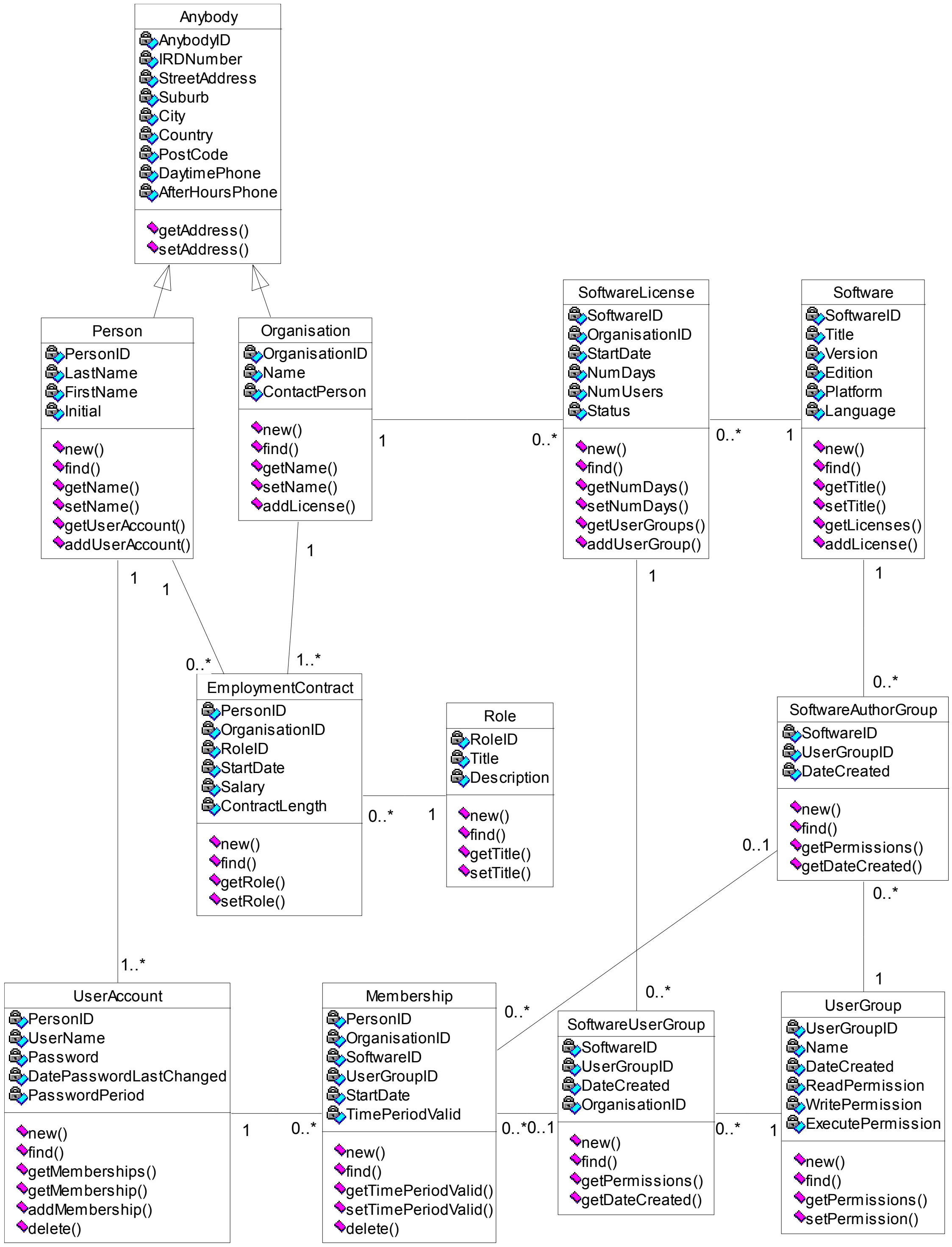


Figure 5.1: Person, software and membership hierarchy object model

Employment Contract

An employment contract symbolises the relationship between a person and an organisation. This is how authors are related to a software vendor organisation and how software users are related to a client organisation. This too may be unnecessary at any further stages in the design process.

Software

This class stores the details of each piece of software that a vendor organisation decides to build support material for.

Software License

A software license represents the contractual relationship between a client organisation (one who purchases the license to use the software) and a particular piece of software produced by the software vendor. A client organisation must first have a software license before new user accounts for their employees are created or before existing user accounts are updated with the permissions to use the support material for the particular piece of software.

User Group

There are a static number of user groups that will be unchangeable in the system. These are likely to be "Viewer", "Author", "Administrator" and "Root". These hold with them the specific permissions that a member of each user group will have i.e. read, write and execute.

Software User Group

This class represents the relationship between a license for a piece of software and a specific user group. It is present so that an administrator can create the fours standard user groups for a specific software license. Then user accounts can be given a membership to any one or more of these software license-specific user groups. This will prevent user accounts from being given broad access as simply a "Viewer" to all pieces of software and will also prevent an administrator from having to set individual permissions for each user account - instead they can just select the group the user account will belong to and set the permissions for the group as a whole.

User Account

Each user, whether a viewer on the client side or an author on the vendor side, will have at least one user account. It is possible to have more than one but not the norm. A user account will allow all users to login to their corresponding program and have information they enter retained under their specific system alias.

Membership

A membership is the relationship between a user account and a software user group. In this sense, a user belongs to any number of software users groups, which means they can access all corresponding software support material to the level that the user group permissions allow. If an existing client organisation obtained a second software license, it would not be necessary to make any changes to user accounts, only to create new software-specific user groups and give the existing user accounts memberships to them.

5.2.1.2 Relationships

The relationship between Person and Anybody and Organisation and Anybody is one of abstraction. A Person is an Anybody and an Organisation is an Anybody. There is another relationship between Person and Organisation that represents employment and that is the Employment Contract class. A Person can have zero or more Employment Contracts and each Employment Contract is for a single Person. The cardinality is similar on the other side of the relationship, where an Organisation can have one or more Employment Contracts and each Employment Contract is for a single Organisation. Employment Contract also has a relationship with Role, where each Employment Contract is for a single Role, and obviously each role can have zero or more Employment Contracts.

Each piece of Software entered into the system can have zero or more Software Licenses issued for its use. A Software License gives an Organisation use of just that one piece of Software. The Software License is related to a single Organisation, the client, or company who has purchased the license. Each Software License is also related to zero or more Software User Groups, which are discussed in more detail below.

Pre-programmed into the system are several User Groups. Each of these groups relates to a particular set of users: Viewers, Authors, and Administrators. When an Organisation purchases a Software License, an administrator would assign User Groups to this Software License to specify the types of user accounts that the client organisation can set up. For instance, if the client was going to have all of its accounts set up and maintained by the software vendor organisation, there would be no need for them to have any administrator accounts, so there would only be a single Software User Group created for them, to allow Viewer users. If they wanted to maintain their own user accounts, then they would get two Software User Groups, one for Viewers and the other for Administrators.

Author permissions are handled in a similar way, with an administrator creating at least one Software Author Group for each piece of Software. This would have read, write and execute permissions, so that the original author could effectively create the support material designs. Optionally, there could be other Software Author groups set up so that authors could read and execute but not write to another author’s designs. As with viewers, a Software Author Group is for a single piece of Software and a single User Group.

Once Software User and Author Groups have been established, it is simply a case of assigning User Accounts a membership to the appropriate group(s). This will give that user account the permissions that it requires to use the system within the appropriate limitations. A Person may have more than one User Account but a User Account is for a single Person. A User Account may have many Memberships, e.g. to the Symphonia Message Mapper Viewer group and to the Microsoft Word Viewer group, but a Membership is for a single User Account.

5.2.2. Support Package Information

This section details the relationships between, and the information associated with, the support material that can be created using the system, focusing mainly on training courses.

5.2.2.1 Classes

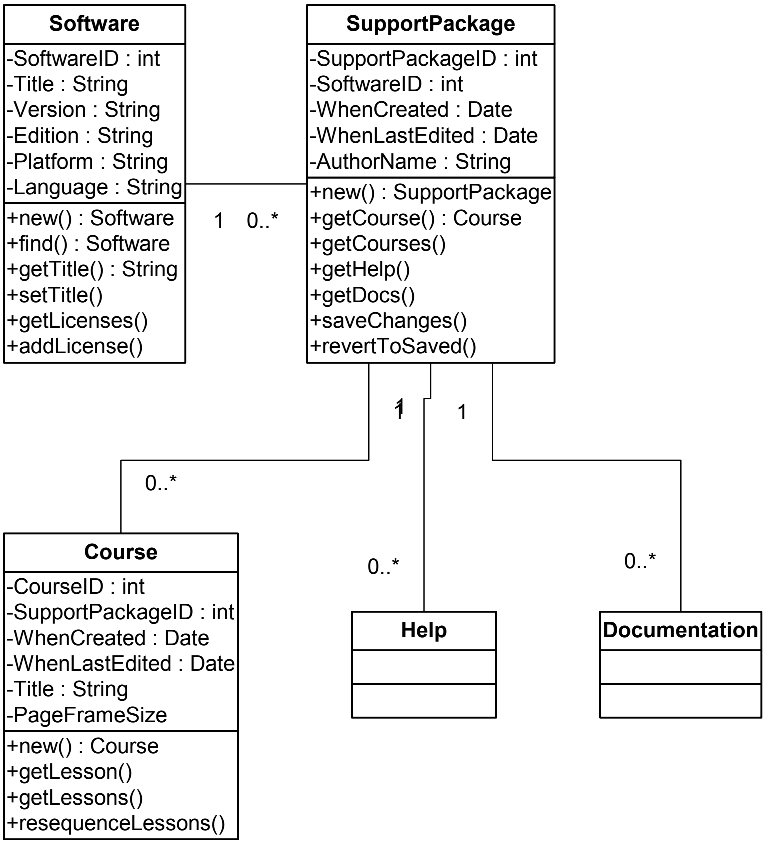


Figure 5.2: Support package object model

Support Package

This is the parent class for all of the support material that an author creates for a single piece of software. It may therefore contain one or more training courses, help and electronic documentation sections.

Software

This class represents a piece of software that the vendor has created and wants to build support material for.

Help

This is a component of a support package for a piece of software. The idea was to allow users to browse the traditional help section that is incorporated with most software systems from inside the support viewer. This has not been detailed because the focus at this stage is on the training course.

Electronic Documentation

This is a component of a support package for a piece of software. The idea was to allow users to browse the traditional electronic documentation available with many software systems from inside the support viewer. This has not been detailed because the focus at this stage is on the training course.

Course

This is designed by an author to provide training for a particular piece of software. It may contain lessons, subjects, and tests, with all information displayed on pages.

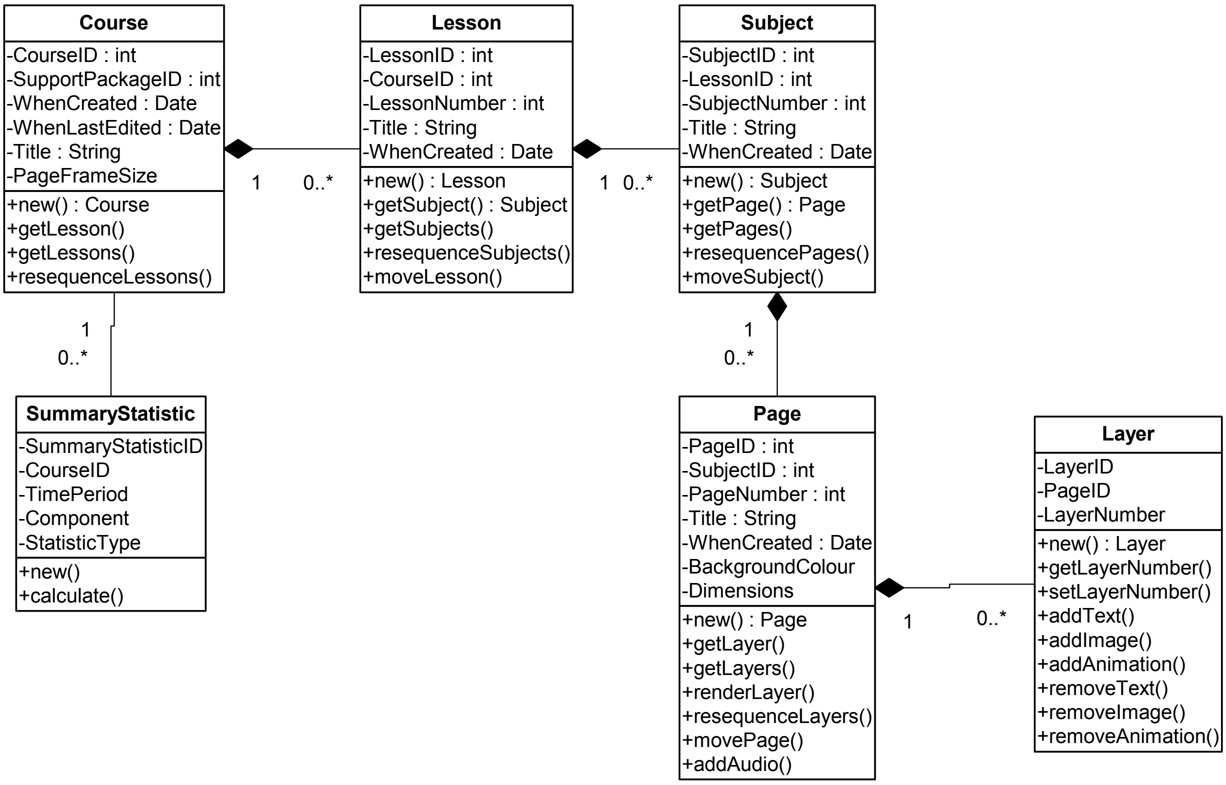


Figure 5.3: Course structure object model

Lesson

This is a component of a course. Each lesson should cover a single high-level user task. An example of this is "Creating a new message map".

Subject

This is a component of a lesson. A lesson is made up of one or more subjects. A subject should cover a single atomic task, which is a subtask of the topic covered by the parent lesson. An example is "Opening a new message map file".

Summary Statistic

This class stores information about what type of summary usage information an author wants to receive for a course they have authored.

Page

This is the building block of a course. Each subject is made up of pages and this will be what the user sees when they view the course.

Test Page

This is a specific type of page that displays question and takes in user input in the form of answers. It is a subtype of Page.

Procedure Page

This is a specific type of page that teaches a procedure. It would display the subtasks of the procedure and wait for the user to complete each subtask before moving on to the next. It is a subtype of Page.

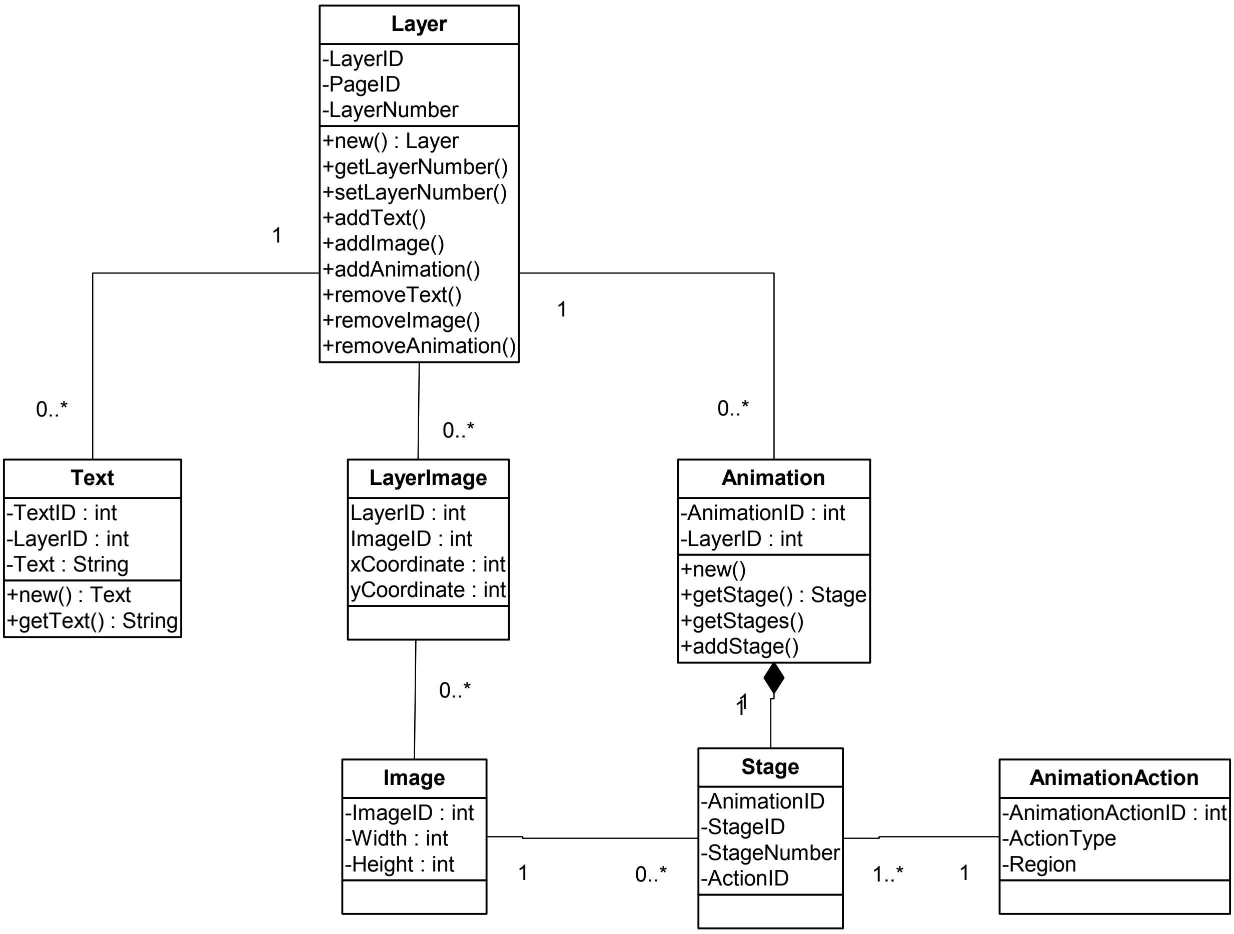


Figure 5.4: Page content structure object model

Layer

This class is used to represent the order of media on a page. Each piece of media, when added to a page, is added inside a layer. Layers can be re-sequenced to change the ordering of media on a page.

Text

This is a media component, which can be added to a page. Unlike audio, text can be imported or created using this system.

Animation

This is a media component that can created using this system and added to a page. Typically, an animation would only be added to a procedure page to demonstrate the steps involved in performing a task with the software being taught. An animation is made up of images and the user action that will take the user to the next image. An image and its corresponding action make up an animation Stage.

Stage

This is a component of an animation. Each animation is made up of one of more stages, which consist of an image (a screenshot a software user interface) and a user action that will move the animation to the next stage e.g. double click, single click, right click in a certain position on the image.

Image

This is a media component that can be added to a page. Image has two subtypes, Background Image and Simple Image.

Background Image

This is an image that has been chosen to be the background image for a page in the course. This means that it would be stretched to fit the shape and size of the page.

Simple Image

This is an image that can be created within this system. It is restricted to a certain number of simple shapes and colours.

Audio

This is a media component that can be added to a page. While media cannot be created using this system, it is planned that media files will be able to be imported into the content catalogue, and added to a page from there.

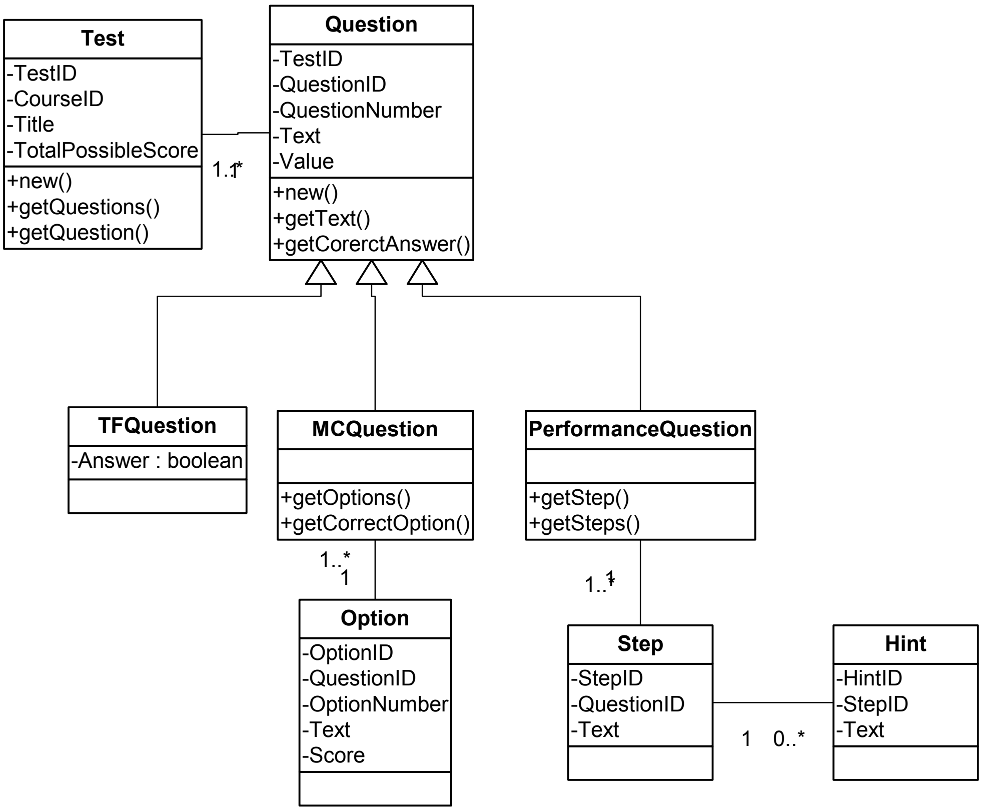


Figure 5.5: Testing object model

Test

This class represents a set of questions that relate to a particular lesson. A test is optional and can be sat at the end of a lesson.

Question

This is a component of a Test. Question is a supertype and has three subtypes: true/false question, multi-choice question and performance question.

TF Question

This is a subtype of Question. It gives a statement and asks for a true of false answer.

MC Question

This is a subtype of Question. It asks a question and then gives several incorrect and one correct option from which to choose for the answer.

Option

This class stores a single option for a single multi-choice question. The option may either be correct or incorrect.

Performance Question

This is a subtype of Question. It asks the user to perform a task without telling them how to do it, and monitors the actions they take for correctness.

Step

This is a component of both a Procedure Page and a Performance Question.

Hint

This class represents hint text that may be displayed to a user if they make a mistake or if they ask for a hint. A hint relates to a single step in a procedure.

5.2.3. Usage Information

This section details the information that will be stored as part of monitoring client use of the system.

5.2.3.1 Classes

Usage

This class represents a single usage of a component of a support package by a single user.

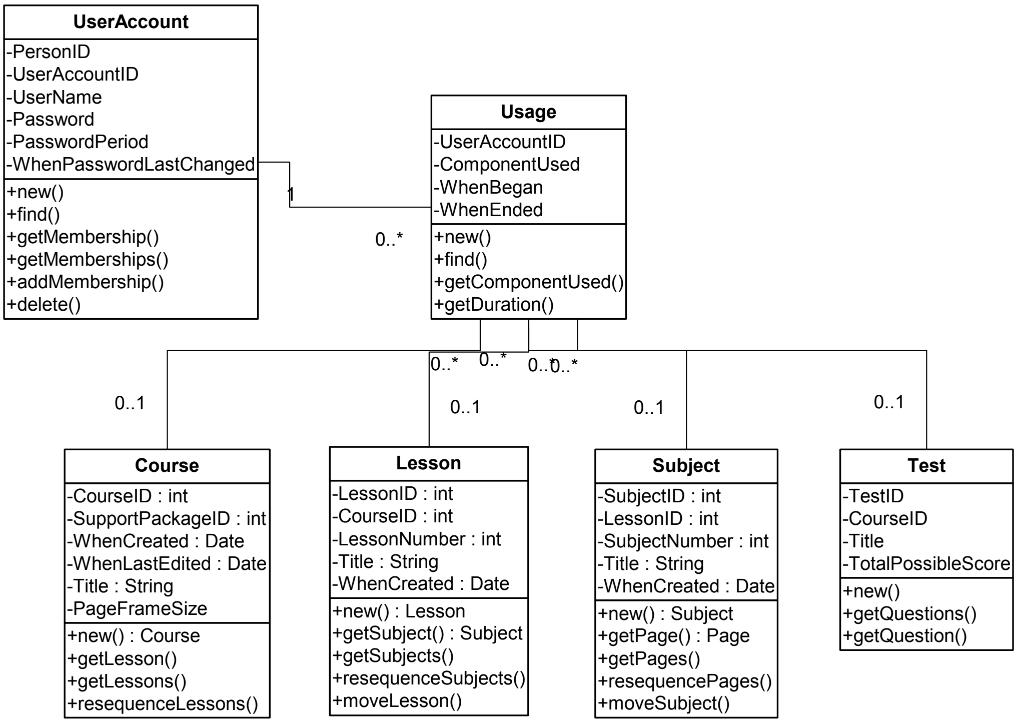


Figure 5.6: Usage history object model

Authoring Tool Server Server

5.3 Architecture

This section outlines the architectural design of the system. It begins by describing the main architectural components of the system and goes on to show how they interact during client-server communication.

Within the authoring tool, a client-server communication is initiated by the user, through interaction with one of the tool's graphical user interface components. This request is relayed to the Author object and then to the AuthorClient object. This then requests a service from the RemoteServer object, which acts as a proxy to the instance of AuthorServer dedicated to serving this particular Author client. The RemoteServer creates a new instance of a message object and writes it to a socket that the AuthorServer object is listening on. Each message object has a process() method, and when this is called it calls the appropriate method in the AuthorServer. The AuthorServer object relays the request to the MasterServer object, since this is responsible for all communication with the data store. This process is shown in Figure 5.7.

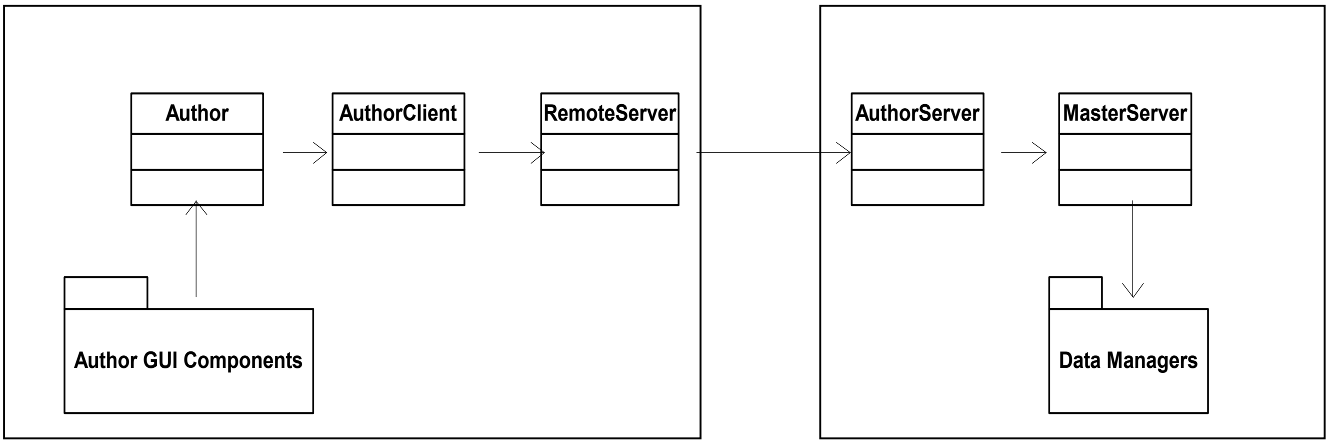


Figure 5.7: Interaction between architectural components during communication from authoring tool to server

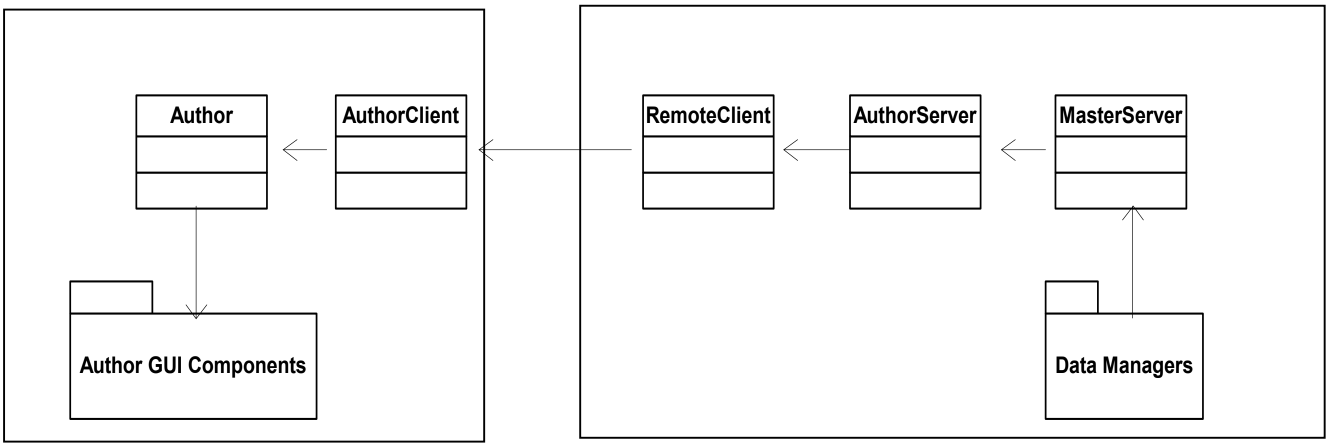


Figure 5.8: Interaction between architectural components during communication from server to authoring tool

A parallel but reverse process is carried out for communication from the server to the client side. This process is shown in Figure 5.8. To demonstrate the sequence of method calls involved in communicating a request from client to server using the architecture described above, Figure x describes the process of a user of the viewing tool logging in to the system.

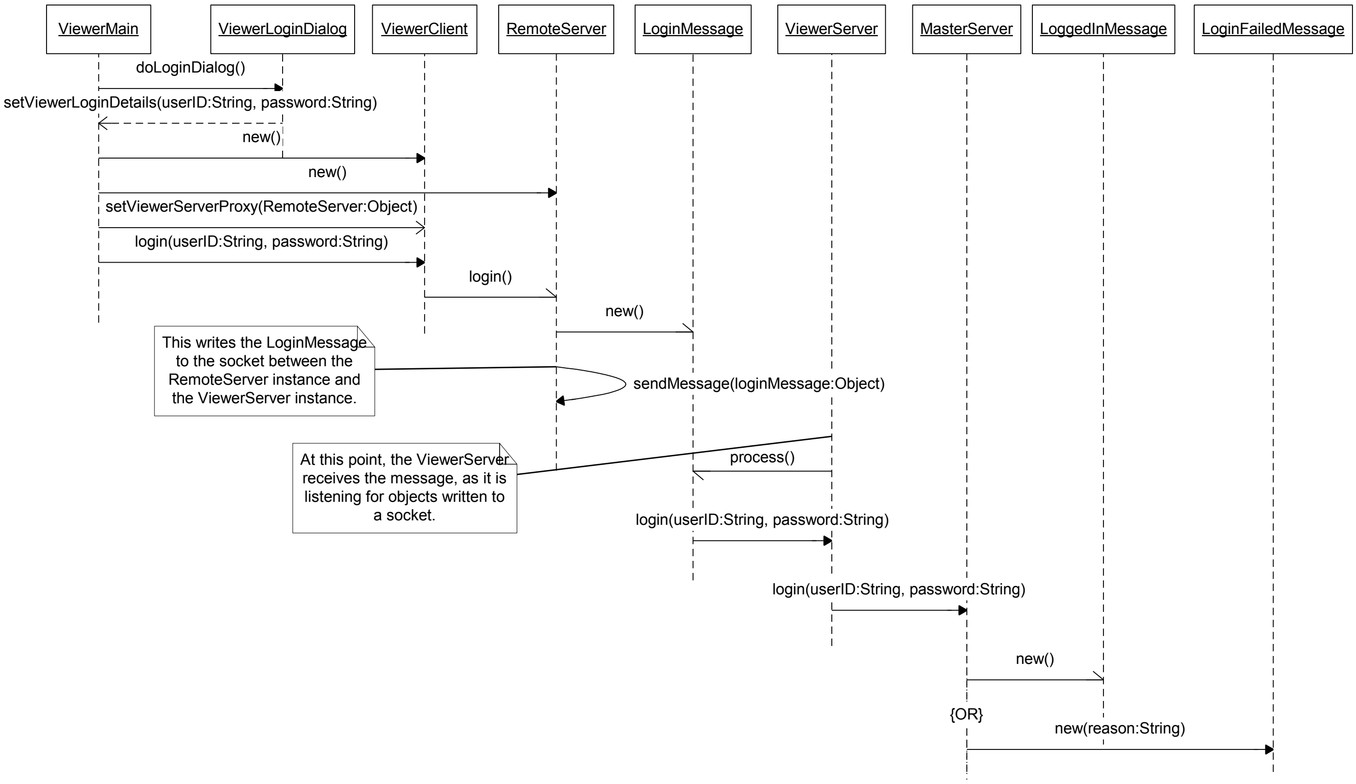
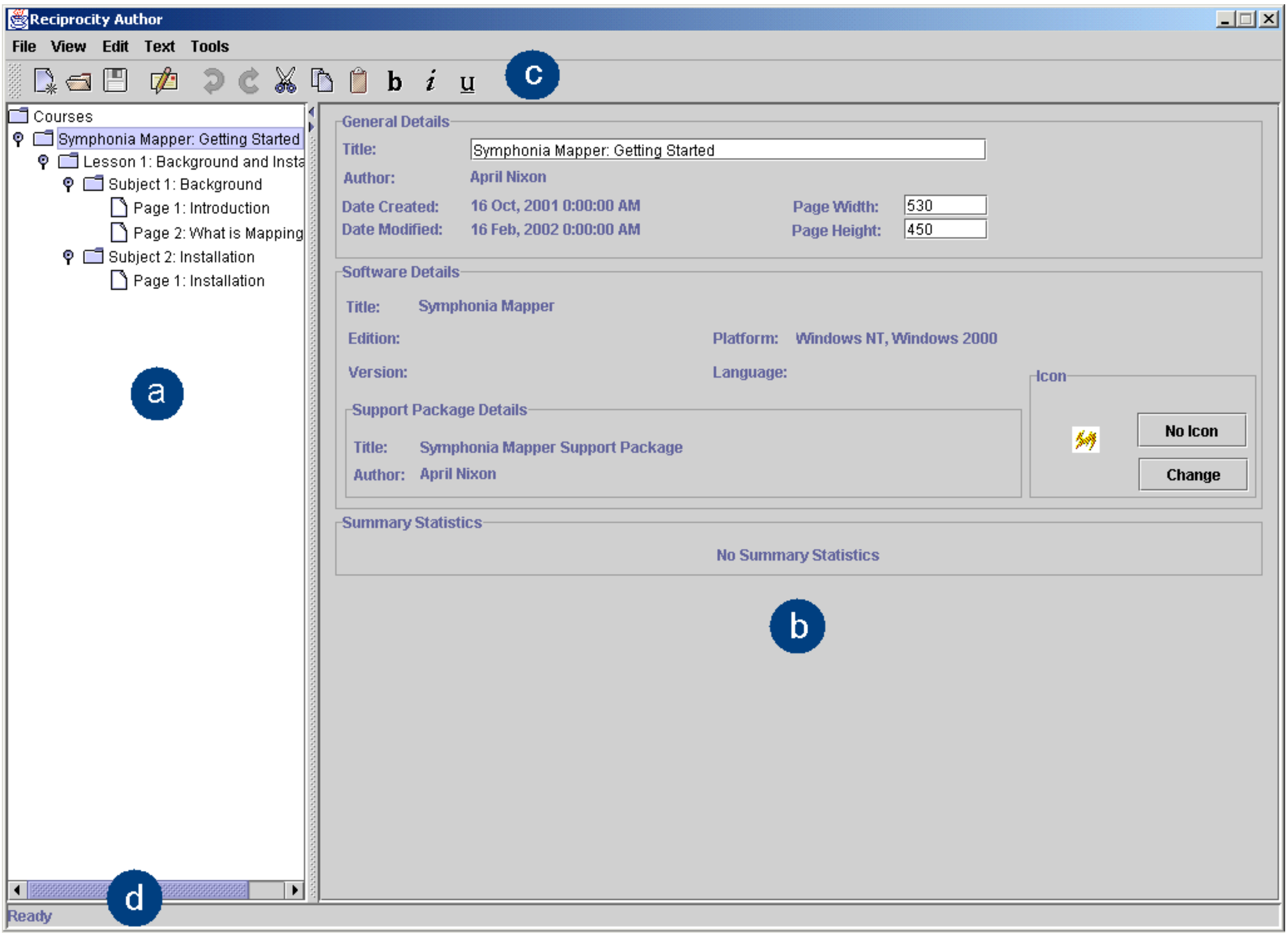

ViewerMain
ViewerLoginDialog
ViewerClient
RemoteServer
LoginMessage
ViewerServer
MasterServer
LoggedInMessage
LoginFailedMessage
doLoginDialog()
setViewerLoginDetails(userID:String, password:String)
new()
new()
setViewerServerProxy(RemoteServer:Object)
login(userID:String, password:String)
login()
new()
This writes the LoginMessage to the socket between the RemoteServer instance and the ViewerServer instance.
sendMessage(loginMessage:Object)
At this point, the ViewerServer receives the message, as it is listening for objects written to a socket.
process()
login(userID:String, password:String)
login(userID:String, password:String)
new()
{OR}
new(reason:String)

Figure 5.9: Sequence diagram showing method call interaction between components during login process

5.4 User Interface Design

5.4.1 Authoring Tool User Interface

The goal of the authoring tool user interface is to provide easy to use, graphical methods for building the structure and content of training courses. The information that needs to be displayed about a particular training course includes its structure, its properties and its content. It was decided that course structure should be viewable at all times, while a user should be able to switch between page design and properties modes, depending on the task at hand. Images of the authoring tool's user interface in both properties and page design modes are shown in Figures 5.10 and 5.11.



Courses
- Symphonia Mapper: Getting Started
  - Lesson 1: Background and Insta
    - Subject 1: Background
      - Page 1: Introduction
      - Page 2: What is Mapping
    - Subject 2: Installation
      - Page 1: Installation

General Details

Title: Symphonia Mapper: Getting Started

Author: April Nixon

Date Created: 16 Oct, 2001 0:00:00 AM

Date Modified: 16 Feb, 2002 0:00:00 AM

Page Width: 530

Page Height: 450

Software Details

Title: Symphonia Mapper

Edition:

Platform: Windows NT, Windows 2000

Version:

Language:

Icon

Support Package Details

Title: Symphonia Mapper Support Package

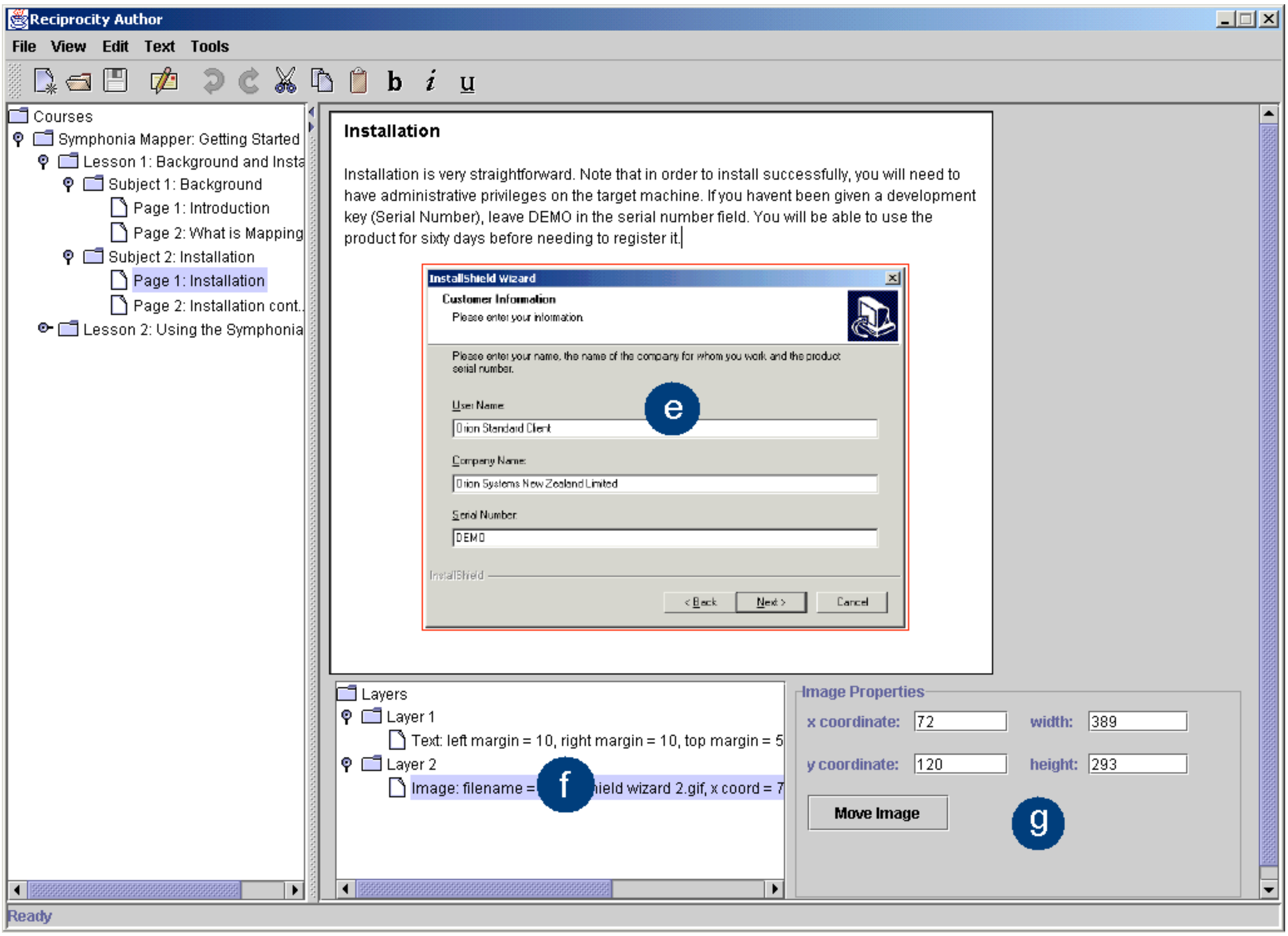
Author: April Nixon

Summary Statistics

No Summary Statistics

Ready

Figure 5.10: Authoring tool user interface – properties view



Installation

Installation is very straightforward. Note that in order to install successfully, you will need to have administrative privileges on the target machine. If you havent been given a development key (Serial Number), leave DEMO in the serial number field. You will be able to use the product for sixty days before needing to register it.

InstallShield Wizard

Customer Information

Please enter your information.

Please enter your name, the name of the company for whom you work and the product serial number.

User Name: Orion Standard Client

Company Name: Orion Systems New Zealand Limited

Serial Number: DEMO

InstallShield

< Back | Next > | Cancel

Text: left margin = 10, right margin = 10, top margin = 5

Image: filename = [f] ield wizard 2.gif, x coord = 7

x coordinate: 72 width: 389

y coordinate: 120 height: 293

Figure 5.11: Authoring tool user interface – page design view

a. *Structural View*. This section of the user interface represents the hierarchical structure of the currently opened course in the form of a tree. All operations that may be performed on the course structure, such as adding and deleting structural elements, are available from the popup menus of the nodes in the tree. The contents of the right hand side of the user interface, which switch between properties display and page design display are determined by the type of element selected from this structural view. If the element selected has no visual representation, such as the course itself, a lesson or a subject, the properties of this element are displayed. If the element does have a visual representation, such as a page or rating page, the right hand side of the user interface displays the page design components.

b. *Detail View – Properties Display*. This section of the user interface displays the properties of the structural element currently selected from the tree in the structural view. In Figure x, the properties of the course itself are displayed, since this is the node selected in the structural tree.

c. *Menu Bar and Toolbar*. The generically applicable functions of the user interface are available from these two sections of the user interface. Menus include File, View, Edit, Text and Tools.

d. *Status Bar.* This is always present at the bottom of the user interface and is used to keep users informed of the status of the system at all times. During down times when the application is waiting for user input, the status bar will read "Ready" as shown in Figure x. However, during operations that may take some time due to communication with the server, such as saving a course, the status bar will display an appropriate message, such as "Saving course…", until the operation is complete when it will again read "Ready".

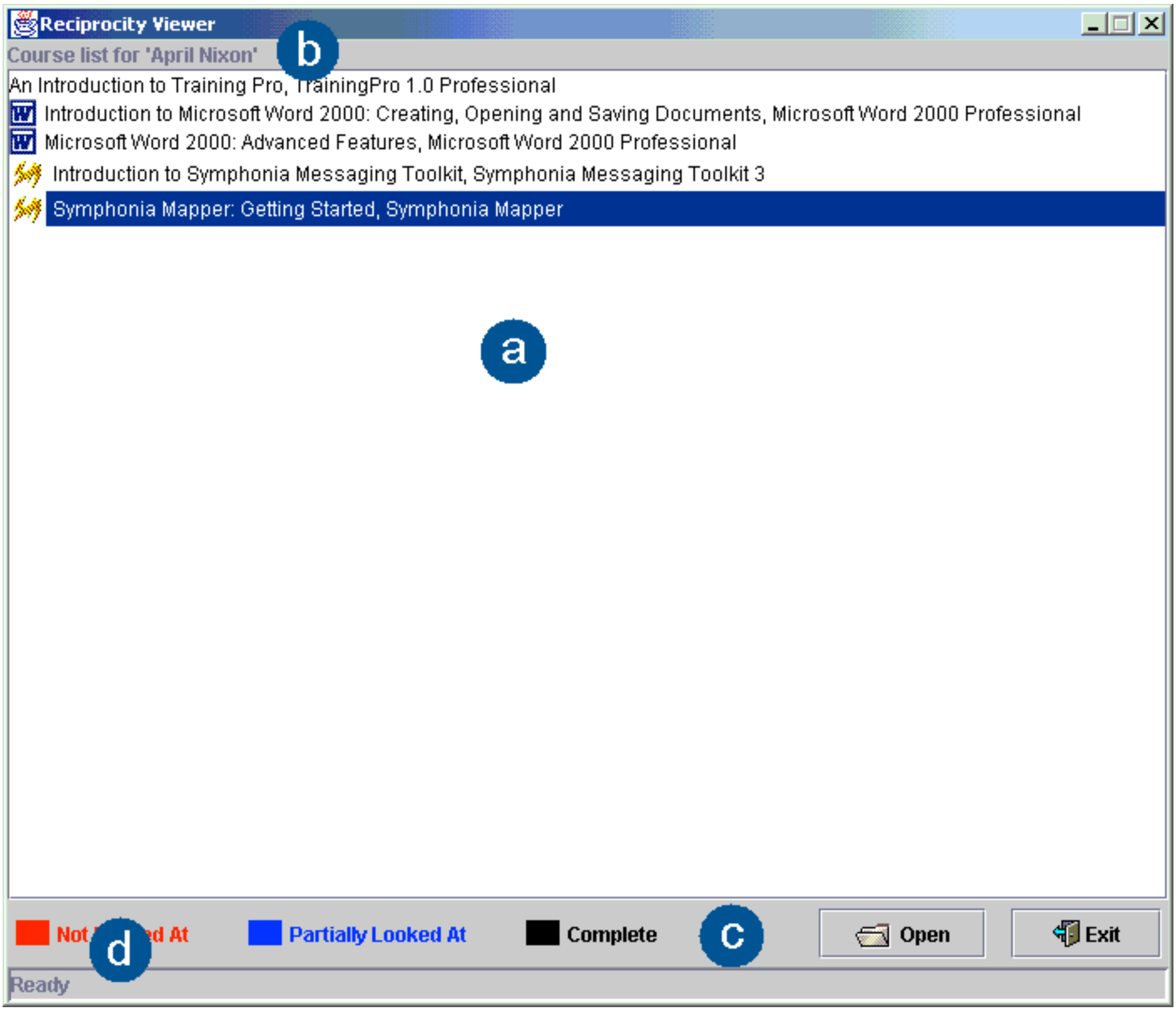
e. *Detail View – Page Design Canvas.* This component of the user interface detail view provides a direct visual representation of the pages of a course. This is exactly how pages will appear to users when the course is viewed using the viewing tool. Text, images and animations may be directly added to the canvas to build up a page's desired look.

f. *Detail View – Page Layer Tree.* Since a page is represented using three dimensions: width, height and depth, authors may build up layers upon a page, and this is displayed using a page layer tree. All layers and the media components contained within them area displayed in this tree.

g. *Detail View – Media Component Properties Display.* This section of the user interface displays the properties of the currently selected media element in the page layer tree. If none is selected, the properties of the page itself are displayed.

5.4.2 Viewing Tool User Interface

The goals of the viewing tool's user interface are to allow users to choose from a personalised list of courses produced using the authoring tool and then display those courses. This section provides images of the two main parts of the viewing tool's user interface, as shown in Figures 5.12, 5.13 and 5.14, and describes their components. Figure 5.12 shows the initial screen of the interface, which presents the list of courses, and Figures 5.13 and 5.14 show the interface used to view the courses.



Course list for 'April Nixon'

An Introduction to Training Pro, TrainingPro 1.0 Professional

Introduction to Microsoft Word 2000: Creating, Opening and Saving Documents, Microsoft Word 2000 Professional

Microsoft Word 2000: Advanced Features, Microsoft Word 2000 Professional

Introduction to Symphonia Messaging Toolkit, Symphonia Messaging Toolkit 3

Symphonia Mapper: Getting Started, Symphonia Mapper

Ready

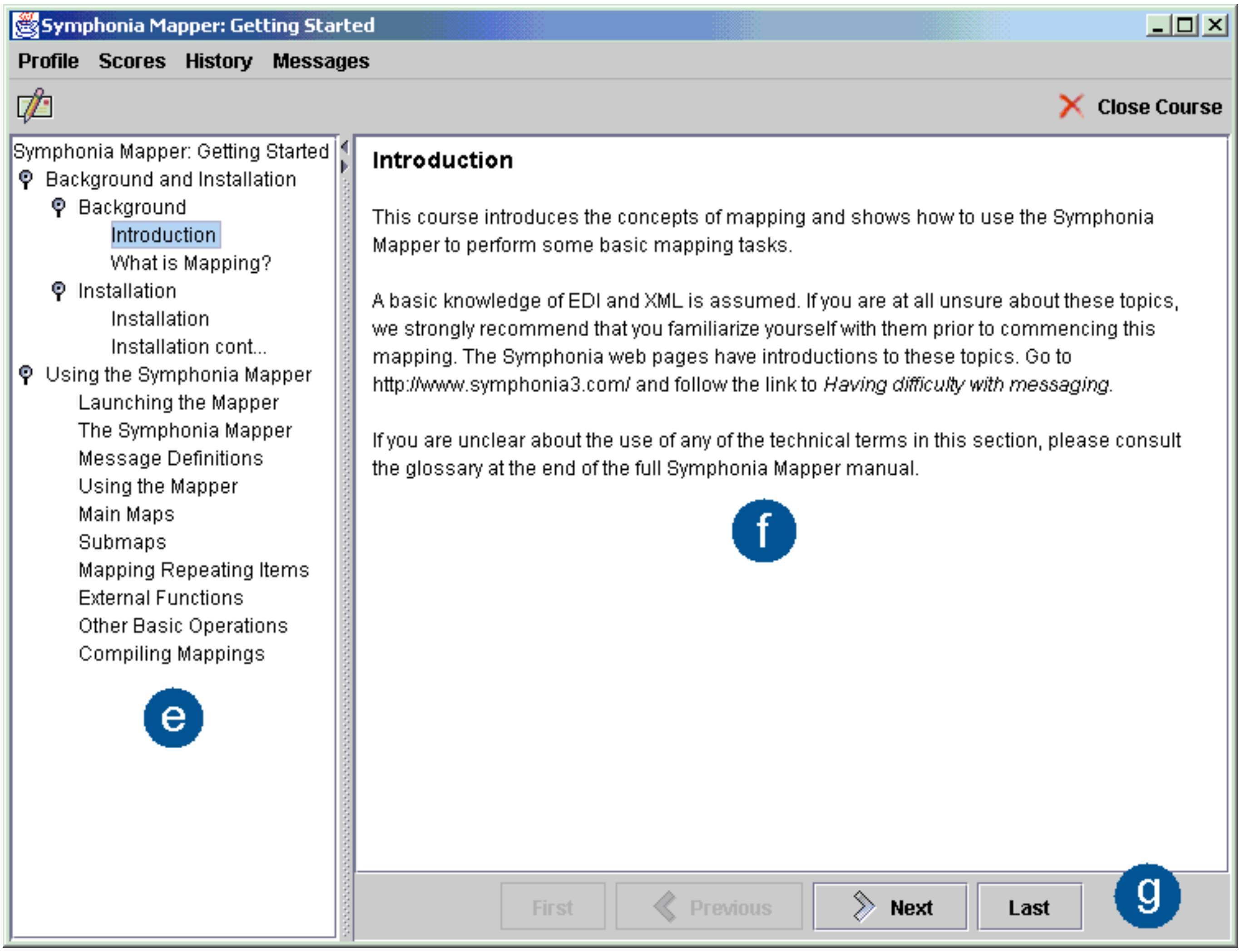
Figure 5.12: Initial viewing tool screen containing list of courses

a. *Personalised list of courses*. This list of courses contains only those courses that the logged in user has access to, via belonging to an organisation that has a current software license for the software the course teaches. A user can select the course they wish to view by clicking on it, and then open it by pressing the 'Open' button at the bottom of the screen. The list of courses is scrollable. The colour used to display the title of the course depends on the logged in user's previous use of the course. If the user has already viewed the entire course, it is displayed in black. If they have viewed parts of the course but not others, it is displayed in blue. Finally, if the user has not viewed any part of the course, it is displayed in red.

b. *User login confirmation.* This is used to acknowledge the fact that a user has logged in under a specific user account, and to add personalisation to the user interface.

c. *Progress feedback legend*. This provides a visual key to the colours used in the display of course titles so that users do not have to recall the meanings of the different colours.

d. *Status bar*. This is used to keep users informed of the status of the system. It is most useful when communication with the server is required and consequently a delay can be experienced, such as when a course is being loaded. During this time, the status bar reads “Loading course…”.



Symphonia Mapper: Getting Started
- Background and Installation
  - Background
    - Introduction
    - What is Mapping?
  - Installation
    - Installation
    - Installation cont...
- Using the Symphonia Mapper
  - Launching the Mapper
  - The Symphonia Mapper
  - Message Definitions
  - Using the Mapper
  - Main Maps
  - Submaps
  - Mapping Repeating Items
  - External Functions
  - Other Basic Operations
  - Compiling Mappings

Introduction

This course introduces the concepts of mapping and shows how to use the Symphonia Mapper to perform some basic mapping tasks.

A basic knowledge of EDI and XML is assumed. If you are at all unsure about these topics, we strongly recommend that you familiarize yourself with them prior to commencing this mapping. The Symphonia web pages have introductions to these topics. Go to http://www.symphonia3.com/ and follow the link to *Having difficulty with messaging*.

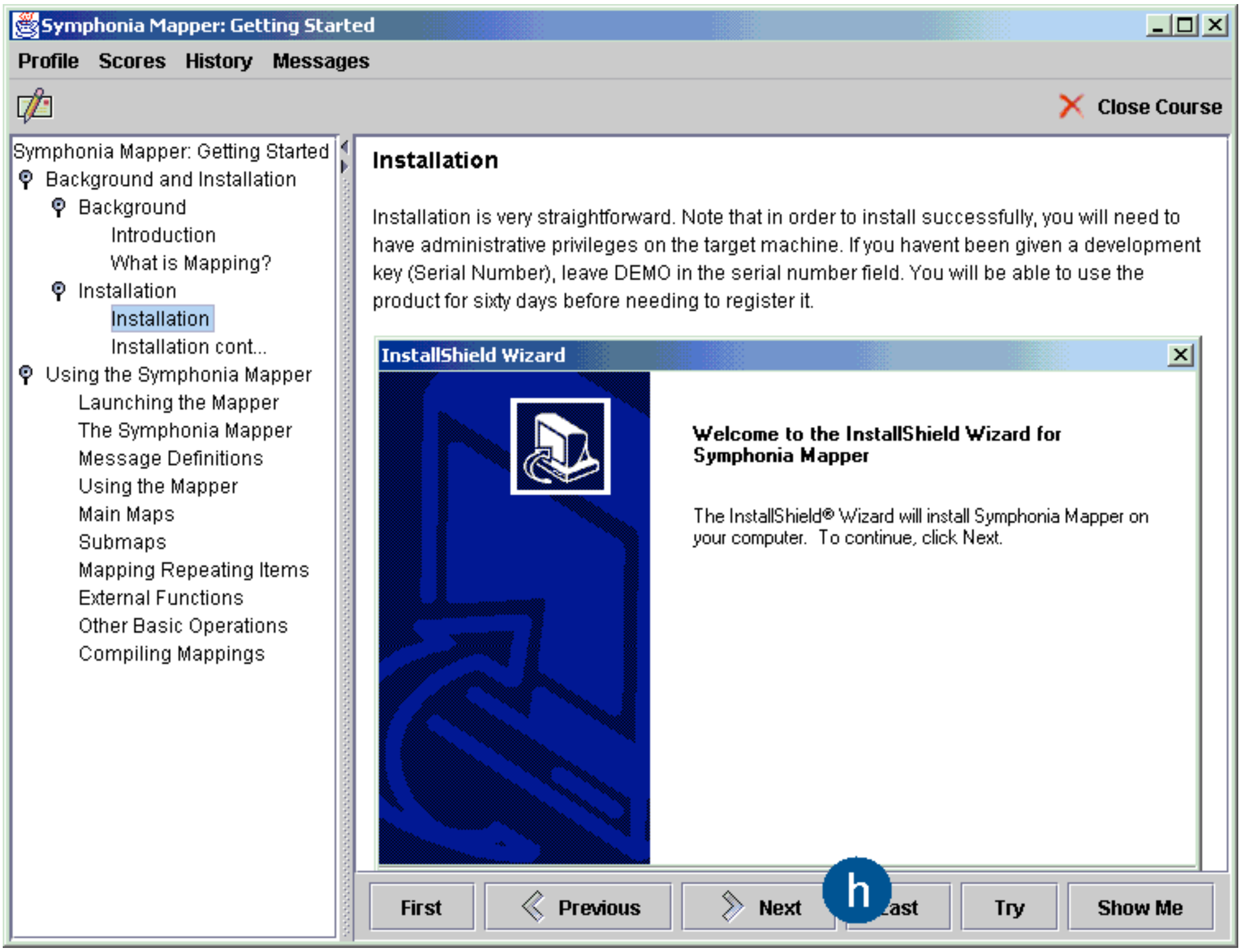
If you are unclear about the use of any of the technical terms in this section, please consult the glossary at the end of the full Symphonia Mapper manual.

Figure 5.13: Course viewing user interface

e. *Table of contents*. This displays the structure of the currently opened course. It shows the hierarchy of lessons, subjects and pages. Each page is an active link, which automatically displays the contents of the page in the right hand side of the interface, as described in f. The page currently being viewed is always selected in the table of contents to provide a constant source of orientation for the user.

f. *Page display area*. This section of the user interface displays the contents of the currently selected page. It is scrollable so that the entire contents of the page may still be viewed when the viewer is resized.

g. *Navigation bar*. This set of navigational buttons includes ‘first’, ‘previous’, ‘next’ and ‘last’. It allows users to navigate a course in a linear fashion. Only the applicable buttons are enabled at any given time, to prevent users making navigational errors, such as moving to the last page in a course when they are already there.



Installation

Installation is very straightforward. Note that in order to install successfully, you will need to have administrative privileges on the target machine. If you havent been given a development key (Serial Number), leave DEMO in the serial number field. You will be able to use the product for sixty days before needing to register it.

Welcome to the InstallShield Wizard for Symphonia Mapper

The InstallShield® Wizard will install Symphonia Mapper on your computer. To continue, click Next.

Figure 5.14: Course viewing user interface including animation functions

h. *Navigation bar with animation functions*. When the viewer detects an animation on a page, the navigation bar is altered to include two more buttons: ‘Try’ and ‘Show Me’. These enable users to participate in an interactive version of the current task they are learning, as well as view a non-interactive demonstration of the task, respectively.

5.5 Summary

This chapter presented the ideal conceptual design of the software training authoring, viewing and monitoring system. This included a complete description of the conceptual object models for training courses and information related to their use. It then outlined the client-server architecture of the system, describing the main components and how they interact during typical communication. Finally, it presented the design of the user interfaces of the authoring and viewing tools, the components of the system with which users interact. An in-depth description of all of the user interface components, including what they are used for and how they work, is provided in Chapter 7, entitled “Case Study: Using the Tools in a Software Development Environment”.

Chapter 6 Prototype Implementation

6.1 Introduction

This chapter presents the implementation of the proof of concept prototype tools, namely Reciprocity Author and Viewer. Where the previous chapter discussed the ideal conceptual design of the tools, this chapter presents the concrete design implemented during the development of the prototype tools. Any deviation from the ideal conceptual design during implementation is due to time and resource constraints, and a focus on exploring the concept of the tools rather than commercial readiness. This chapter includes important and interesting design and implementation decisions made and the reasons for them. We also discuss the advantages and disadvantages of choices made.

6.2 Author Implementation

The main components implemented in the Author tool are shown in Figure 6.1. Although there are many Author GUI and data manager components implemented in the tool, these have been grouped for clarity.

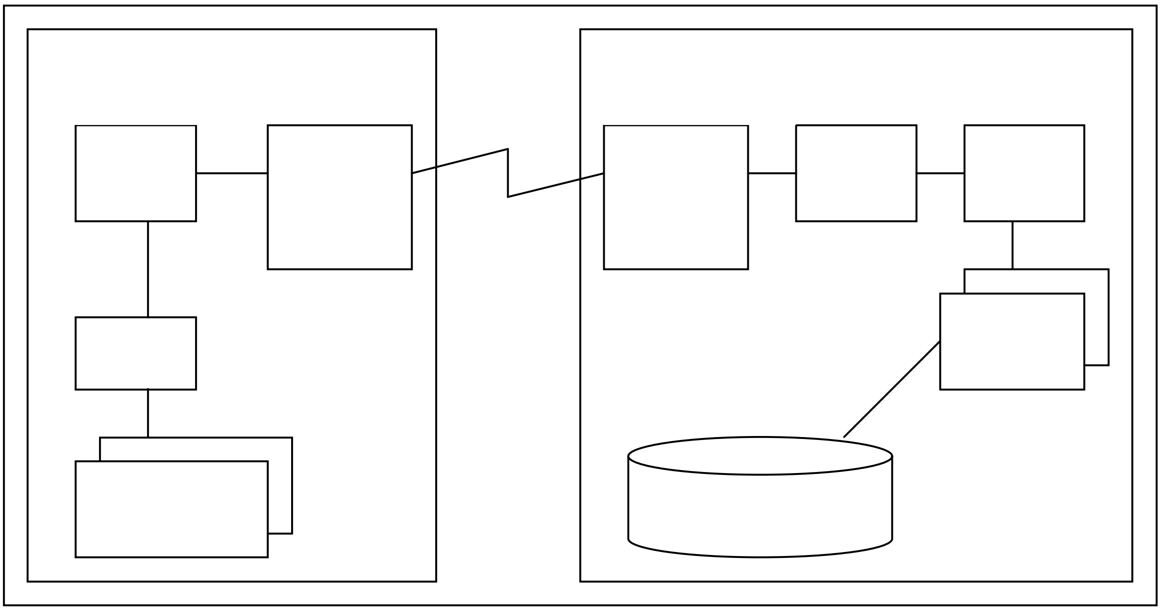


Figure 6.1: Prototype author tool infrastructure objects

The first stage of implementation required the development of the client-server infrastructure. On the server side, a master server communicates with a separate author server for each author currently logged in to the system. Each author server then has one instance of a remote author client, used to represent the author client on the server side and provide access to the author client functionality. The master server also controls all data access through communication with data managers, which directly access the design database. On the client side, a single instance an author client communicates with a single instance of a remote author server, the client-side representation of author server functionality. The author client also has a single instance of author, which controls all graphical interface components and the flow of control within the client-side application. The rest of this section details the implementation of the author client, author and author GUI components.

The implementation of the authoring tool consisted of the following components:

1. The implementation of a function to enable course structure design.

   This function must allow an author to build the structure of a course from the set of elements including lessons, subjects, pages and rating pages. To restrict and guide users in creating course structure, a hierarchical structure has been imposed that must be followed in order to create a complete and correct course.

2. The implementation of "core" electronic publishing media elements.

   Once the structure of a course has been created, the content of each page must be designed. For this, authors must be able to add “core” media elements to a page. This includes text, images and colour.

3. The implementation of an animation designer and viewing functionality to introduce user interactivity and allow demonstration.

   In addition to the core media elements presented above, the ability to demonstrate through animation was seen as very important in bringing interactivity to training courses. The authoring tool therefore had to allow users to construct animations that demonstrate a particular software situation. It also had to allow users to view their animations and try them out to ensure they are correct before being made available to end-users.

4. The implementation of functionality to enable set up and display of summarised statistical data gathered from course usage.

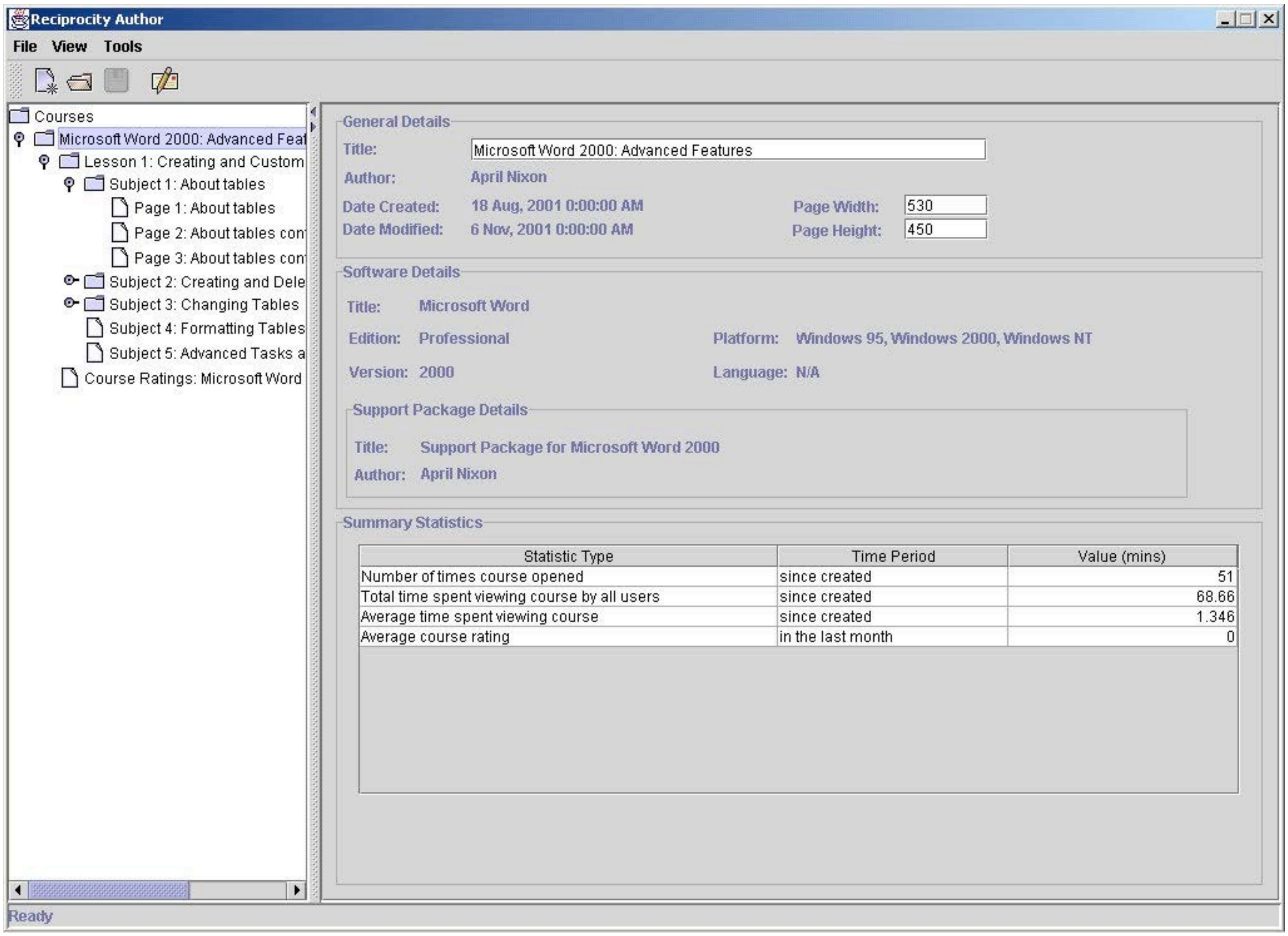
   During course usage, the progress of users is monitored and recorded in a log. Within the authoring tool, it was important to enable users to extract information from this data. The implementation therefore had to allow authors to choose what information they wanted to see and allow them to view this information at any time.

5. The implementation of an author – user direct messaging component.

This functionality had to allow for both broadcast and unicast communication between course authors and end-users.

6.2.1 Author User Interface Overview

The authoring tool user interface consists of a split screen, with structural information displayed on the left and details on the right. The structural information on the left shows the structural makeup of the currently loaded course, in the form of a tree. A tree representation was chosen because of the hierarchical nature of allowable course structures. The tree is always visible in the left hand side, however users can choose to hide it if they wish. The right hand side of the screen is used for two distinct activities. The first use is shown in Figure 6.2. In this figure, the author is viewing the properties of the currently loaded course. By selecting the course from the structural tree in the left hand side of the screen, its properties are automatically displayed in the right hand side.



Courses
- Microsoft Word 2000: Advanced Feat
  - Lesson 1: Creating and Custom
    - Subject 1: About tables
      - Page 1: About tables
      - Page 2: About tables con
      - Page 3: About tables con
    - Subject 2: Creating and Dele
    - Subject 3: Changing Tables
    - Subject 4: Formatting Tables
    - Subject 5: Advanced Tasks a
  - Course Ratings: Microsoft Word

General Details

Title: Microsoft Word 2000: Advanced Features

Author: April Nixon

Date Created: 18 Aug, 2001 0:00:00 AM

Page Width: 530

Date Modified: 6 Nov, 2001 0:00:00 AM

Page Height: 450

Software Details

Title: Microsoft Word

Edition: Professional

Platform: Windows 95, Windows 2000, Windows NT

Version: 2000

Language: N/A

Support Package Details

Title: Support Package for Microsoft Word 2000

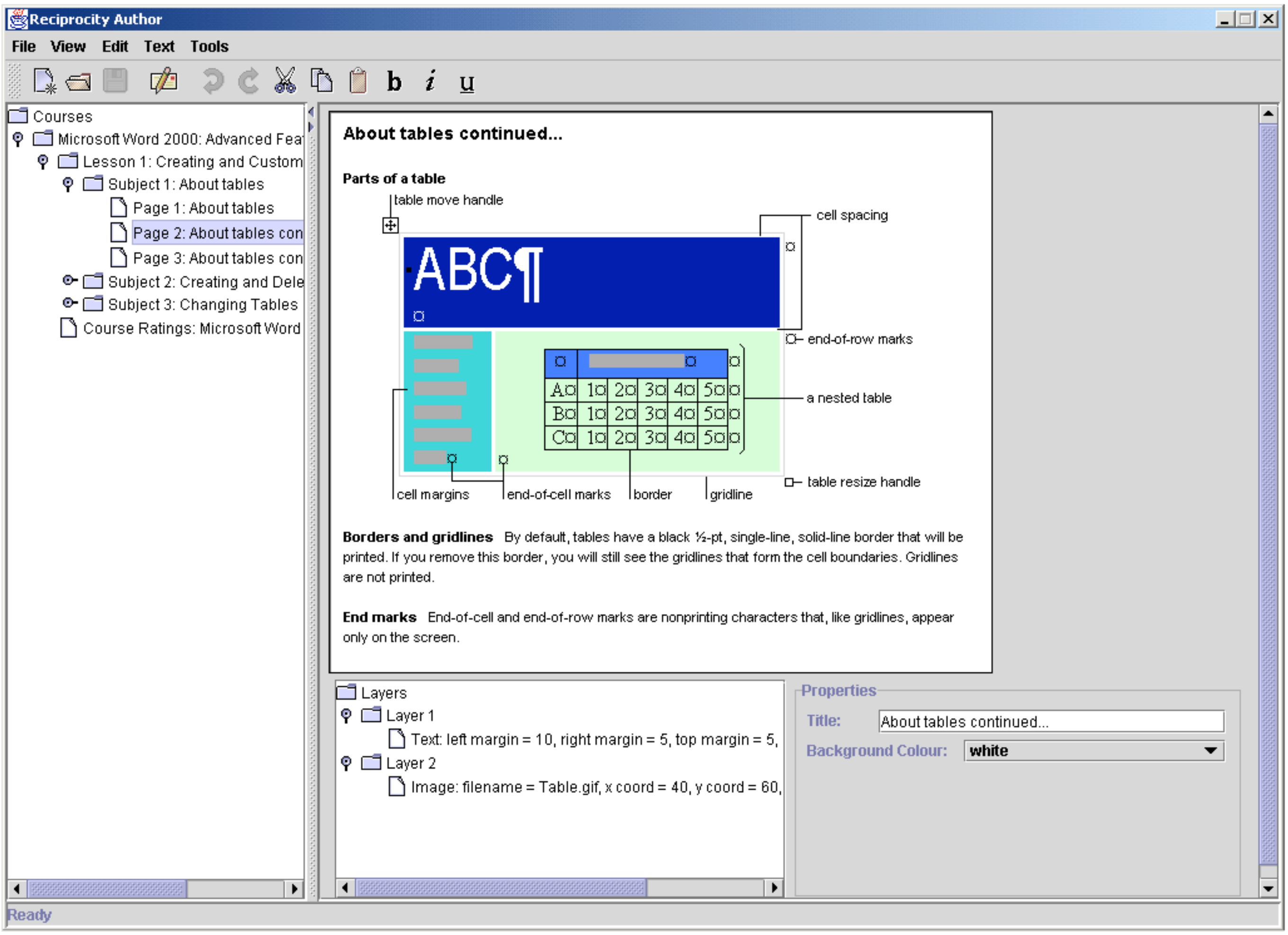
Author: April Nixon

Summary Statistics

Statistic Type	Time Period	Value (mins)
Number of times course opened	since created	51
Total time spent viewing course by all users	since created	68.66
Average time spent viewing course	since created	1.346
Average course rating	in the last month	0

Figure 6.2: Author tool user interface in use – properties display

The second use of the right hand side of the screen is shown in Figure 6.3. Here the author is viewing the content of a page selected from the structural tree. This view allows the author to design and edit page content.



About tables continued...

Parts of a table

Borders and gridlines By default, tables have a black ½-pt, single-line, solid-line border that will be printed. If you remove this border, you will still see the gridlines that form the cell boundaries. Gridlines are not printed.

End marks End-of-cell and end-of-row marks are nonprinting characters that, like gridlines, appear only on the screen.

Text: left margin = 10, right margin = 5, top margin = 5,

Image: filename = Table.gif, x coord = 40, y coord = 60,

Properties

Title: About tables continued...

Background Colour: white

Figure 6.3: Author tool user interface in use – page canvas display

The content of the right hand side of the screen then is determined by what type of course element is selected in the structural tree. If the element has a visual representation in the final course, this is displayed, along with the tools for working with it. Otherwise, the element's properties are displayed.

6.2.2 Course Structure Design

When designing a course using the Author tool, the structure of the course must follow the imposed hierarchical structure of elements. Figure 6.4 displays the subset of course elements that are of interest when building a course structure. Briefly, a course must have one or more lessons, each of which must have one or more subjects, each of which must have one or more pages. In addition to this, the course itself and any of the lessons may have a rating page. This structure is explained in more detail in the chapter named 'Specification'.

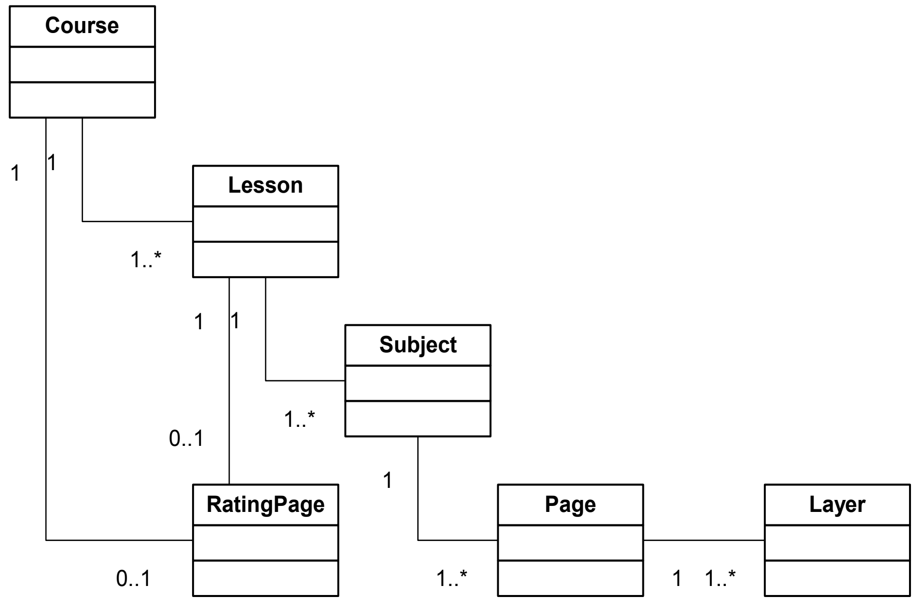


Figure 6.4: Course element structure

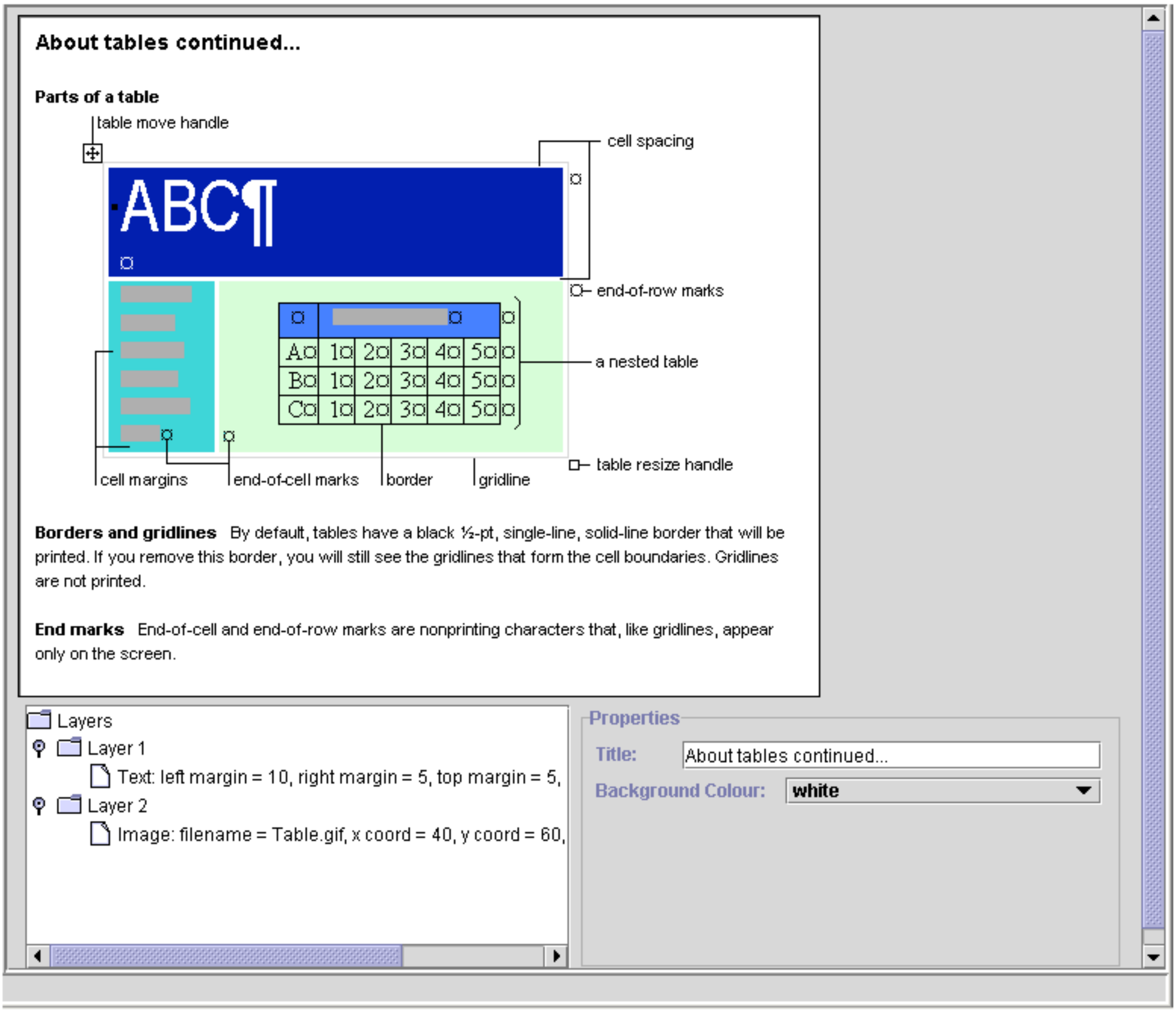
A user of the Author tool is able to create a new instance of a course, add any number of lessons to the course, subjects to the lesson and pages to the subjects. This also required the ability to delete any of these structural components. The abilities to move child components to a different parent and to reposition child components under the same parent were also necessary due to the significance of order in the progression of a course. For example, an author may want to move a page from one subject to another or to move the first page of a subject to the second page.

6.2.3 Core Media Elements

In order for the prototype author tool to allow the production of electronic training material adequate to evaluate the concept of the tool, several core media elements were essential in its implementation. These were text, images and colour.

Other simple forms of media considered included sound, to enable both narration and exact replicas of situations during software demonstrations, text and image based navigational links between two points within a course and links from a point in a course to an external web address. Although these forms of media would have added to the usability of the authoring tool and therefore the training material produced, they were not implemented partly because they would not be necessary to prove the concept of the tool and partly due to implementation time restrictions.

A user of the authoring tool is able to insert both text and images at any point on a page. In order to provide a usable text-editing tool, editing functions allow changes to font, font size, style and colour. Selection, deletion, undo and redo functions are also available. Existing images may be loaded and inserted onto a page. They may also be moved, resized and deleted.



About tables continued...

Parts of a table

¤	¤					¤
A¤	1¤	2¤	3¤	4¤	5¤	¤
B¤	1¤	2¤	3¤	4¤	5¤	¤
C¤	1¤	2¤	3¤	4¤	5¤	¤

Borders and gridlines By default, tables have a black ½-pt, single-line, solid-line border that will be printed. If you remove this border, you will still see the gridlines that form the cell boundaries. Gridlines are not printed.

End marks End-of-cell and end-of-row marks are nonprinting characters that, like gridlines, appear only on the screen.

Layers
- Layer 1
  - Text: left margin = 10, right margin = 5, top margin = 5,
- Layer 2
  - Image: filename = Table.gif, x coord = 40, y coord = 60,

Properties

Title: About tables continued...

Background Colour: white

Figure 6.5 Page design area

The page design area resides in the right hand side of the main screen, as shown in Figure 6.3. An enlarged version focusing on just the page design area is shown in Figure 6.5. This section of the user interface contains a page canvas, which shows the true visual representation of the currently selected page, above a tree representation of the page's layers. To the right of the layer tree is a properties sheet that displays the properties for the element currently selected in the layer tree. If no element is selected, the general properties of the page itself are displayed.

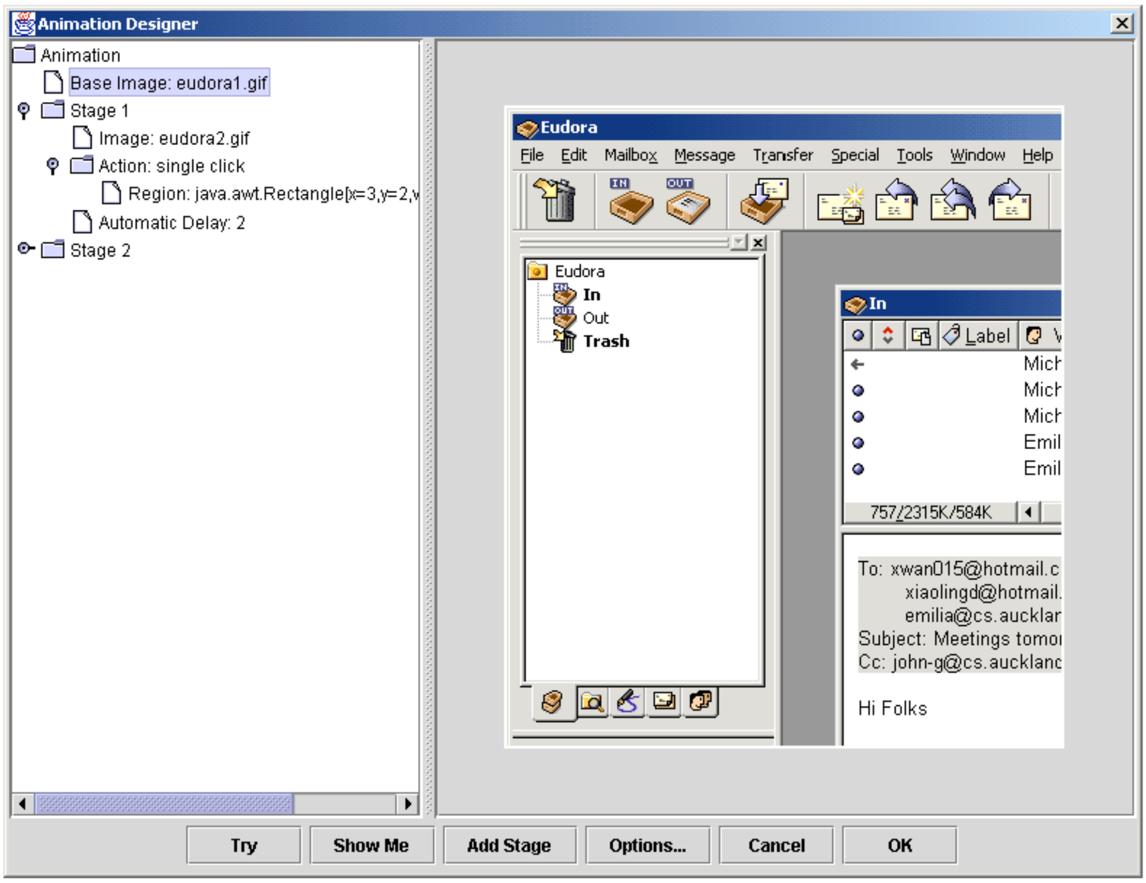
When a page node is selected in the course structure tree, the corresponding page design is displayed in the page design area. To enable authoring flexibility, the page area has been represented with three dimensions: width, height and depth. This allows users to layer media in any order they choose. To this end, users are able to add any number of layers to a page and text and images may then be added to these layers. The existing layers of the current page and their contents are listed below the page itself, in the form of a tree. By default, one layer is added to every new page and a textual element is added to that layer. This allows users to create a new page and immediately begin entering textual content.

Two methods of placing text on a page were originally considered. The first was a document editing metaphor, where the entire page is like a page in a word processor. The cursor would initially be placed at the top of the page and pressing 'enter' would move it down the page. The second method viewed the page more as a picture to which text boxes could be added. A text box could be placed at any point on the page and could be resized to fit the text within it. The word processor metaphor was chosen for prototypical implementation for two reasons. Firstly, it was felt that user familiarity with word processing software would contribute to increased usability and secondly, the implementation of a single, static sized textual component was much simpler than allowing multiple, dynamically sized textual components.

Text editing functions are available through the drop down menus in the menu bar at the top of the screen and through the toolbar. Users are able to make changes to the font, size, style and colour of all text. The options available are as follows: font: serif, sanserif; font size: 6 – 24; style: plain, bold, italic; colour: black, white, red, green, blue.

6.2.4 Animation Designer

In addition to the simple forms of media already discussed, we believed it was essential to include the ability to animate in order to create interactive software demonstrations. An animation is represented as a base image and then a series of one or more stages, which consist of a correct user action and a resulting image. For example, a base image may show a menu bar and the first stage could consist of the action of clicking once with the left mouse button over the File menu and a resulting image that showed the drop down File menu. The user actions implemented in the tool are single click with the left mouse button, single click with the right mouse button, double click with the left mouse button, a key press including modifiers and the typing of a string of characters. Drag and drop was considered as a user action and would be suitable in a complete implementation but was not implemented in the prototype due to time restrictions.



Animation
Base Image: eudora1.gif
Stage 1
Image: eudora2.gif
Action: single click
Region: java.awt.Rectangle[x=3,y=2,w
Automatic Delay: 2
Stage 2

Mich
Mich
Mich
Emil
Emil

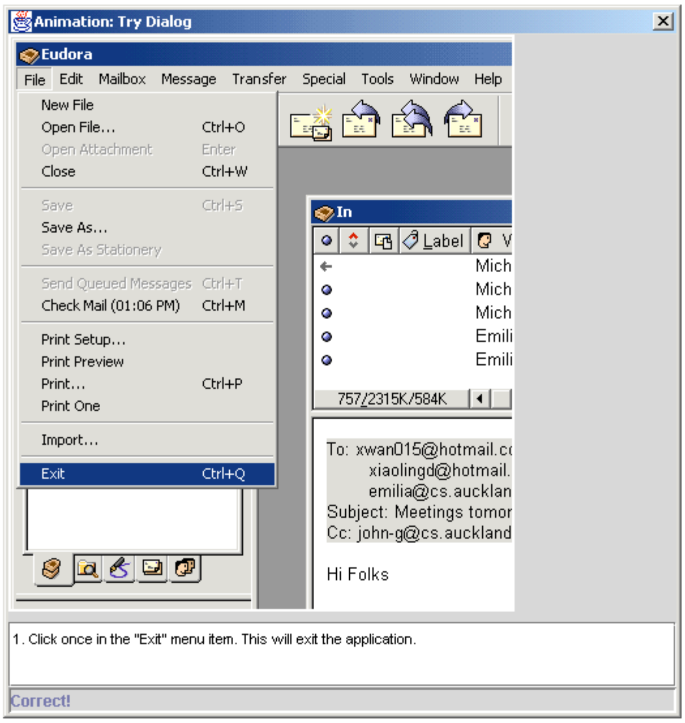
757/2315K/584K

To: xwan015@hotmail.c
xiaolingd@hotmail.
emilia@cs.aucklar
Subject: Meetings tomoi
Cc: john-g@cs.aucklanc

Hi Folks

Figure 6.6 Animation Designer

At any stage, a user may view the animation, either as a non-interactive demonstration by choosing the ‘show me’ function, or interactively by choosing the ‘try’ function. The try function opens a separate dialog window, which loads the base image of the animation, displays the instructions for the first stage and then waits for input from the user. If the user carries out the correct action, the user is informed that it was the correct action and the animation moves to the next stage. If the user makes an incorrect action, they are told this and are allowed to try again. In Figure 6.7 shows the try dialog waiting for input from the user. The correct action for the user to make in this example would be to click once on the left mouse button over the ‘exit’ menu option.

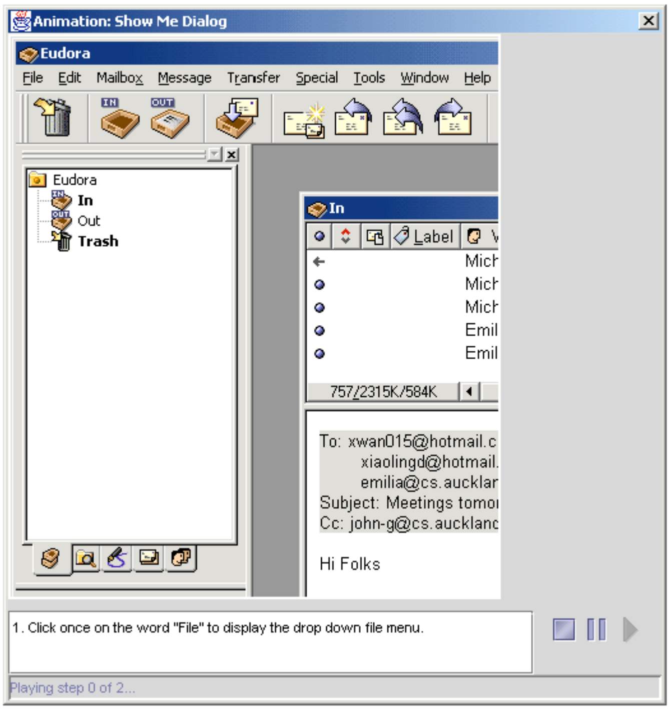


1. Click once in the "Exit" menu item. This will exit the application.

Correct!

Figure 6.7 Animation "Try" dialog

The 'show me' dialog, shown in Figure 6.8, is very similar to the 'try' dialog, only it contains three extra buttons: play, pause and stop. The reason for this is that the 'show me' function runs the animation continuously, from start to finish. The user may control the animation by pausing, restarting and stopping at any time.



1. Click once on the word "File" to display the drop down file menu.

Playing step 0 of 2...

Figure 6.8 Animation "Show Me" dialog

6.2.5 Summarised Statistical Data

Data is gathered both implicitly and explicitly from training course end users. While users traverse a course, their movements are recorded, including course component visitation and time spent. Explicit data is gathered by allowing course users to rate individual subjects, lessons and courses as well as to submit general comments about any of these parts of a course. The purpose of gathering this data is to benefit course authors by providing insight into both usage patterns and end user experience while using their courses. To gain these benefits, course authors need to be able to specify exactly what information they want to view and be able to view this information at any time. To achieve this with suitable flexibility, authors are able to choose any number of statistics from a predefined set that covers all of the data collected, as well as a time period over which they wish each statistic to be calculated.

The statistics provided include:

- The total number of times a course has been accessed
- The total number of distinct users who have accessed a course
- The total amount of time spent viewing a course
- The average amount of time spent viewing a course
- The total number of times a lesson has been accessed
- The total number of distinct users who have accessed a lesson
- The total amount of time spent viewing a lesson
- The average amount of time spent viewing a lesson
- The total number of times a subject has been accessed
- The total number of distinct users who have subject a lesson
- The total amount of time spent viewing a subject
- The average amount of time spent viewing a subject
- The average rating given to a course
- The average rating given to a lesson
- The average rating given to a subject
- The most common rating given to a course
- The most common rating given to a lesson
- The most common rating given to a subject

The following time periods have been implemented in the prototype tool:

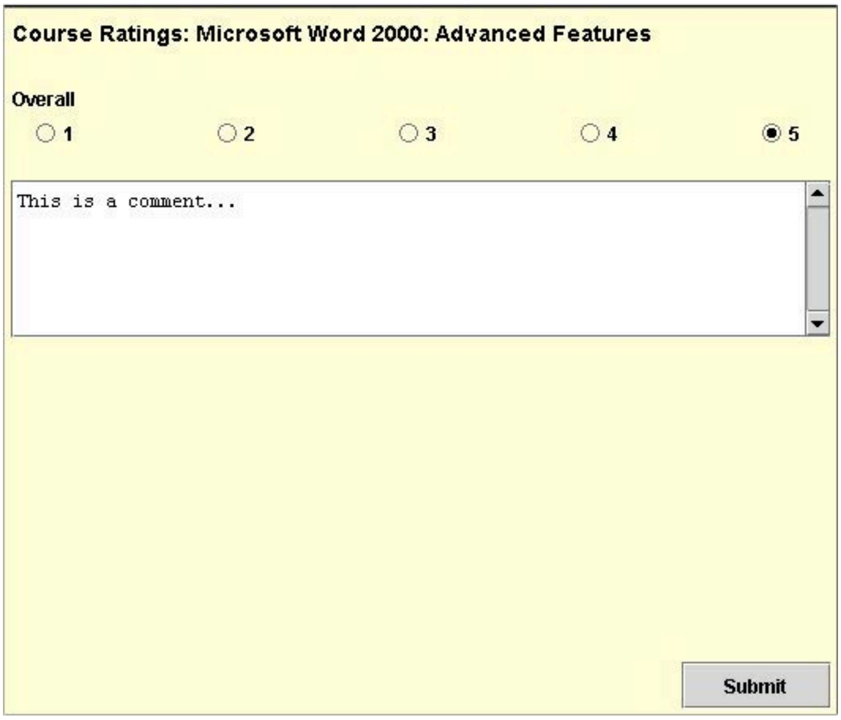
- Today
- Yesterday
- In the last week
- In the last two weeks

- In the last month
- In the last 2 months
- In the last 6 months
- In the last year
- Since created

Ideally, users should also be able to enter two dates that constitute the beginning and end of a desired time period. All statistics are summarised for complete end user anonymity.

In addition to usage statistics, authors are able to view the results of explicitly asking course end users for feedback. However, before this may be done, an author must first decide on the feedback they wish to receive and design a rating page. A rating page may be designed for the entire course and any of the lessons and subjects within it, but is completely optional. To completely specify a rating page, an author must enter one or more criteria on which they wish the course component to be rated and the possible rating choices users will be given for each criterion. Authors may provide either categorical choices, such as 'poor', 'good', 'very good', 'excellent', numerical choices, such as 1,2,3,4,5 or a combination of the two. For instance, the user may be presented with a scale of integers from 1 to 10 and the words 'poor' and 'excellent' corresponding to the numbers 1 and 10 respectively. The author may also choose whether or not to ask users for general written comments. There is no restriction on the number of choices for each criterion or the number of criteria that may be specified. The order in which the choices are specified is the order they will appear in the generated rating page.

Once all criteria are specified, a rating page is generated and added to the course structure tree. It may be viewed by selecting it from the structure tree. An example of such a rating page is shown in Figure 6.9. A page title is generated, along with sets of radio buttons, one for each criterion, which allow only one from each set to be selected. By default, the last choice is initially selected. If the author chose for comments to be requested, an editable text box will also appear at the bottom of the page.



Course Ratings: Microsoft Word 2000: Advanced Features

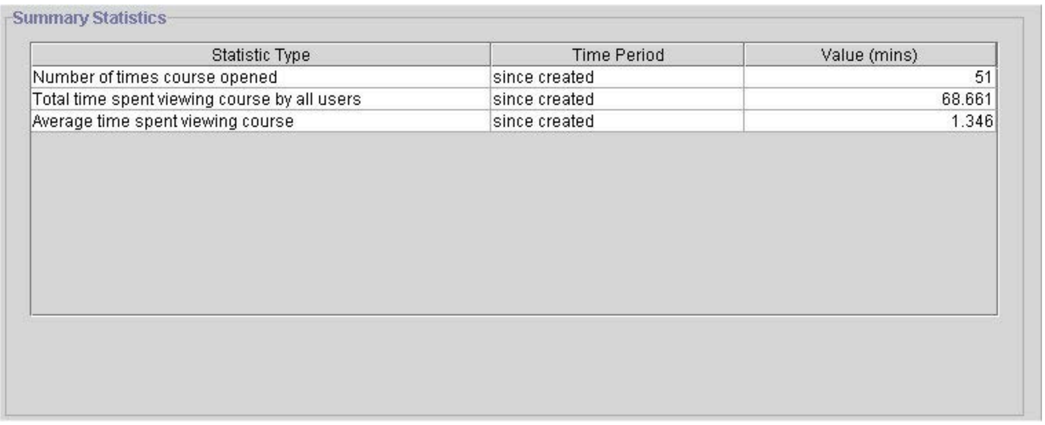
Overall

○ 1 ○ 2 ○ 3 ○ 4 ● 5

This is a comment...

Figure 6.9: Generated course rating page

An author may set up the reporting of any number of summary statistics for a particular course. The type of statistic can be selected from a predefined list as well as the time period over which to calculate the statistic. At any time, an author can view their set of statistics for a particular course by selecting the course node in the structure tree. All statistics currently specified are displayed along with the course properties on the right hand side of the main screen, as shown in Figure 6.10. These statistics are available at all times and are updated when a course is loaded into the authoring tool.



Statistic Type	Time Period	Value (mins)
Number of times course opened	since created	51
Total time spent viewing course by all users	since created	68.661
Average time spent viewing course	since created	1.346

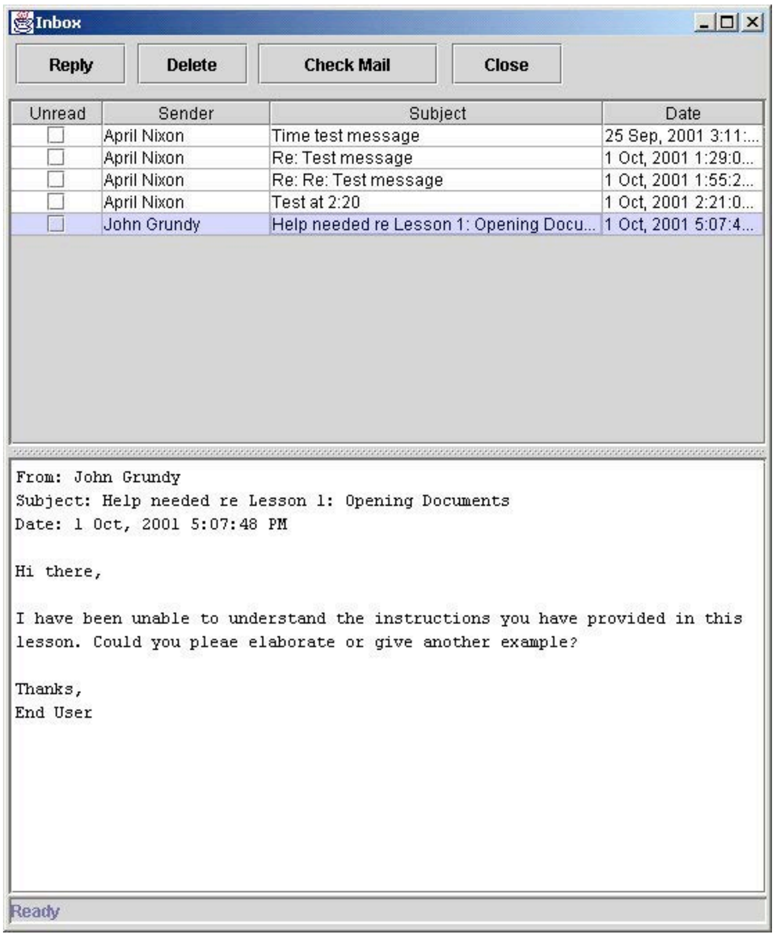
Figure 6.10: Summary statistics table

The example shown displays three summary statistics. The first statistic shows that this particular course has been opened 51 times since it was created. The second statistic listed shows that the total amount of time that end users have spent viewing the course since it was created is about 68 minutes. The last statistic listed shows that users have viewed the course for about 1.3 minutes on average, since it was created.

To enable complete flexibility, statistics chosen are individual to both the course they are specified for and the author who specified them.

6.2.6 Author – User Direct Messaging Component

The direct messaging component was intended to provide a direct communication channel between course authors and users. For authors, broadcast messages may to be sent to all users of a particular course to inform them of anything that needed their attention, such as the availability of a new course, or changes or additions to an existing course. For end users, the messaging channel allows them to send a unicast message to the author of a course, for example to ask for clarification on a point, to inform the author of a perceived error or even to suggest future course topics.



Unread	Sender	Subject	Date
□	April Nixon	Time test message	25 Sep, 2001 3:11:...
□	April Nixon	Re: Test message	1 Oct, 2001 1:29:0...
□	April Nixon	Re: Re: Test message	1 Oct, 2001 1:55:2...
□	April Nixon	Test at 2:20	1 Oct, 2001 2:21:0...
□	John Grundy	Help needed re Lesson 1: Opening Docu...	1 Oct, 2001 5:07:4...

```
From: John Grundy
Subject: Help needed re Lesson 1: Opening Documents
Date: 1 Oct, 2001 5:07:48 PM

Hi there,

I have been unable to understand the instructions you have provided in this
lesson. Could you pleae elaborate or give another example?

Thanks,
End User
```

Figure 6.11: Messaging inbox

The inbox is the start point for all messaging functions. From here, an author may compose a new broadcast message, view all read and unread messages and reply to any received messages. Upon opening the inbox, all existing messages, read and unread, are displayed in a list at the top of the inbox window. At this point, all new messages are automatically retrieved from the server and loaded beneath the existing messages. If there are no new messages the author is informed of this through the status bar at the bottom of the inbox window. To read any message listed in the inbox, the author would select the message by clicking on it. The message is then displayed in the message frame in the bottom half of the inbox window.

When deciding how to implement the direct messaging component, the main decision was between a completely internal messaging service and making use of external messaging through email. The advantage of using email would have been the availability of the messages. Email may be accessed from any number of commercially available inboxes. However, using email would have been a lot more difficult and time consuming to implement than a completely internal service, with the need to give individual email accounts to every course user, to subscribe to POP and IMAP protocols to communicate with an external mail server or even to implement our own mail server. A fully internal messaging service on the other hand, could make use of the existing client-server communication channel set up for course delivery and provide a completely secure messaging environment available only to course authors and subscribers. For these reasons, the direct messaging component was implemented as a completely internal service.

6.3 Viewer Implementation

The main components implemented in the Viewer tool are shown in Figure 6.18. Although there are many more viewer GUI and data manager components implemented in the tool, these have been grouped for clarity.

As in the previous section, this section focuses on the implementation of the viewer and viewer GUI components, as the hold more interest and relevance to this research than the fairly typical server and data access configuration.

The implementation of the course-viewing tool consisted of the following components:

1. The implementation of a personalised course selection menu.
2. The implementation of a course viewing mechanism, including navigation.
3. The implementation of end user progress reporting.
4. The implementation of end user usage monitoring.
5. The implementation of an author-user direct messaging component.

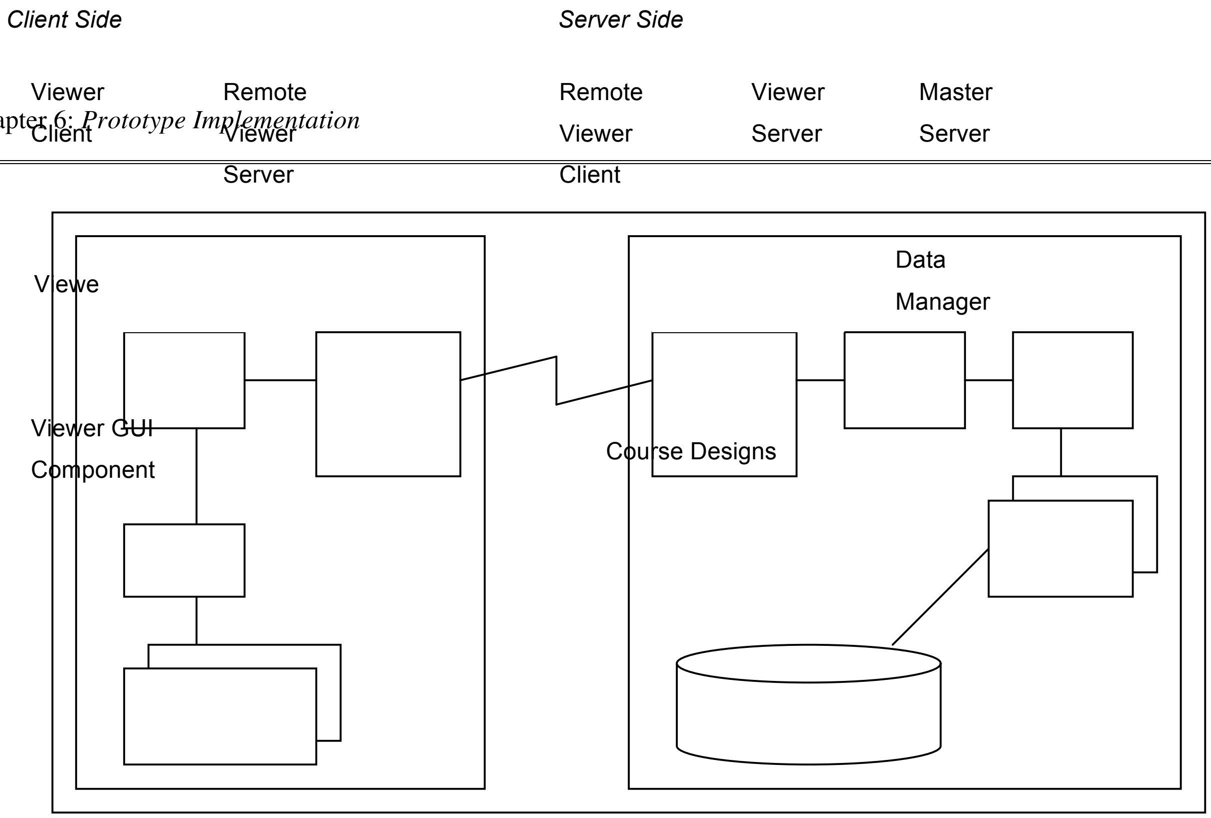
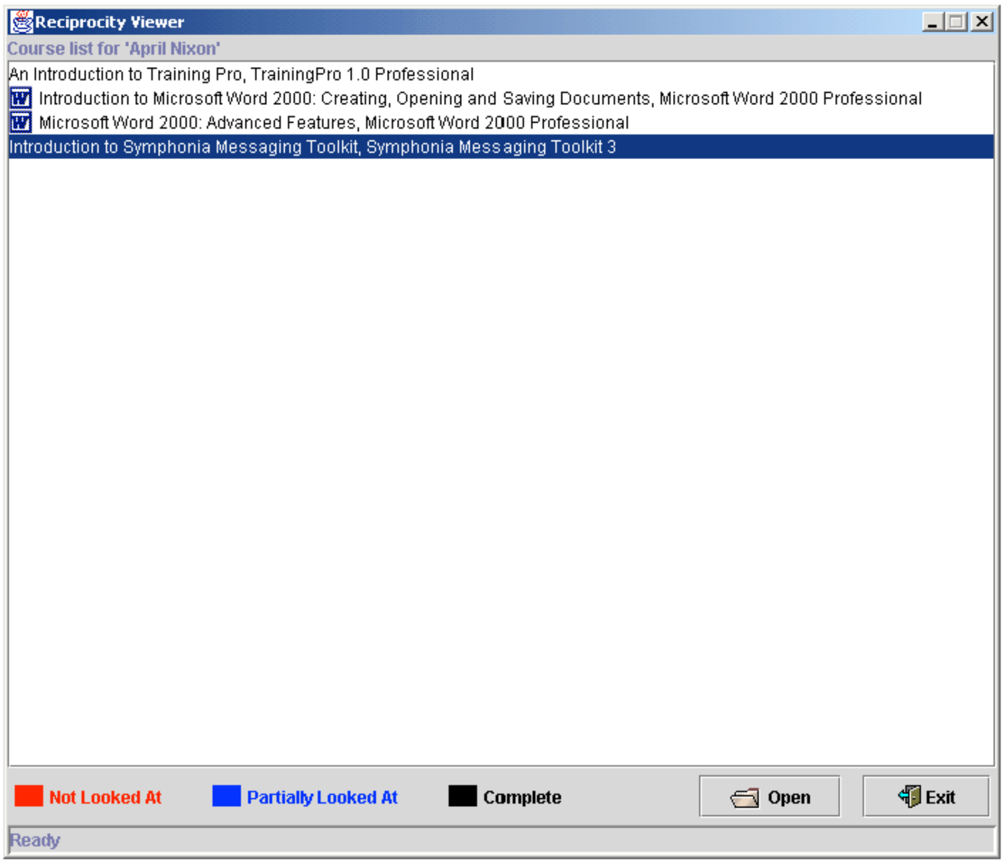


Figure 6.12: Prototype viewer tool infrastructure objects

6.3.1 Personalised Course Selection Menu

The implementation of an end user course membership hierarchy means that the database maintains a relationship between a user and those courses that are available to them at any given point in time. Availability of a course depends on the existence of a membership relationship between a specific user and a course and the premise that the current date is not past the expiry date of that membership. When a user logs into the viewing tool, they are initially presented with a personalised list of courses that are available to them at that point in time, as shown in Figure 6.13. Each item in the list corresponds to a single training course and includes the title of the course, the name of the software it instructs and a visual iconic representation of that software.

As an initial indication to the user of their progress with those courses currently available to them, the textual representation of each course in the list is coloured. The colour red indicates that the user has not yet viewed any part the course. The colour blue indicates that the user has viewed some parts of the course but not others. Lastly, the colour black indicates that the user has viewed the course in its entirety. A legend is included at the bottom of the course-viewing window to inform users of the meaning of the colours. Should a course be changed in any way, for example by the addition of a new lesson, these progress records would be updated. For example, a user who may have previously completed a course would see the course represented using blue to indicate partial completion should a new component be added to the course.



Reciprocity Viewer

Course list for 'April Nixon'

An Introduction to Training Pro, TrainingPro 1.0 Professional
Introduction to Microsoft Word 2000: Creating, Opening and Saving Documents, Microsoft Word 2000 Professional
Microsoft Word 2000: Advanced Features, Microsoft Word 2000 Professional
Introduction to Symphonia Messaging Toolkit, Symphonia Messaging Toolkit 3

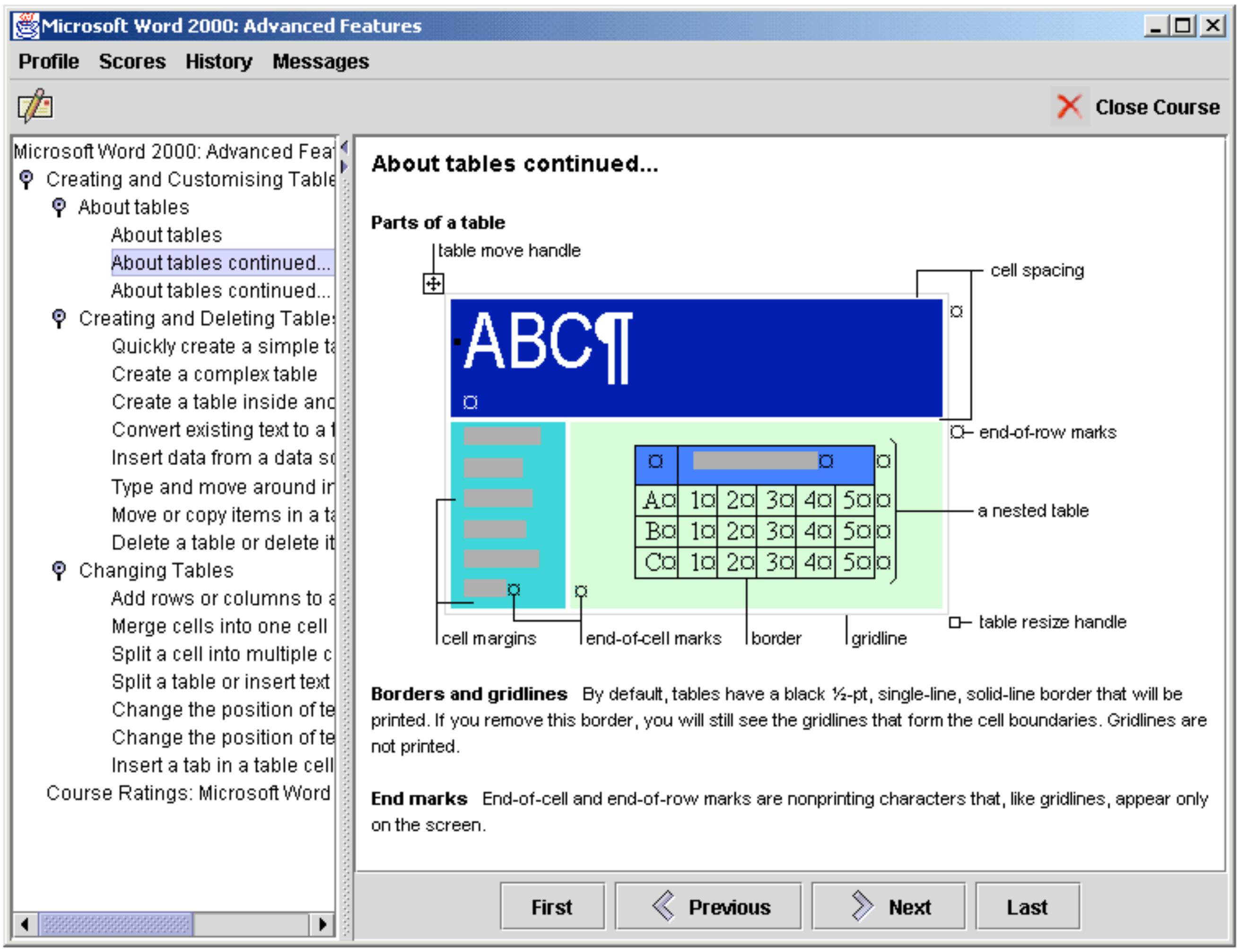
🟥 Not Looked At 🟦 Partially Looked At ■ Complete

Ready

Figure 6.13: Personalised course selection menu

6.3.2 Course Viewing Mechanism

A course-viewing window is loaded immediately after a course is received from the server in response to a user request. The aim of this window is to display both the summarised contents of a course so that a user can quickly ascertain what a course contains, and the actual training material. The layout of the course-viewing window is shown in Figure 6.14. The left hand side contains a scrollable table of contents, showing the structure of the course. Each element in the table of contents is an active link that displays the corresponding page in the right hand side. On the right hand side is a canvas that displays each page as it is selected for viewing. At any time, either side of the screen can be hidden to reveal only the desired side.



Microsoft Word 2000: Advanced Features
Profile
Scores
History
Messages
Close Course
Microsoft Word 2000: Advanced Fea
Creating and Customising Table
About tables
About tables
About tables continued...
About tables continued...
Creating and Deleting Table
Quickly create a simple t
Create a complex table
Create a table inside and
Convert existing text to a
Insert data from a data s
Type and move around ir
Move or copy items in a t
Delete a table or delete it
Changing Tables
Add rows or columns to
Merge cells into one cell
Split a cell into multiple c
Split a table or insert text
Change the position of te
Change the position of te
Insert a tab in a table cell
Course Ratings: Microsoft Word
table move handle
cell spacing
ABC¶
end-of-row marks
a nested table
cell margins
end-of-cell marks
border
gridline
table resize handle
First
Previous
Next
Last

About tables continued...

Parts of a table

A¤	1¤	2¤	3¤	4¤	5¤
B¤	1¤	2¤	3¤	4¤	5¤
C¤	1¤	2¤	3¤	4¤	5¤

Borders and gridlines By default, tables have a black ½-pt, single-line, solid-line border that will be printed. If you remove this border, you will still see the gridlines that form the cell boundaries. Gridlines are not printed.

End marks End-of-cell and end-of-row marks are nonprinting characters that, like gridlines, appear only on the screen.

Figure 6.14: Viewer tool user interface in use

Navigation

Ideally, we wanted users to be able to navigate a course in two ways. The first way would allow users to start at the beginning of the course and progress to the end in a linear fashion. This would be suitable for users who had no prior knowledge of the material being taught and who therefore may not know what to look for from a table of contents. The second form of navigation would allow users to search through a table of contents for a specific topic, and then be taken directly to that topic.

To enable linear navigation, we implemented the set of navigation buttons that can be seen at the bottom of Figure 6.14. The set of buttons consists of ‘first’, ‘previous’, ‘next’, and ‘last’. At any given point in time only the appropriate buttons are enabled. For instance, when a course is loaded, the first page of the course is automatically displayed in the course-viewing window and the buttons enabled are ‘next’ and ‘last’. These buttons allow users to navigate through an entire course, so are capable of passing through lessons and subjects.

To enable topic-specific navigation, a table of contents has been implemented, as discussed earlier. Every element in the table of contents is an active link. However, since only pages have actual content to be displayed, clicking on either the course itself, a lesson or a subject will display the first page in the course, lesson or subject respectively.

6.3.3 Usage Monitoring Mechanism

The aim of this mechanism is twofold. Firstly, it records user progress through a course and feeds this information back to the user through the use of colour. Secondly, it records every use of every course component including time spent and feeds this information back to course author(s). The second form of monitoring must be carried out without the knowledge of course users.

To achieve this, each time a course component is accessed, including the course itself, a new instance of a usage record is created and the time recorded. Once the component is no longer in use, the time is again recorded and the record is added to a list of complete usage records. For example, when a course is opened a usage record for the course is created. Then when a user accesses any lesson or subject within the course a new usage record is created for each of those. For this reason, it is often the case that more than one active usage record is maintained at any given point in time. Only when the course is closed is the original course record completed and added to the list. At the conclusion of a user's session, all usage records are sent in a batch to the server.

6.3.4 Author-User Direct Messaging Component

Since this component had already been implemented for the authoring tool, all that was required was to insert it into the viewing tool, with a few minor adjustments. Firstly, the ability to compose and send broadcast messages (see section 6.2.5) was removed. Secondly, the ability to compose messages that would be sent directly to the author of a course was added. The ability to reply to any message received was left untouched as this applies to both authors and viewers. The inbox interface used in the authoring tool version of this component also remained.

6.4 Implementation Choices

The tools described in this chapter are prototypes, built to prove the concept of a software training development environment that supports iterative improvement and two-way communication between authors and users. As such, the development of an acceptable system that would allow time for meaningful evaluation was more important than focusing on commercial priorities such as performance, completeness and usability. Due to constraints on time available for implementing the prototype system, implementation choices were greatly influenced by a necessity to work within these constraints rather than to satisfy the ideal requirements of the tools. Therefore choices regarding implementation language, platforms and communications protocols were based on both suitability for the task and our familiarity with them.

6.4.1 Programming Language

When deciding on a programming language two main factors were taken into consideration. The first was possible graphical user interface components. We wanted both the authoring and viewing tool user interfaces to be as expressive and flexible as possible for designing and rendering a range of electronic media. The second consideration was the time available for implementation. Due to our previous experience developing with Java and the suitability of the Swing Toolkit for developing a flexible and expressive user interface, the decision was made to implement both tools using Sun Corporation's Java 2 SDK, Standard Edition, v1.3 [Java 2 SDK], [Java 2 API]. The main disadvantage of this choice is performance of the final system. Java 2 requires a java virtual machine to run all java applications and applications programmed in Java 2 also consume a large amount of memory. However, we did not experience any significant performance problems when running the prototype system on the hardware available - a 600 MHz Pentium III with 512 MB of memory running Windows NT version 4.0.

6.4.2 Communications Protocol

The choice of communications protocol when implementing a client server system is a very important. The protocol chosen must often deal with differences in application language and platform, while adding a minimum of overhead to avoid downgrading the performance of the distributed system. Due to the decision already made to implement all components of the prototype system using Java 2, language incompatibility between the system components was not an issue. The major components of the prototype system, the server and author and viewer tools, were not required to communicate with any other applications, so no issues with application standards or platform compatibility arose. Performance, while not a major consideration in the implementation of the prototype, would contribute to the usability of the tools, which is a factor in later evaluation.

Due to these relatively loose constraints and our familiarity with Java and Internet technology, the choice was narrowed to Java's Remote Method Invocation (RMI) and TCP/IP over sockets, and in the end TCP/IP over sockets was chosen for implementing the prototype system's distributed communications. The main reason for this was the lower performance overhead of sockets compared to RMI, but sockets provided the added advantage of complete flexibility of data transferred. Any file format could be serialized and sent down a socket, including any type of media that may be required as part of a training course. The only requirement would be an active TCP/IP connection on both the host of the server and the host of any client who wished to connect. Other forms of distributed communication considered were Microsoft's DCOM and Object Management Group's CORBA. The main reason these were not selected was our unfamiliarity with them and the impact this would have had on implementation time.

6.4.3 Data Persistency

An important feature of the author tool us the ability to save course designs to a store of courses available for use. This enables authors to make changes to courses and have these immediately available for end users. When considering persistency options, we first took into account the potential complexity and size of a complete course structure. The course structure data model consists of more than 40 objects when the membership structure is included. Also taken into consideration was the use of a client-server implementation paradigm. This lends itself well to centralised data storage and would allow tight control over data access, while still allowing some client side caching for improved performance. For these reasons, we decided to use relational database technology for persistent storage of course designs. Due to its negligible cost and ready availability, Microsoft Access 2000 was used to implement the persistent data store. Since Java 2 was used to implement both the prototype server and the two prototype tools, JDBC was used for data access between the server and the data store. The other main persistence option considered was writing designs to a text file. The structure of a course could quite easily have been turned into a hierarchical structure using XML-like tags that would then be parsed when reading the file to re-create the original object structure. While this would have been acceptable for simply persisting designs, it became much more complex when both author and end user membership rights, usage records and the relationships between these had to be recorded. For this reason, file writing was not chosen as the method of persistence.

In the prototype implementation concurrent authoring is not permitted. Although many authors may make changes to a single course, they are not able to do so simultaneously. Concurrent authoring would increase usability of the authoring tool, allowing simultaneous and co-ordinated work for multiple authors. The implementation of concurrent authoring would be recommended for a full-scale or commercial version of the authoring tool, however it was not necessary to prove the concept of the tool and therefore not implemented in the prototype.

6.5 Summary

The purpose of this chapter is to describe the issues we came up against and the choices we made to deal with them during the implementation of the prototype system. It is also to suggest how the prototype could have been better developed, if it were implemented outside the scope of this thesis.

This chapter discussed in detail the implementation of the prototype course authoring and viewing tools. This included descriptions of the main functions implemented in both tools, some of the decision decisions made and the reasons for them. We then briefly presented the main implementation choices made, including programming language, development platform and hardware, communications protocol and persistency mechanism, and the advantages and disadvantages of the choices made.

Chapter 7 Case Study: Using the Tools in a Software Development Organisation

7.1 Introduction

This chapter presents an in-depth description of the prototype tools in use. First, a complete training course for Orion Systems Ltd's Symphonia Mapper software product is specified including its structure and content. Then the complete process of authoring the course using the authoring tool prototype is described, and finally the process of viewing the course is described. The purpose of this chapter is to provide an insight into the workings and capabilities of the tools.

7.2 Course Specification

Since training material already exists for Symphonia Mapper in both paper-based manual form, [Brewerton 2001], [Symphonia User Guide 2001], [Symphonia Mapper 2001] and PDF format, [Symphonia Mapper Guide], most of the structure and content of the course described in this chapter has been extracted from this material. Some content, such as the animation components, has been constructed from scratch since it was not present in any existing training material.

7.2.1 Structure Outline

Course: Symphonia Mapper: Getting Started Guide

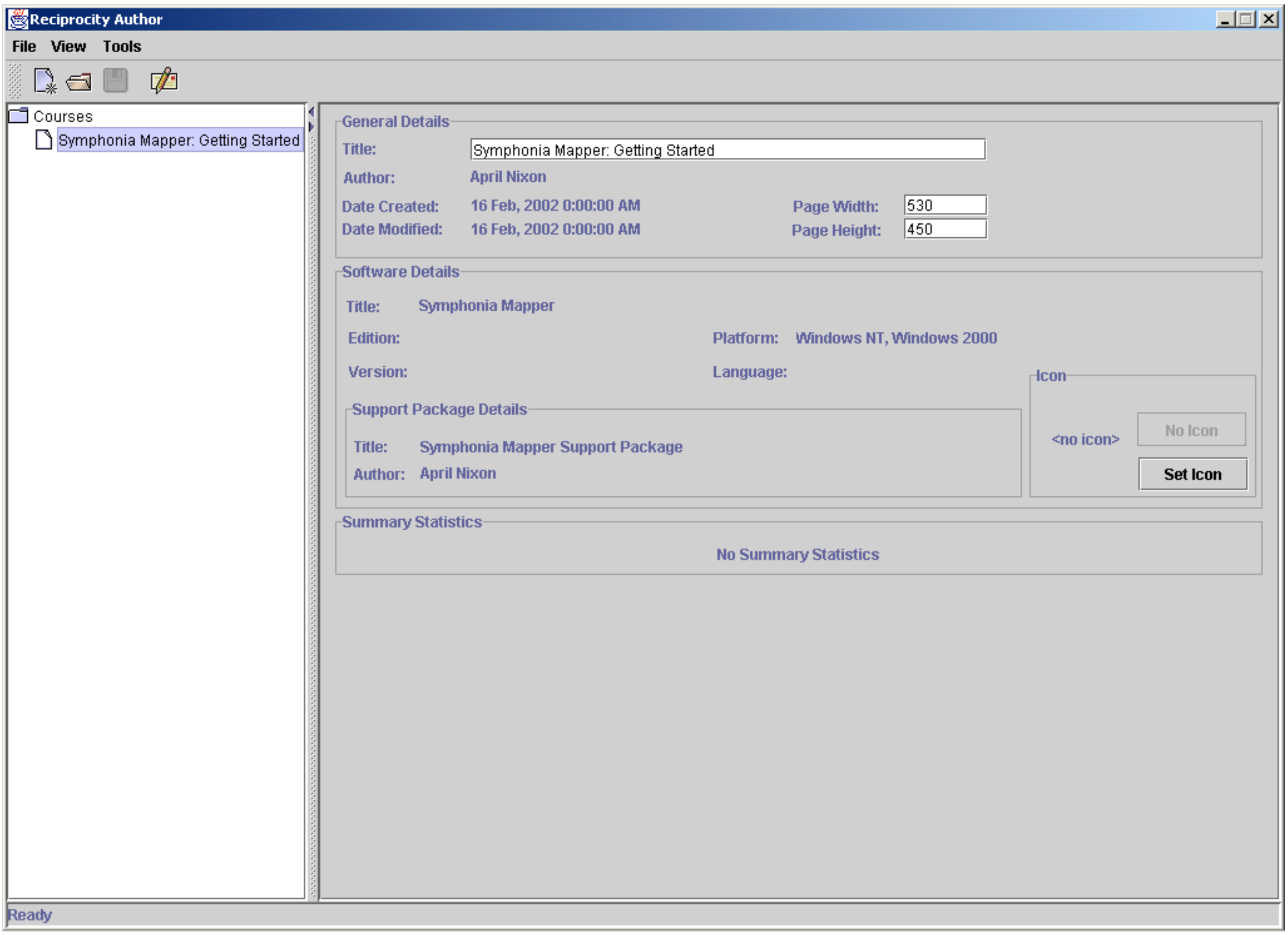
- Lesson 1: Background and Installation
    - Subject 1: Introduction
    - Subject 2: Installation
- Lesson 2: Using the Mapper
    - Subject 1: Launching the Mapper
    - Subject 2: The Symphonia Mapper
    - Subject 3: Message Definitions
    - Subject 4: Using the Mapper
    - Subject 5: Main Maps
    - Subject 6: Submaps
    - Subject 7: Mapping Repeating Items
    - Subject 8: External Functions

- Subject 9: Other Basic Operations
- Subject 10: Compiling Mappings

7.3 Authoring the Course

7.3.1 Creating a New Course

The first action I took after opening the author application was to register the software I wanted to create a course for, namely the Symphonia Mapper. I did this by selecting the ‘new’ function from the File menu and choosing the ‘new software’ option in the first step of the dialog presented. In the second step of the dialog, I entered the details of the software, including its title, version, edition, platform(s) and development language. All of this information excluding the title is optional, but is included to help users distinguish between courses for similar software so they may find the specific course they require for the software they are using. By pressing ‘OK’, this new software record was then available for the next step in the process, which was to create a support package.



General Details

Title: Symphonia Mapper: Getting Started

Author: April Nixon

Date Created: 16 Feb, 2002 0:00:00 AM
Page Width: 530

Date Modified: 16 Feb, 2002 0:00:00 AM
Page Height: 450

Software Details

Title: Symphonia Mapper

Edition:
Platform: Windows NT, Windows 2000

Version:
Language:

Support Package Details

Title: Symphonia Mapper Support Package

Author: April Nixon

Icon

<no icon>

Summary Statistics

No Summary Statistics

Figure 7.1: Displaying the general details of a course

This support package would hold all courses created for the Symphonia Mapper in the future. To do this, I again selected the ‘new’ function from the File menu, this time choosing the ‘new support package’ option. The second step in the dialog displayed a list of all currently registered software and from this list I selected ‘Symphonia Mapper’ and pressed ‘next’ to move on in the dialog. All that remained was to enter the title of the support package, entered as “Symphonia Mapper Support Package” and press ‘finish’. At this point, I was now able to create the new course by selecting ‘new course’ from the ‘new’ function dialog. I then selected the previously create Symphonia Mapper Support Package from the list presented and gave the course the title “Symphonia Mapper: Getting Started”. Pressing ‘finish’ completed the creation of the course and opened it ready for editing. A root node for the course could now be seen in the structure tree at the left of the user interface. Selecting this node would display the general details of the course on the right of the screen, as shown in Figure 7.1. At this stage I chose to save the structure I had just created by choosing the ‘save’ function from the File menu. Alternatively, I could have pressed the ‘save’ button on the toolbar, situated just below the menu bar.

It is possible to set the graphical icon to be displayed next to the title of this software when it is included in learners’ personalised menu of courses. To do this, I first displayed the properties of the course by selecting it from the structure tree. In the software details section of the properties displayed is a frame entitled ‘Icon’, which as shown in Figure 7.1 currently contains no icon and has two functions, ‘no icon’ and ‘set icon’. I pressed the ‘set icon’ button and was presented with an image selection dialog, as shown in Figure 7.2.

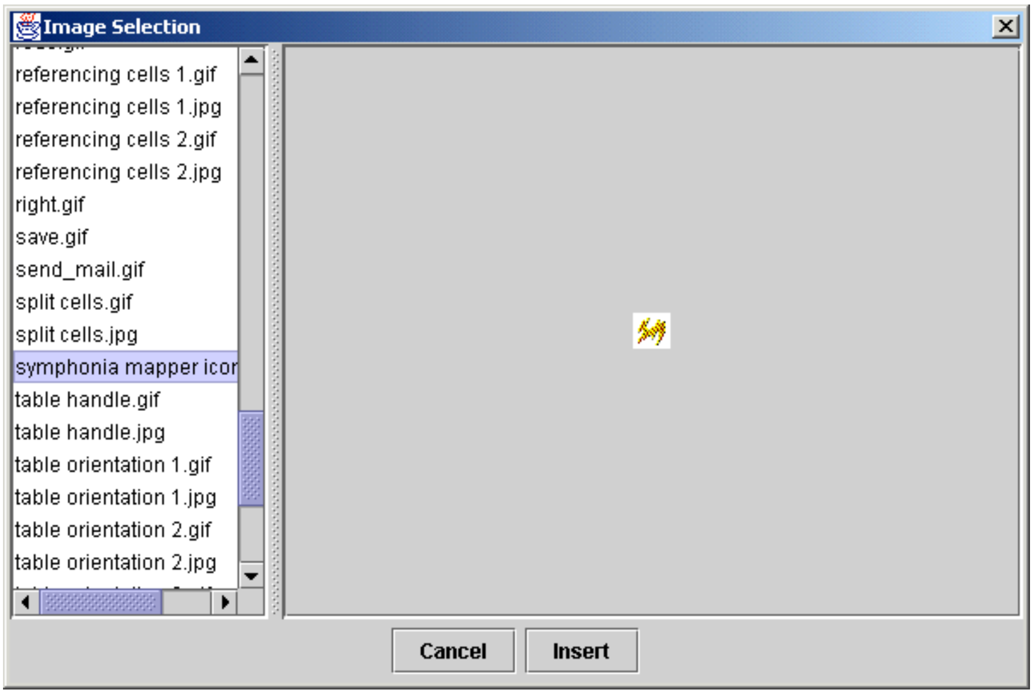
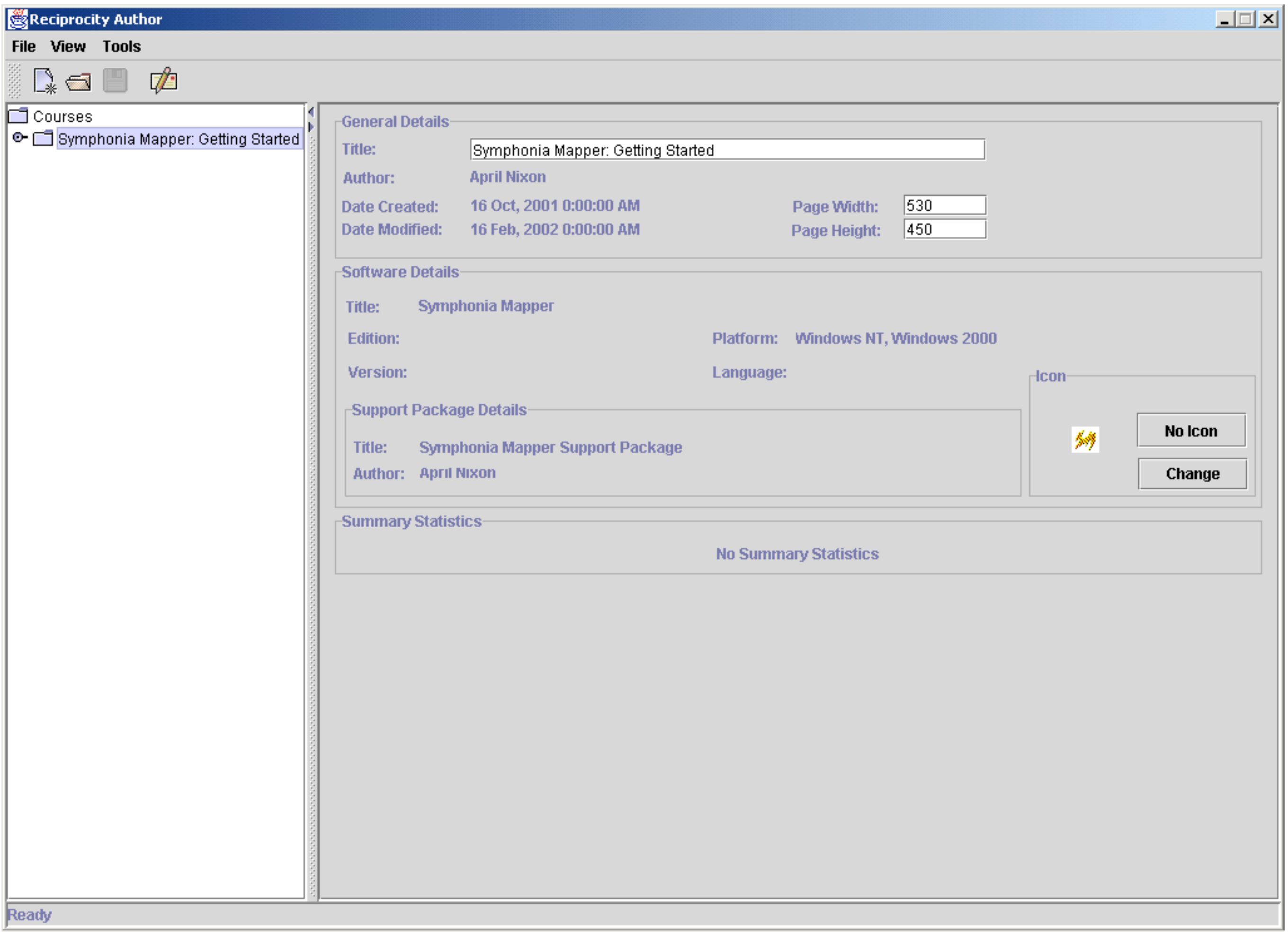


Figure 7.2: Selecting an image from the image selection dialog

This dialog displays the list of all images currently in the central design store in alphabetical order on the left, and each individual image as it is selected from the list on the right. Once I found the image I wanted as the course's icon, I pressed 'insert', which closed the dialog and placed the icon where previously there had been none. This result of this is shown in Figure 7.3. At this stage, if I wanted to change the icon I would press 'set icon' and select a different image from the image selection dialog. If I wanted to remove the icon entirely, I would press 'no icon'.



General Details

Title: Symphonia Mapper: Getting Started

Author: April Nixon

Date Created: 16 Oct, 2001 0:00:00 AM

Date Modified: 16 Feb, 2002 0:00:00 AM

Page Width: 530

Page Height: 450

Software Details

Title: Symphonia Mapper

Edition:

Platform: Windows NT, Windows 2000

Version:

Language:

Icon

Support Package Details

Title: Symphonia Mapper Support Package

Author: April Nixon

Summary Statistics

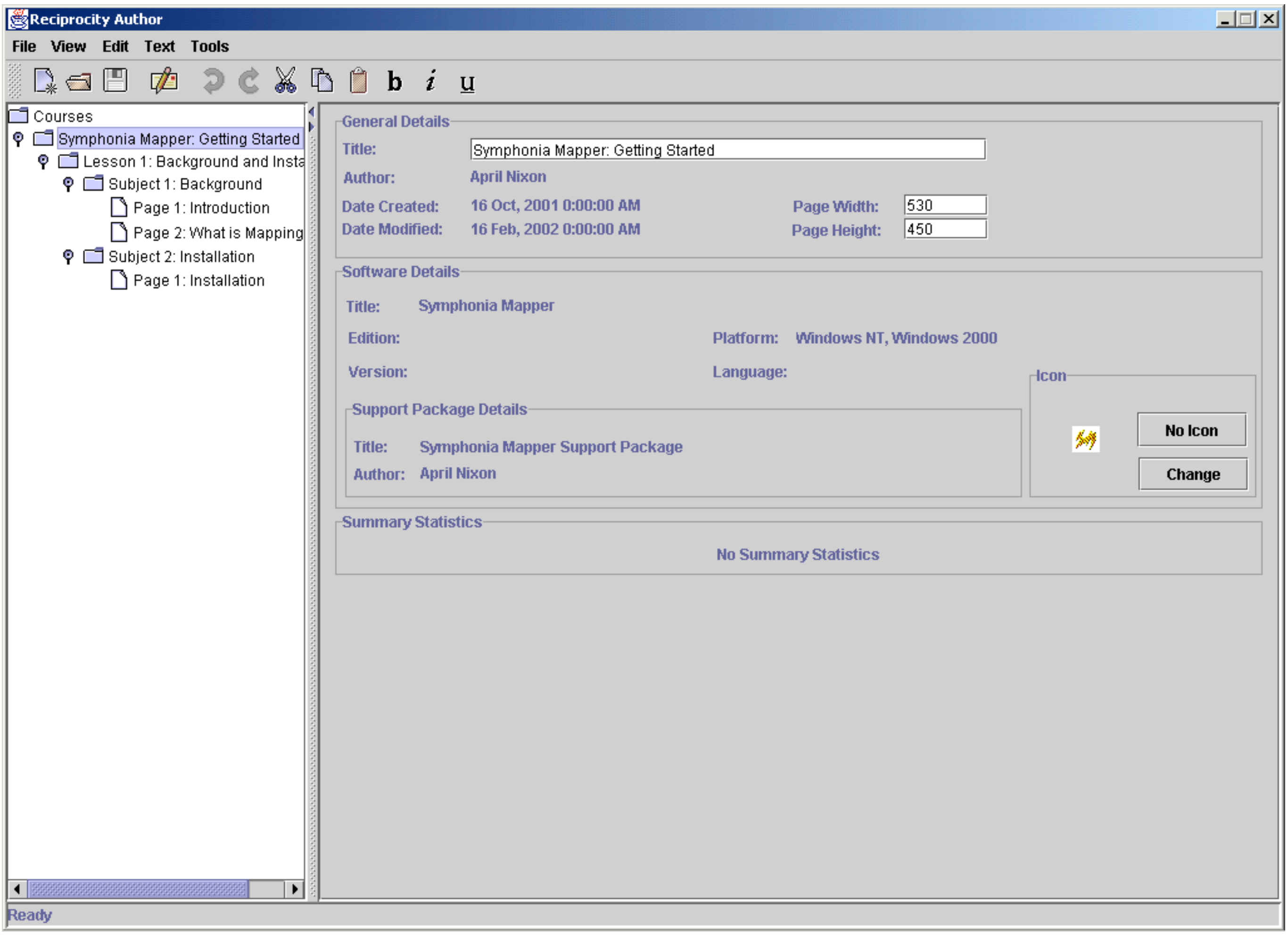
No Summary Statistics

Figure 7.3: Setting the software icon

7.3.2 Building the Structure

The next task was to create the lessons and subjects that would constitute the structure of the course as described in the previous section. To add the first lesson, I clicked with the right mouse button on the course node in the structure tree and selected 'new lesson' from the popup menu displayed. In the dialog that appeared prompting for a title for the lesson, I typed 'Introduction' and pressed 'OK'. A new node representing the lesson appeared in the beneath the course node in the structure tree. At this stage, I decided to change the name of the lesson to

‘Background and Installation’. To do this I selected the lesson’s node from the structure tree to display its properties in the right hand side of the user interface, just as the course’s properties were displayed in Figure 7.1. In the input component provided for the title of the lesson, I selected the current title and typed the new title, pressing ‘enter’ to complete the change. This update was immediately reflected in the course structure tree.



Courses
- Symphonia Mapper: Getting Started
  - Lesson 1: Background and Insta
    - Subject 1: Background
      - Page 1: Introduction
      - Page 2: What is Mapping
    - Subject 2: Installation
      - Page 1: Installation

General Details

Title: Symphonia Mapper: Getting Started

Author: April Nixon

Date Created: 16 Oct, 2001 0:00:00 AM
Page Width: 530

Date Modified: 16 Feb, 2002 0:00:00 AM
Page Height: 450

Software Details

Title: Symphonia Mapper

Edition:
Platform: Windows NT, Windows 2000

Version:
Language:

Icon

Support Package Details

Title: Symphonia Mapper Support Package

Author: April Nixon

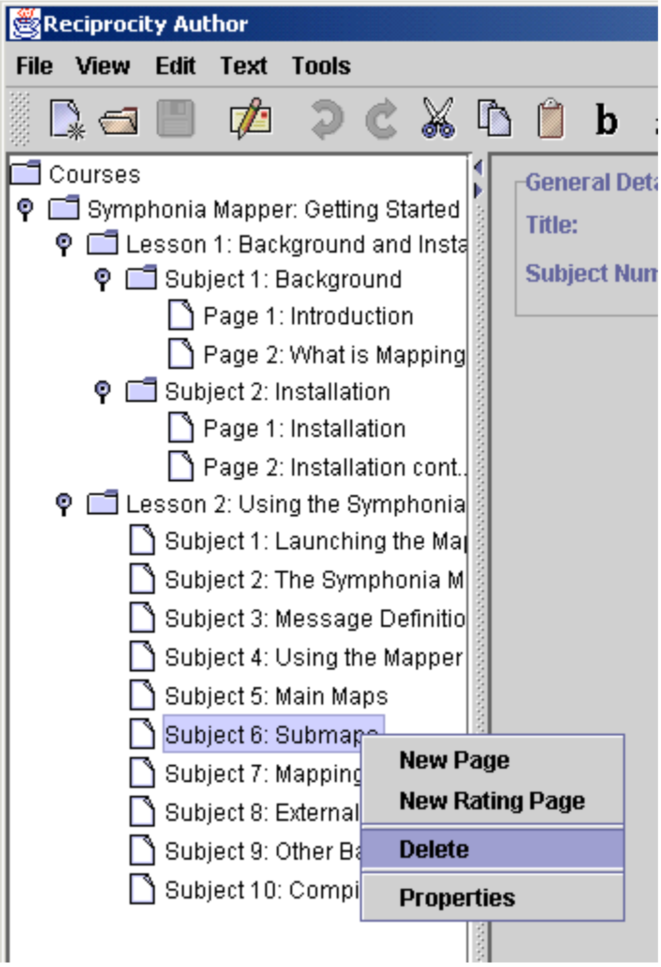
Summary Statistics

No Summary Statistics

Figure 7.4: Course structure after adding the first lesson

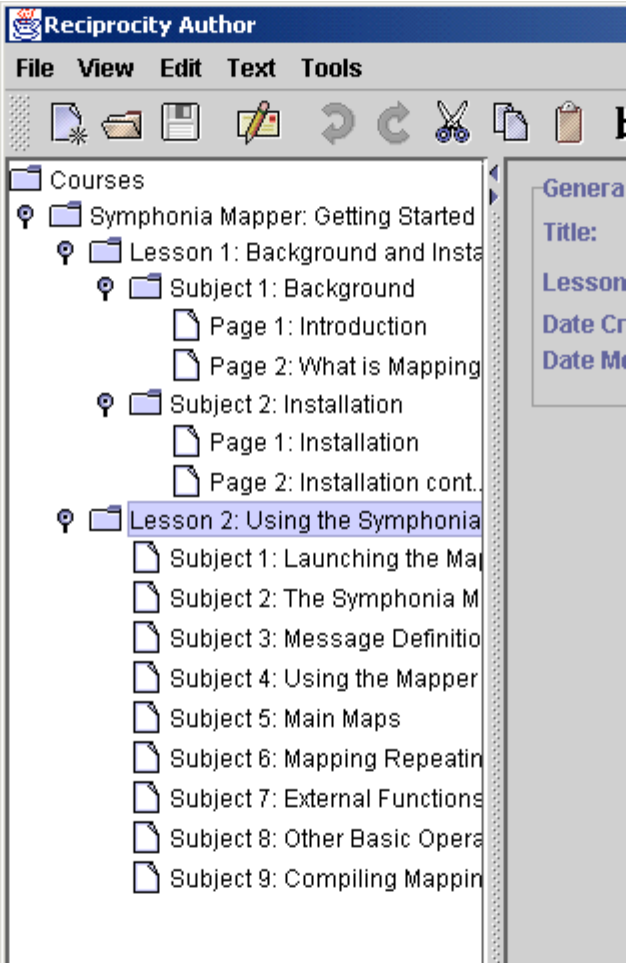
The next step was to add the required subjects to this lesson. The two subjects for this lesson were to be ‘Introduction’ and ‘Installation’. I began by first right clicking on the lesson node and selecting ‘new subject’ from the popup menu that appeared. I entered the title of the first subject as ‘Introduction’ and pressed ‘OK’, noticing that a node for this subject had been added under the lesson in the structure tree. I repeated this process for the second subject. From the existing training material from which I would extract the content for this course, I decided that at least two pages would be required for the introduction subject, the first containing introductory information and the second giving a definition of what mapping is in the context of this software. To add these pages I right clicked on the ‘background’ subject node and selected ‘new page’ from the popup menu, in much the same way as creating a new lesson or subject. I entered the title of the page as ‘Introduction’ and pressed ‘OK’. I repeated this process to create the second page of this subject and the single page for the ‘installation’ subject. The structure of the course at this point is shown in Figure 7.4. I then proceeded to add the rest of the lessons and subjects as outlined in section 7.2.1.

According to the previously outlined structure of the course, the second lesson, entitled “Using the Mapper”, contained ten subjects. If I wanted to delete one of these subjects, for example removing the subject entitled “Submaps”, I would right click on the corresponding node in the structure tree to display its popup menu, as shown in Figure 7.5, and from this choose ‘delete’. First I would be prompted for confirmation that I did indeed want to delete the subject, and after pressing ‘OK’ the subject would be deleted. The result of this is shown in Figure 7.6.



- Courses
  - Symphonia Mapper: Getting Started
    - Lesson 1: Background and Insta
      - Subject 1: Background
        - Page 1: Introduction
        - Page 2: What is Mapping
      - Subject 2: Installation
        - Page 1: Installation
        - Page 2: Installation cont.
    - Lesson 2: Using the Symphonia
      - Subject 1: Launching the Ma
      - Subject 2: The Symphonia M
      - Subject 3: Message Definitio
      - Subject 4: Using the Mapper
      - Subject 5: Main Maps
      - Subject 6: Submaps
      - Subject 7: Mapping
      - Subject 8: External
      - Subject 9: Other B
      - Subject 10: Compi

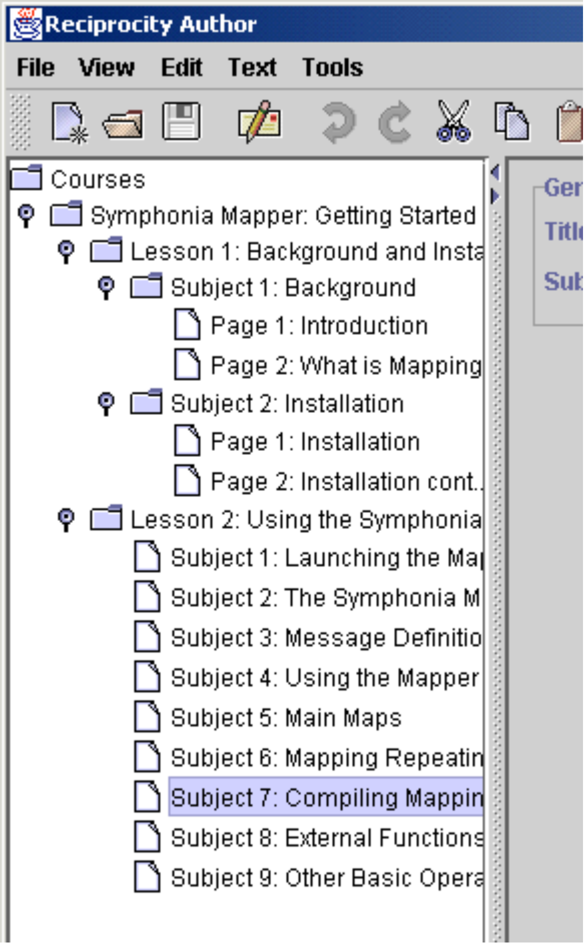
Figure 7.5: Choosing to delete a subject



- Courses
  - Symphonia Mapper: Getting Started
    - Lesson 1: Background and Insta
      - Subject 1: Background
        - Page 1: Introduction
        - Page 2: What is Mapping
      - Subject 2: Installation
        - Page 1: Installation
        - Page 2: Installation cont.
    - Lesson 2: Using the Symphonia
      - Subject 1: Launching the Ma
      - Subject 2: The Symphonia M
      - Subject 3: Message Definitio
      - Subject 4: Using the Mapper
      - Subject 5: Main Maps
      - Subject 6: Mapping Repeatin
      - Subject 7: External Functions
      - Subject 8: Other Basic Opera
      - Subject 9: Compiling Mappin

Figure 7.6 Result of deleting a subject

If I now wanted to change the order of the subjects, for example to insert the last subject, “Compiling Mappings” as the seventh subject, I would click on the subject’s node and drag it to the desired position. This automatically re-orders and renumbers the subjects from the inserting position down. The result of moving this subject is shown in Figure 7.7.



Reciprocity Author

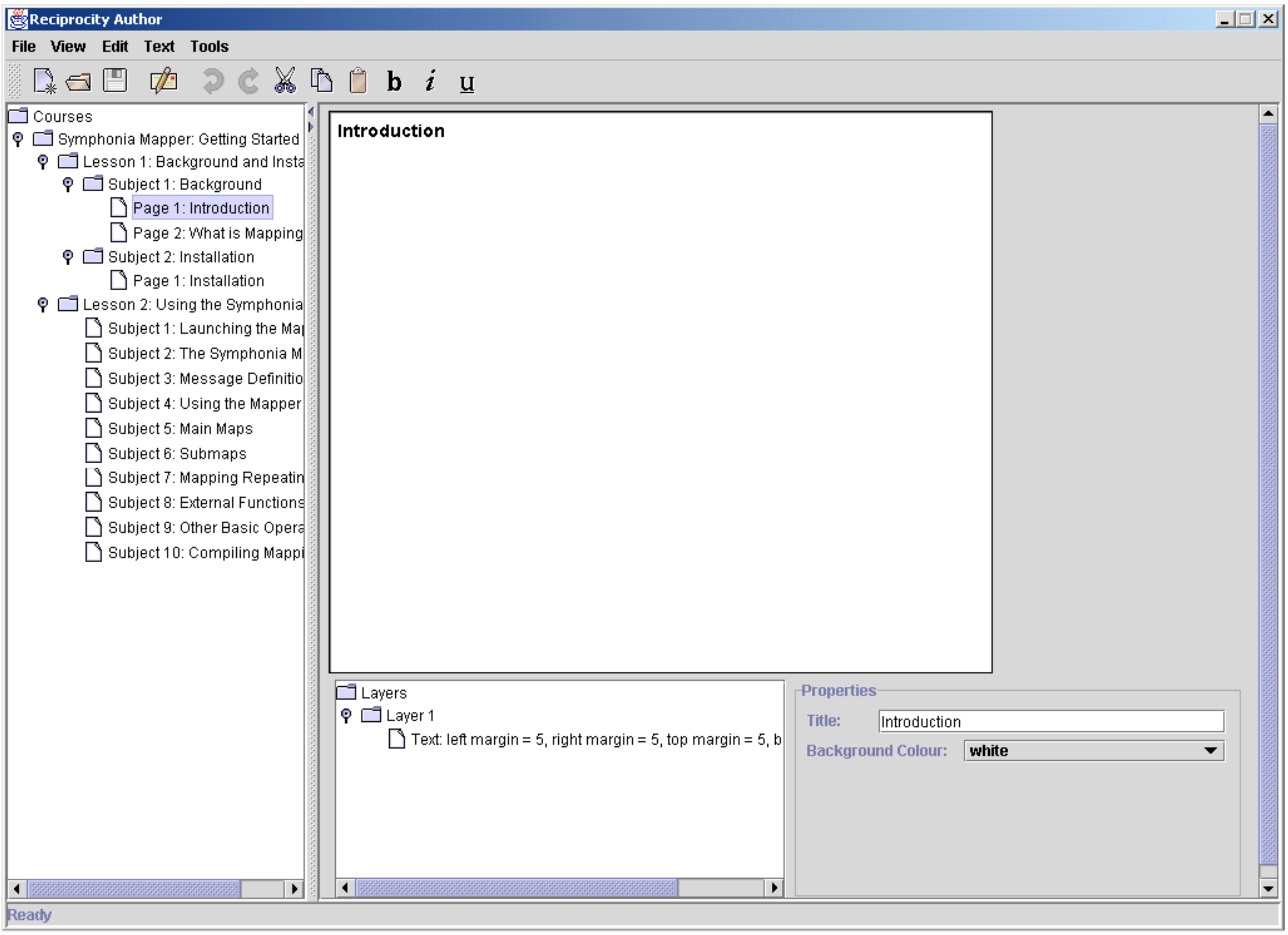
File View Edit Text Tools

Courses
Symphonia Mapper: Getting Started
Lesson 1: Background and Insta
Subject 1: Background
Page 1: Introduction
Page 2: What is Mapping
Subject 2: Installation
Page 1: Installation
Page 2: Installation cont.
Lesson 2: Using the Symphonia
Subject 1: Launching the Ma
Subject 2: The Symphonia M
Subject 3: Message Definitio
Subject 4: Using the Mapper
Subject 5: Main Maps
Subject 6: Mapping Repeatin
Subject 7: Compiling Mappin
Subject 8: External Functions
Subject 9: Other Basic Opera

Figure 7.7: Result of re-ordering subjects

7.3.3 Creating Content

The next step was to create the content of the course by designing the text, images and animations to place on individual pages. By selecting the first page from the structure tree, entitled ‘Introduction’, the display in the right hand side of the user interface changes from showing properties to showing a page design canvas, a tree representation of the layers of the page and a properties sheet. At the same time, the text editing functions become available in both the menu bar and the toolbar. As shown in Figure 7.8, the title of the page is automatically placed onto the page, however this can be deleted if desired by selecting it and pressing the delete key.


Reciprocity Author
File View Edit Text Tools
b i u
Courses
Symphonia Mapper: Getting Started
Lesson 1: Background and Insta
Subject 1: Background
Page 1: Introduction
Page 2: What is Mapping
Subject 2: Installation
Page 1: Installation
Lesson 2: Using the Symphonia
Subject 1: Launching the Ma
Subject 2: The Symphonia M
Subject 3: Message Definitio
Subject 4: Using the Mapper
Subject 5: Main Maps
Subject 6: Submaps
Subject 7: Mapping Repeatin
Subject 8: External Functions
Subject 9: Other Basic Opera
Subject 10: Compiling Mappi
Introduction
Layers
Layer 1
Text: left margin = 5, right margin = 5, top margin = 5, b
Properties
Ready

Title: Introduction

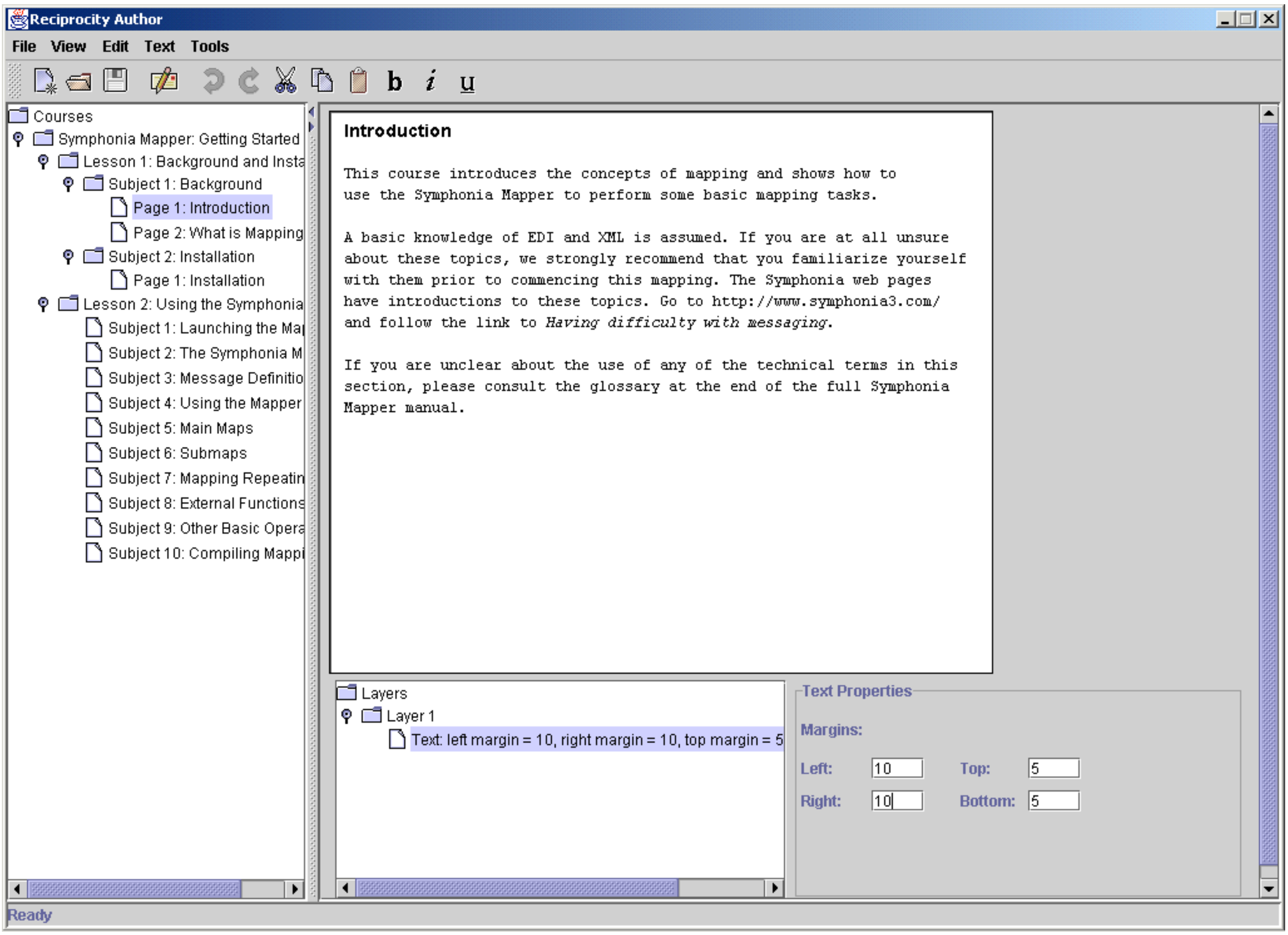
Background Colour: white

Figure 7.8: Displaying a new page in the page design area

Text

By placing the cursor anywhere on the page design canvas, the default text element that is automatically placed on every page becomes highlighted in the page layer tree and its particular properties are displayed in the properties sheet. After doing this, I was then able to type text directly onto the page following a document-editing metaphor as would be found in a word processor-type application. For example, to type text at the bottom of the page I would have to press the 'enter' key until the cursor was in the desired position and then begin typing.

At this stage I entered the text I wanted on the introductory page, as shown in Figure 7.9. I decided I wanted to format the words 'Having difficulty with messaging' as italic to accentuate and differentiate them. I did this by selecting the phrase with the mouse and pressing the italic toolbar button. Alternatively, I could have selected the Style submenu from the Text menu and chosen the Italic option from there. I also decided to increase the left and right margins of the text on the page from five to ten pixels. To do this, I replaced the values of 5 by entering 10 in the input boxes for the left and right margins in the text properties sheet, as shown at the bottom of Figure 7.9, and pressed 'enter' to action the changes. The text was automatically realigned to fit within these new margins.


Reciprocity Author
File
View
Edit
Text
Tools
b
i
u
Courses
Symphonia Mapper: Getting Started
Lesson 1: Background and Insta
Subject 1: Background
Page 1: Introduction
Page 2: What is Mapping
Subject 2: Installation
Page 1: Installation
Lesson 2: Using the Symphonia
Subject 1: Launching the Ma
Subject 2: The Symphonia M
Subject 3: Message Definitio
Subject 4: Using the Mapper
Subject 5: Main Maps
Subject 6: Submaps
Subject 7: Mapping Repeatin
Subject 8: External Functions
Subject 9: Other Basic Opera
Subject 10: Compiling Mappi
Layers
Layer 1
Text Properties
Ready

Introduction

```
This course introduces the concepts of mapping and shows how to
use the Symphonia Mapper to perform some basic mapping tasks.

A basic knowledge of EDI and XML is assumed. If you are at all unsure
about these topics, we strongly recommend that you familiarize yourself
with them prior to commencing this mapping. The Symphonia web pages
have introductions to these topics. Go to http://www.symphonia3.com/
and follow the link to Having difficulty with messaging.

If you are unclear about the use of any of the technical terms in this
section, please consult the glossary at the end of the full Symphonia
Mapper manual.
```

Text: left margin = 10, right margin = 10, top margin = 5

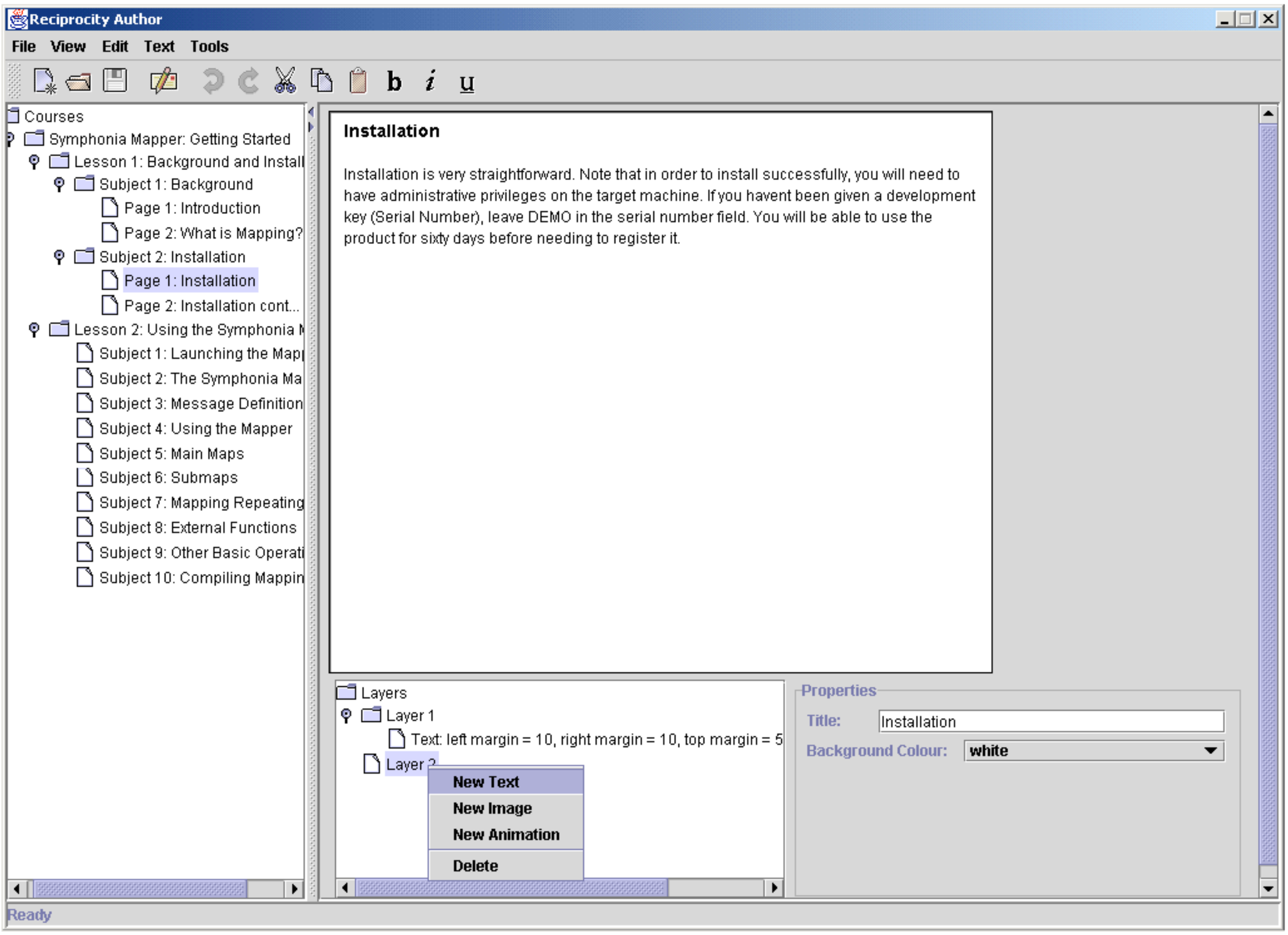
Margins:

Left:	10	Top:	5
Right:	10	Bottom:	5

Figure 7.9: Entering text and changing text margins

Images

On the first page of the second subject, entitled 'Installation', I wanted to place an image as well as text. To ensure that the image would always be placed above any text on the page, I created a new layer in which to place it. To do this, I right clicked on the root node of the layer tree and selected 'new layer' from the popup menu, as shown in Figure 7.10. The new layer immediately appeared in the tree. To add the image to this new layer, I right clicked on the layer and chose 'new image' from the popup menu. This opened the image selection dialog as described earlier. Once I found the image I wanted in the list, I pressed 'Insert', which closed the dialog and placed the image by default at coordinate (0,0), or the top left hand corner of the page. The image was also added to the layer tree, in Layer 2, as shown in Figure 7.11.



Reciprocity Author
File View Edit Text Tools
b i u
Courses
Symphonia Mapper: Getting Started
Lesson 1: Background and Install
Subject 1: Background
Page 1: Introduction
Page 2: What is Mapping?
Subject 2: Installation
Page 1: Installation
Page 2: Installation cont...
Lesson 2: Using the Symphonia M
Subject 1: Launching the Map
Subject 2: The Symphonia Ma
Subject 3: Message Definition
Subject 4: Using the Mapper
Subject 5: Main Maps
Subject 6: Submaps
Subject 7: Mapping Repeating
Subject 8: External Functions
Subject 9: Other Basic Operati
Subject 10: Compiling Mappin
Layers
Layer 1
Text: left margin = 10, right margin = 10, top margin = 5
Layer 2
New Text
New Image
New Animation
Delete
Properties
Ready

Installation

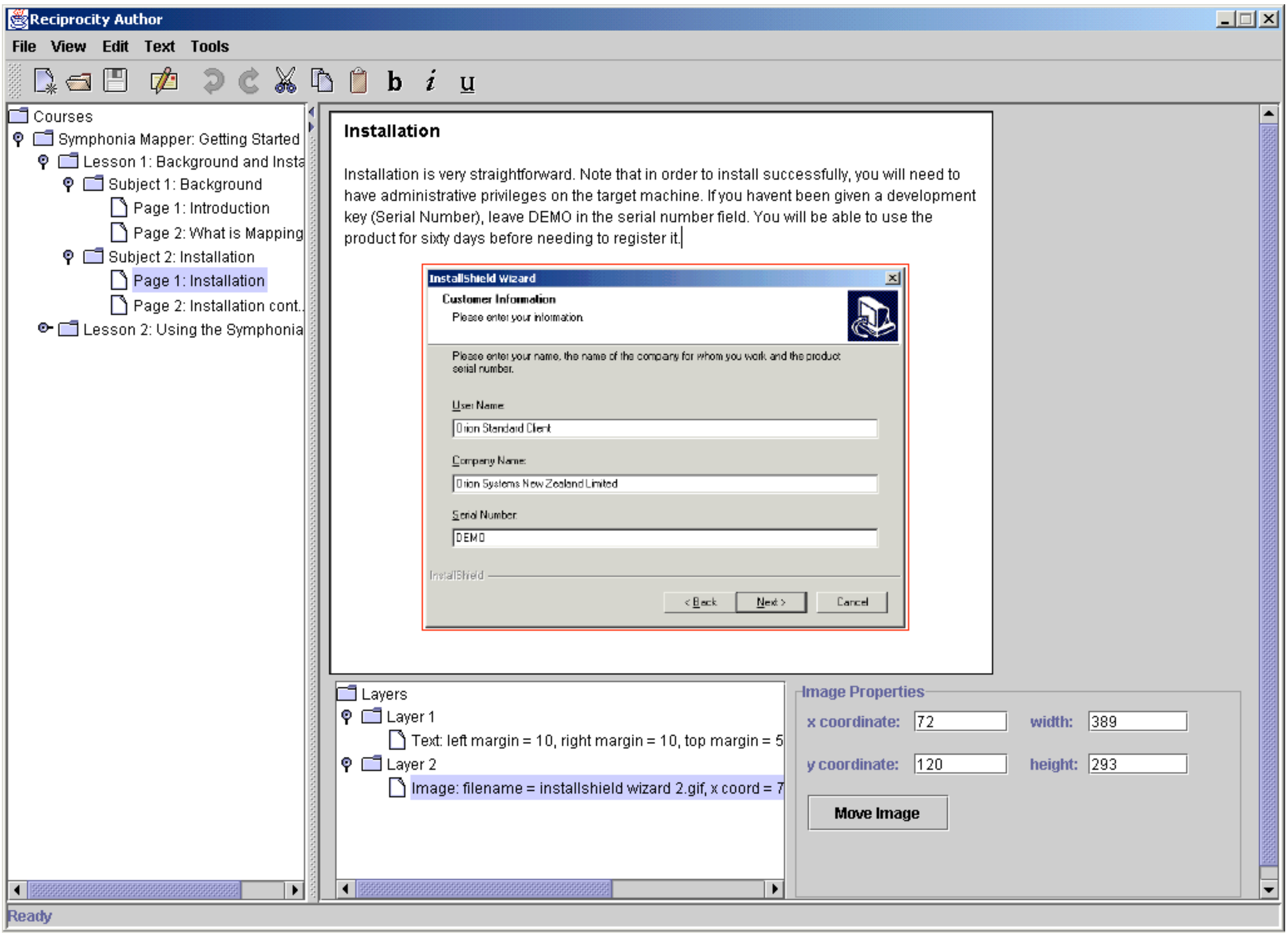
Installation is very straightforward. Note that in order to install successfully, you will need to have administrative privileges on the target machine. If you havent been given a development key (Serial Number), leave DEMO in the serial number field. You will be able to use the product for sixty days before needing to register it.

Title: Installation

Background Colour: white

Figure 7.10: Choosing new image from the layer popup menu

In the same way as for the textual element on this page, selecting the image, either by clicking on it in the page design canvas or by clicking on it in the layer tree, displays its specific properties in the properties sheet, as well as a function for moving the image on the page. This can be seen at the bottom of Figure 7.11. Since I wanted to place the image at the bottom of the page, below the text already entered, I pressed the 'move image' button and then clicked no the page where I wanted the top left hand corner of the image to go. It was immediately moved to this position. Alternatively, I could have entered the exact coordinates in the input boxes for these properties, but this function is intended more for the precise or very slight relocation of images, which was not required in this situation.



Courses
- Symphonia Mapper: Getting Started
  - Lesson 1: Background and Insta
    - Subject 1: Background
      - Page 1: Introduction
      - Page 2: What is Mapping
    - Subject 2: Installation
      - Page 1: Installation
      - Page 2: Installation cont.
  - Lesson 2: Using the Symphonia

Installation

Installation is very straightforward. Note that in order to install successfully, you will need to have administrative privileges on the target machine. If you havent been given a development key (Serial Number), leave DEMO in the serial number field. You will be able to use the product for sixty days before needing to register it.

InstallShield Wizard

Customer Information
Please enter your information.

Please enter your name, the name of the company for whom you work and the product serial number.

User Name: Orion Standard Client

Company Name: Orion Systems New Zealand Limited

Serial Number: DEMO

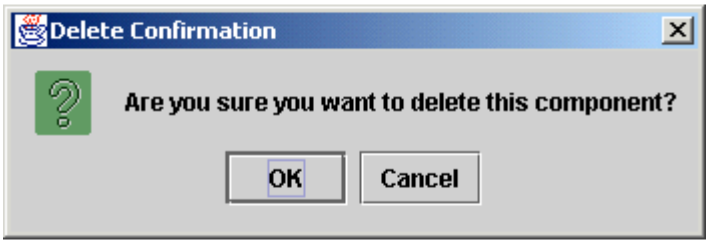
Layers
- Layer 1
  - Text: left margin = 10, right margin = 10, top margin = 5
- Layer 2
  - Image: filename = installshield wizard 2.gif, x coord = 7

Image Properties

x coordinate:	72	width:	389
y coordinate:	120	height:	293

Figure 7.11: The new image can be seen on the page and in the layer tree

At this point, if I realised that I had inserted an incorrect image or simply wanted to change the image, my first step would be to delete the image that is currently on the page. To do this, I would right click on the image's node in the layer tree and select 'delete' from the popup menu displayed. I would be prompted for confirmation that I did indeed want to delete this image, as shown in Figure 7.12, and after pressing 'OK' the image would be deleted from both the page design canvas and the layer tree.



Are you sure you want to delete this component?

Figure 7.12: Prompting for confirmation before deleting an image

Animation

To allow users to experience the process of installation, rather than simply viewing an image of the first of many screens they would be presented with, I decided to remove the image I had previously placed on the 'Installation' page and place an animation in its place. Adding an animation to a page is done in a very similar way to adding an image. Since I already had a suitable layer with no media elements in it, which originally held the image of the first installation screen, I used this to hold the new animation. I selected 'new animation' from option from the layer's popup menu and was presented with a separate Animation Designer window, as shown in Figure 7.13. The window contains a tree showing the structure of the animation on the left, which is initially empty, and a display area on the right, which will eventually show the images of each step in the animation. Along the bottom of the window is a set of functional buttons.

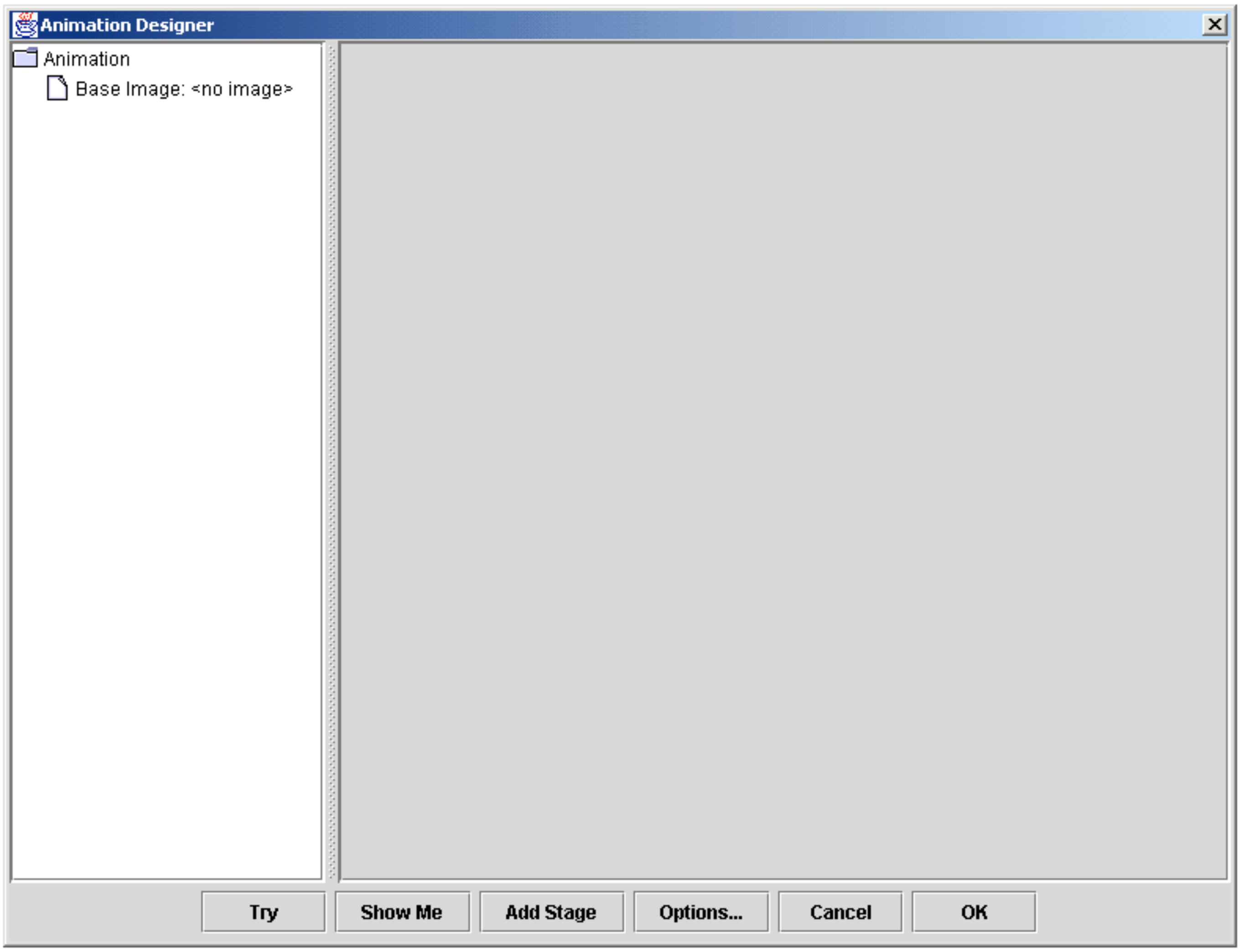


Figure 7.13: Initial Animation Designer screen

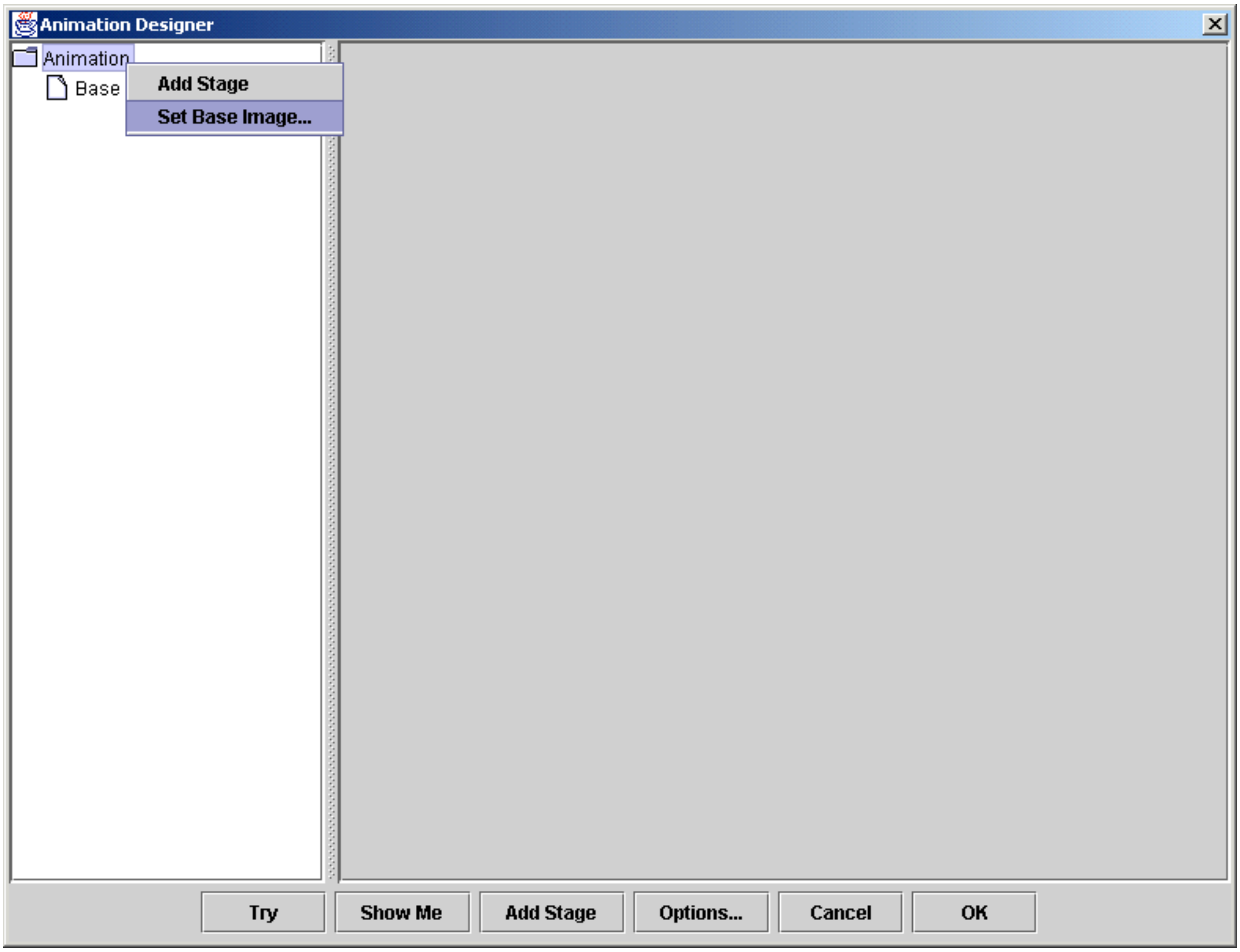
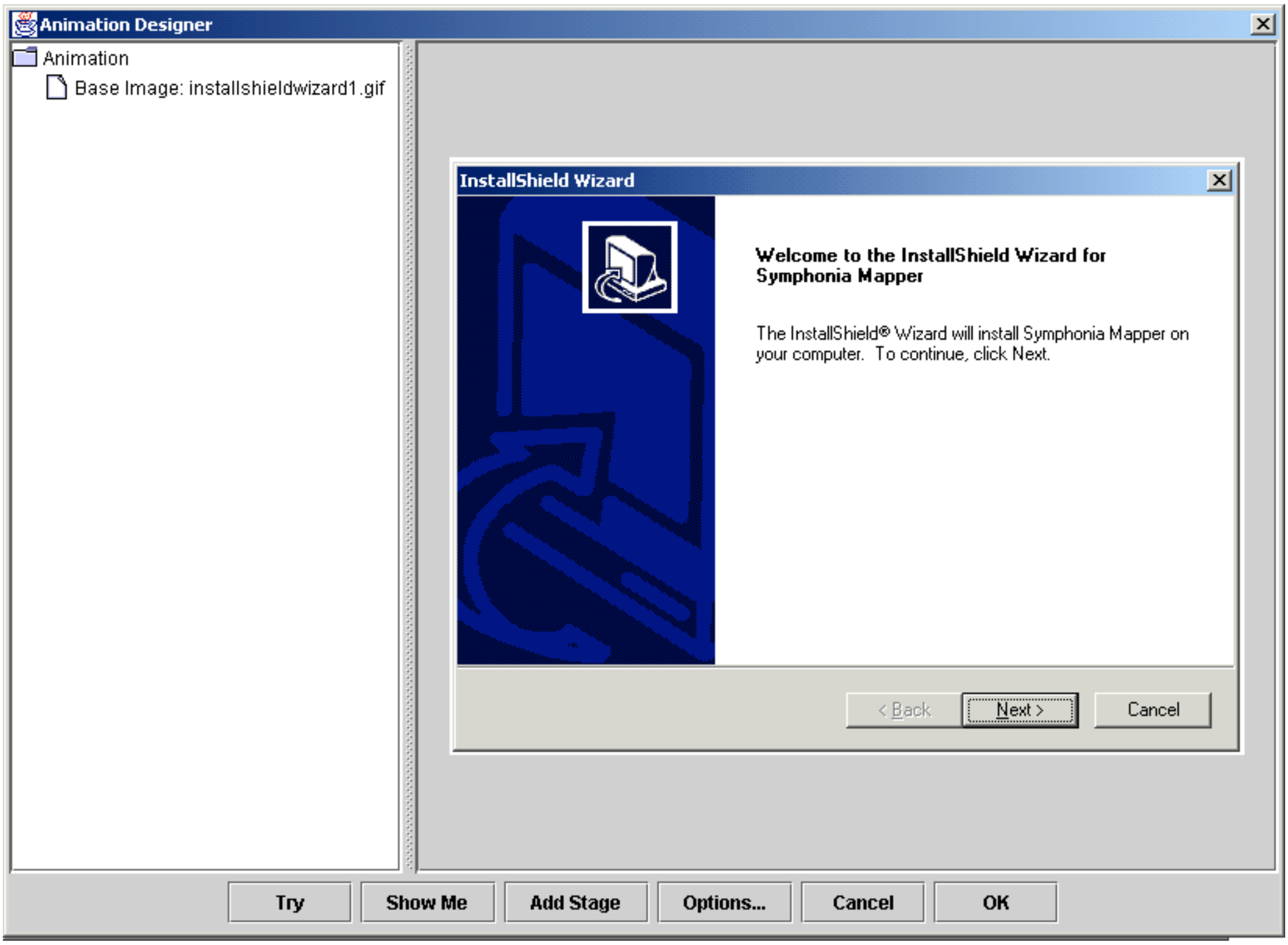


Figure 7.14: Choosing to set the animation's base image

The first step is to set the base image of the animation. This is the image that will be placed on the course page to indicate that there is an animation to play and is also the starting point of interactivity, for instance, the first screen in a series of screens that make up a dialog. To set the base image, I right clicked on the root node of the animation's structure tree and selected 'set base image…' from the popup menu that appeared, as shown in Figure 7.14. This opened an image selection dialog, as previously described, from which I selected the first screen of the installation dialog as the base image. After choosing the image, the animation's structure was updated to reflect the selection and the image was displayed in the right hand side of the window. This is shown in Figure 7.15.



Welcome to the InstallShield Wizard for Symphonia Mapper

The InstallShield® Wizard will install Symphonia Mapper on your computer. To continue, click Next.

Figure 7.15: After setting the animation's base image

The next step I took was to add a stage to the animation. A stage constitutes one step in an animation, including an acceptable user action and location for this action, which is carried out on the previous image, the image that results from carrying out this action and a time delay, used in the automatic demonstration of the animation. To add a stage, I pressed the 'Add Stage' button at the bottom of the screen. Since no parameters are required for this action, an empty stage is immediately added to the structure of the animation, as can be seen in Figure 7.16.

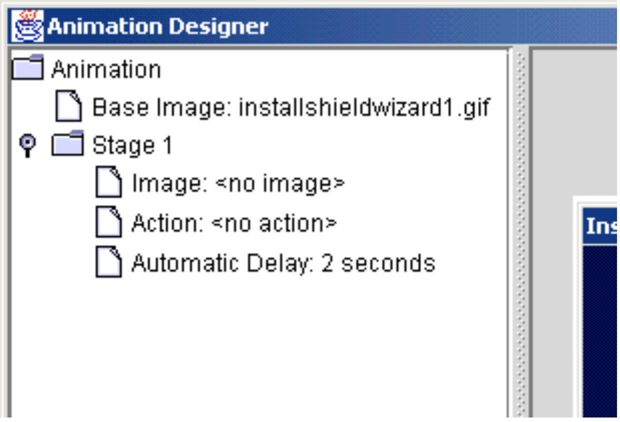


Figure 7.16: After adding a stage to an animation

All of the functions that can be performed on a stage are available in a popup menu that appears after right clicking on the stage's node, as shown in Figure 7.17. From this menu, I chose to edit the action of the stage, which opened a separate action editing dialog, as can be seen in Figure 7.18.

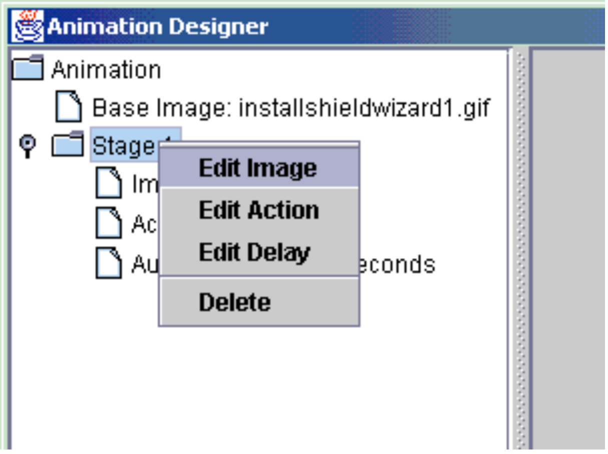
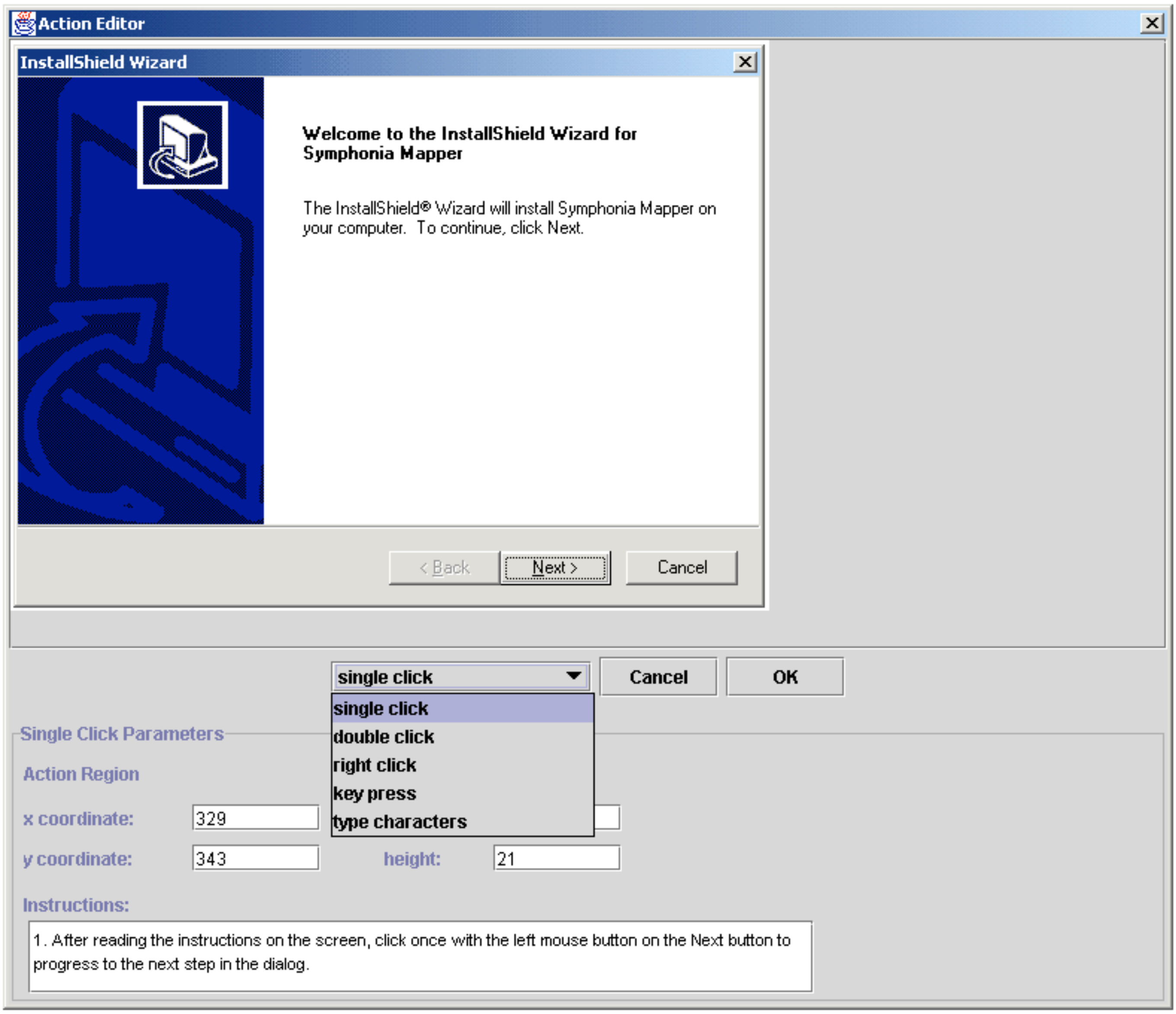


Figure 7.17: Stage popup menu



Welcome to the InstallShield Wizard for Symphonia Mapper

The InstallShield® Wizard will install Symphonia Mapper on your computer. To continue, click Next.

single click

Single Click Parameters

Action Region

x coordinate: 329

y coordinate: 343

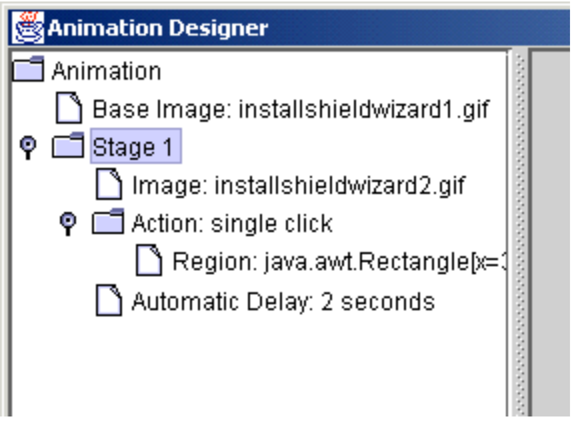
height: 21

Instructions:

1. After reading the instructions on the screen, click once with the left mouse button on the Next button to progress to the next step in the dialog.

Figure 7.18: Editing the action of an animation stage

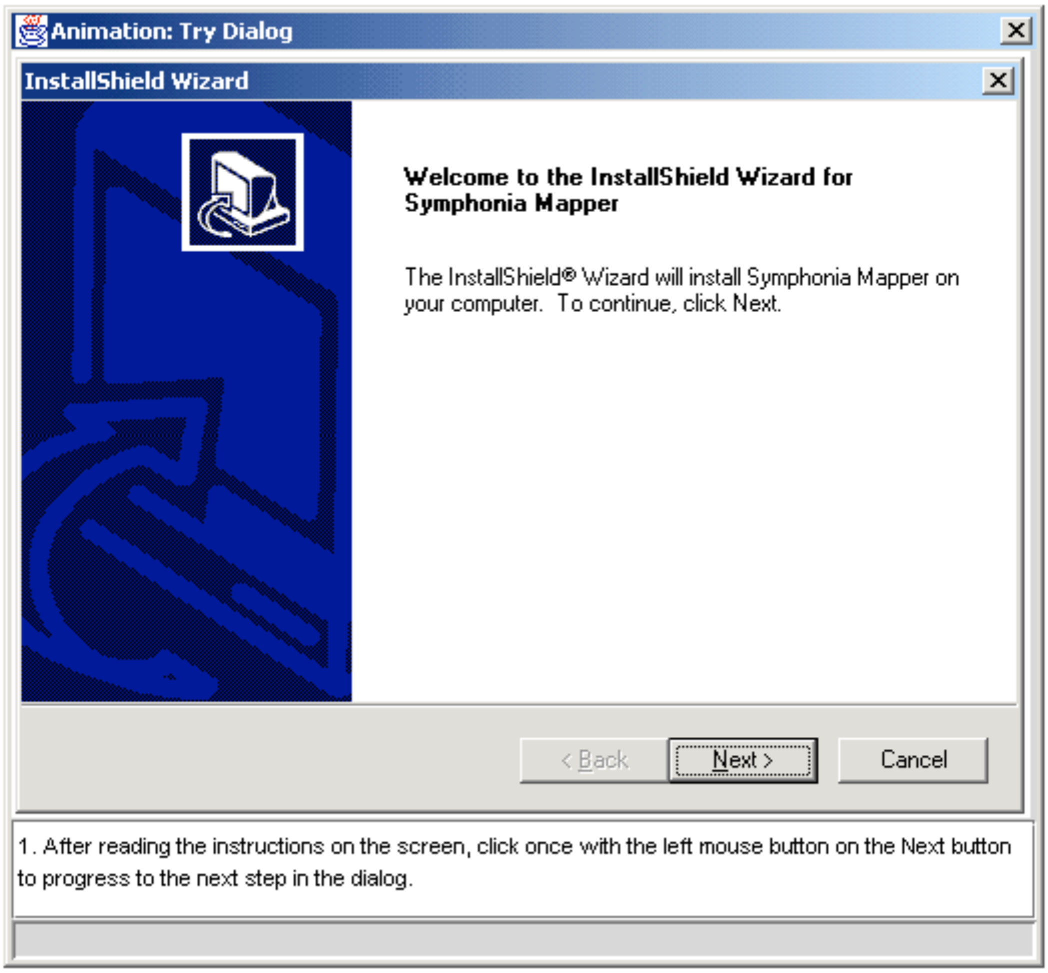
The top section of the dialog displays the pervious image, upon which the action will be taken. In this instance, since I was specifying the first stage of the animation, the base image is displayed. However, for all following stages, the image of the previous stage is displayed. The middle section of the dialog contains a combo box and two buttons. The combo box allows the user to choose the type of action they want to be carried out. Possible actions include single click, double click, right click, key press and type characters. Once an action is selected, the bottom section of the dialog displays a panel containing the parameters of the action. For a single click action, as shown in Figure 7.18, the parameters include the region in which the click is acceptable, consisting of the x and y coordinates of the top left hand corner of the region and its width and height, and instructions to be given to the learner when they are carrying out the action. To specify the acceptable region I clicked and dragged a rectangle directly on the image displayed. The rectangle appears in white and the coordinates of the rectangle are automatically entered into the parameter panel. Only a single acceptable region can be specified but it can be easily corrected by repeating the click and drag process until the region is correct. At this point I was satisfied with the action I had created, so pressed ‘OK’, which closed the action-editing dialog and returned to the animation designer window. The structure of the animation was updated to reflect the specified action of the first stage, as can be seen in Figure 7.19.



Animation
- Base Image: installshieldwizard1.gif
- Stage 1
  - Image: installshieldwizard2.gif
  - Action: single click
    - Region: java.awt.Rectangle[x=
  - Automatic Delay: 2 seconds

Figure 7.19: Structure of the animation after adding one complete stage

I proceeded to set the resulting image of stage one by selecting ‘edit image’ from the stage’s popup menu and selecting the desired image from the image selection dialog, as previously described. I repeated this process of adding a stage, specifying its action and resulting image for four more stages, until the animation of the installation process was complete.



Welcome to the InstallShield Wizard for Symphonia Mapper

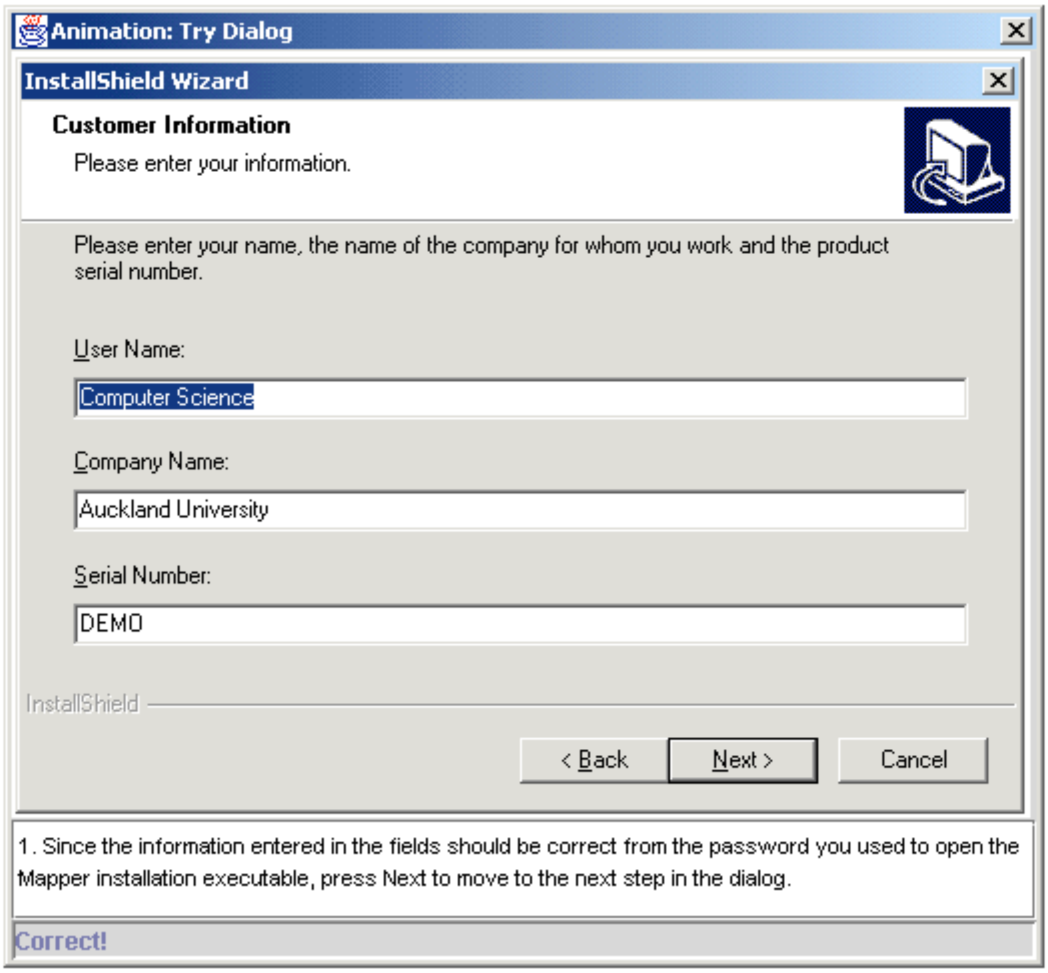
The InstallShield® Wizard will install Symphonia Mapper on your computer. To continue, click Next.

1 . After reading the instructions on the screen, click once with the left mouse button on the Next button to progress to the next step in the dialog.

Figure 7.20: Initial screen in animation 'try' dialog

My next step was to run through the animation using the 'try' function. This function has been included so that authors may test animations to ensure they have been fully specified, work correctly and accurately represent the situation as it occurs in the real software. To do this, I pressed the 'Try' button at the bottom of the animation designer user interface. This opened a separate dialog, as shown in Figure 7.20. The dialog simply consists of an image displayed in the top section, instructions for carrying out the correct action and a status bar, used to give feedback to users on the correctness of their actions.

Following the instructions given, I read the blurb on the installation screen and then clicked on the 'Next' button. The result of this is shown in Figure 7.21, where the status bar now reads "Correct!" and the image and instructions have changed to those of the next stage in the animation. If I had carried out an incorrect action, such as clicking somewhere else on the image, the status bar would have read "Incorrect action" and the image and instructions would have remained the same so that I could make another attempt. Once all stages have been completed, the status bar and instructions tell the author that they have completed the animation and can close the dialog.



Please enter your name, the name of the company for whom you work and the product serial number.

User Name: Computer Science

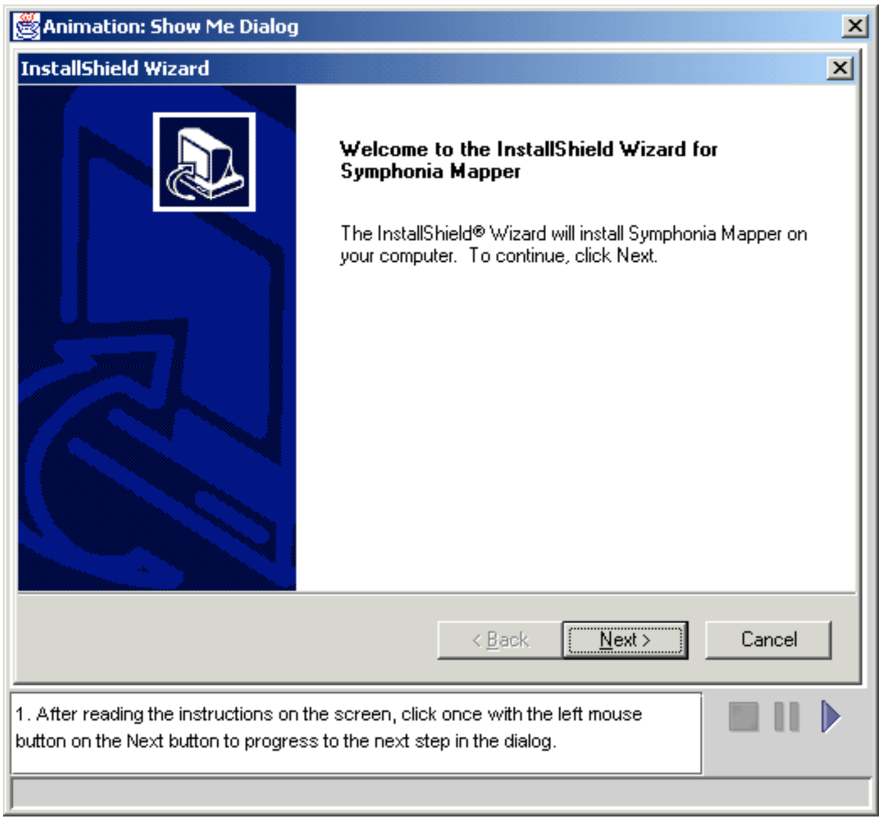
Company Name: Auckland University

Serial Number: DEMO

1. Since the information entered in the fields should be correct from the password you used to open the Mapper installation executable, press Next to move to the next step in the dialog.

Figure 7.21: Screen presented after first user action in 'try' dialog

Another feature that I used to test the animation was the 'Show Me' function. Similarly to the 'Try' function, pressing the 'Show Me' button opened a separate dialog that was almost identical to the try dialog, except for the addition of a set of control buttons, consisting of play, pause and stop. The 'show me' function differs from 'try' in that it does not monitor for user actions, but instead plays the stages of the animation automatically, allowing the animation to be demonstrated. The timing of the animation is determined by the automatic delay specified during design of the animation. The default value for the time between stages is two seconds. Once I pressed play, the animation began, displaying the current stage in the status bar and pausing two seconds before displaying each subsequent image, along with its instructions. The control buttons became active, allowing me to pause or stop the animation, as shown in Figure 7.23. By pressing the pause button, I could then resume the animation by pressing pause again, and I could stop the animation at any time. Once the demonstration was complete, I could have restarted it by pressing start again, but since I was satisfied with the animation, I closed the dialog.

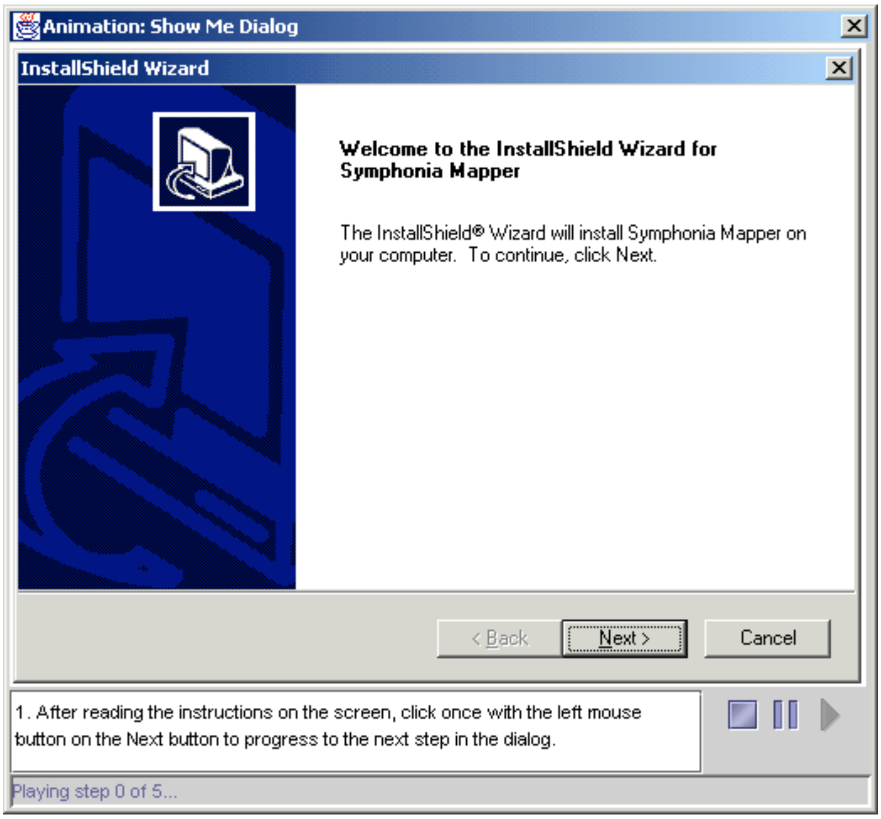


Welcome to the InstallShield Wizard for Symphonia Mapper

The InstallShield® Wizard will install Symphonia Mapper on your computer. To continue, click Next.

1. After reading the instructions on the screen, click once with the left mouse button on the Next button to progress to the next step in the dialog.

Figure 7.22: Initial screen in animation 'show me' dialog



Welcome to the InstallShield Wizard for Symphonia Mapper

The InstallShield® Wizard will install Symphonia Mapper on your computer. To continue, click Next.

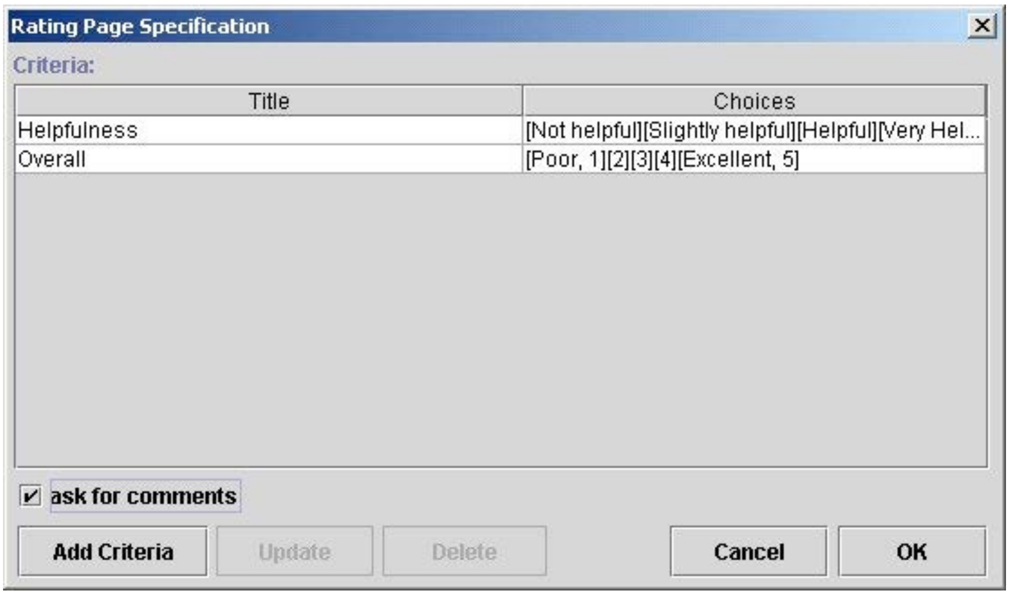
1. After reading the instructions on the screen, click once with the left mouse button on the Next button to progress to the next step in the dialog.

Playing step 0 of 5...

Figure 7.23: During playing of demonstration in animation 'show me' dialog

7.3.4 Setting Feedback Options

In order to receive explicit feedback from users on their perceptions of the training course I had created, I created a course rating page. To design the rating page, I right clicked on the course node in the structure tree and selected 'new rating page' from the popup menu that appeared. This opened a rating page specification dialog, as shown in Figure 7.24.

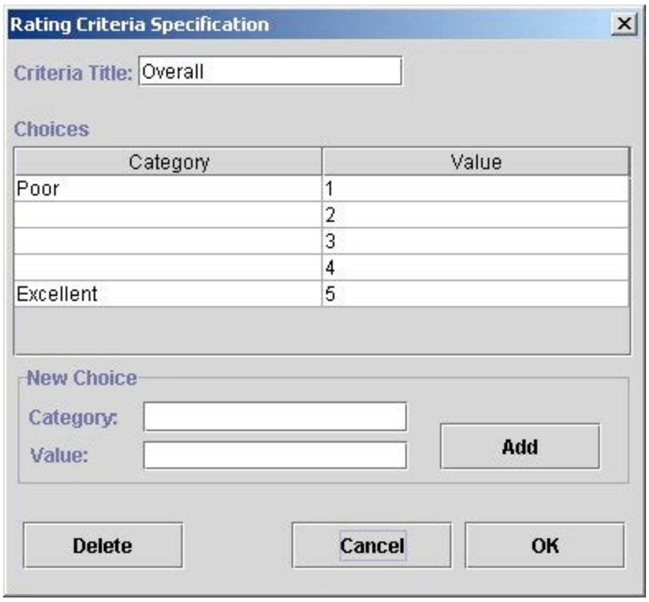


Title	Choices
Helpfulness	[Not helpful][Slightly helpful][Helpful][Very Hel...
Overall	[Poor, 1][2][3][4][Excellent, 5]

☑ ask for comments

Figure 7.24: rating page specification dialog

The dialog consists of a table of the currently set criteria for this rating page, including their choices, and a row of function buttons. At this stage, no criteria were set since this was a new rating page. To create a new criterion, I pressed 'add criteria'. This opened a new dialog as shown in Figure 7.25. I first entered the title of the criterion, in this case 'overall', and added as five choices, correlating to the numbers 1-5, with 1 being 'poor' and 5 being 'excellent'. To add each new choice, I entered both a category and a value and pressed 'Add'. If I had added an incorrect choice, to delete this I would have selected it from the list and pressed 'Delete'. The order in which the choices are specified is the order they will appear in the generated rating page.



Rating Criteria Specification

Criteria Title: Overall

Choices

Category	Value
Poor	1
	2
	3
	4
Excellent	5

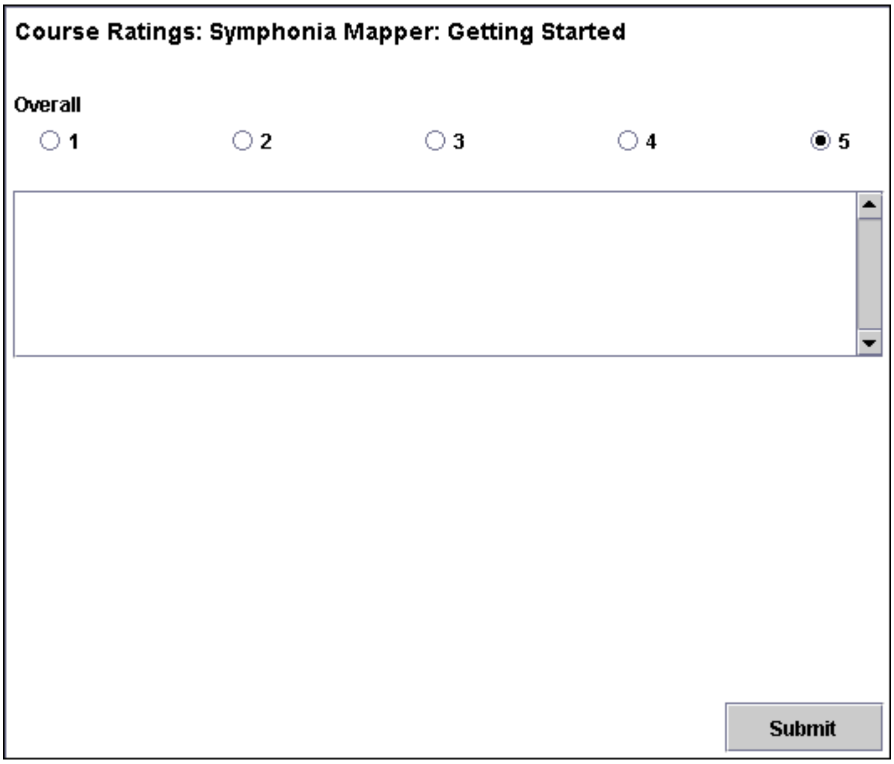
New Choice

Category:

Value:

Figure 7.25: rating criteria specification dialog

Once I had completed specifying the rating criteria, I selected the checkbox provided for asking for comments and pressed "OK". At this stage, a rating page was generated and added to the course structure tree. The appearance of the rating page is shown in Figure 7.26.



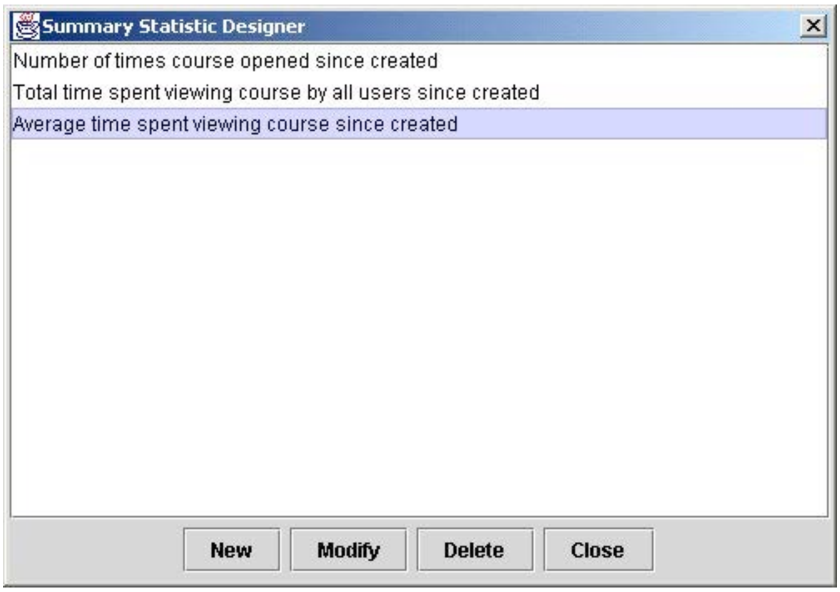
Course Ratings: Symphonia Mapper: Getting Started

Overall

○ 1 ○ 2 ○ 3 ○ 4 ● 5

Figure 7.26: generated course rating page

Now that it was possible for users to provide explicit feedback as well as the feedback implicitly gathered on their course usage, I wanted to specify summary statistics to be reported. To do this, I first accessed the summary statistic designer through the Tools menu in the menu bar at the top of the screen. This opened the designer, as shown in Figure 7.27, which initially displays all statistics currently set for this particular course and this particular author. Since no statistics had yet been defined, the list was empty. To create a new statistic I choose the 'new' function from those available at the bottom of the dialog. I then selected the desired statistic and time period from the drop down lists provided. In this case, I choose the statistics 'number of times course opened', 'total time spent viewing course by all users' and 'average time spent viewing the course' all for the time period 'since created'.



Number of times course opened since created

Total time spent viewing course by all users since created

Average time spent viewing course since created

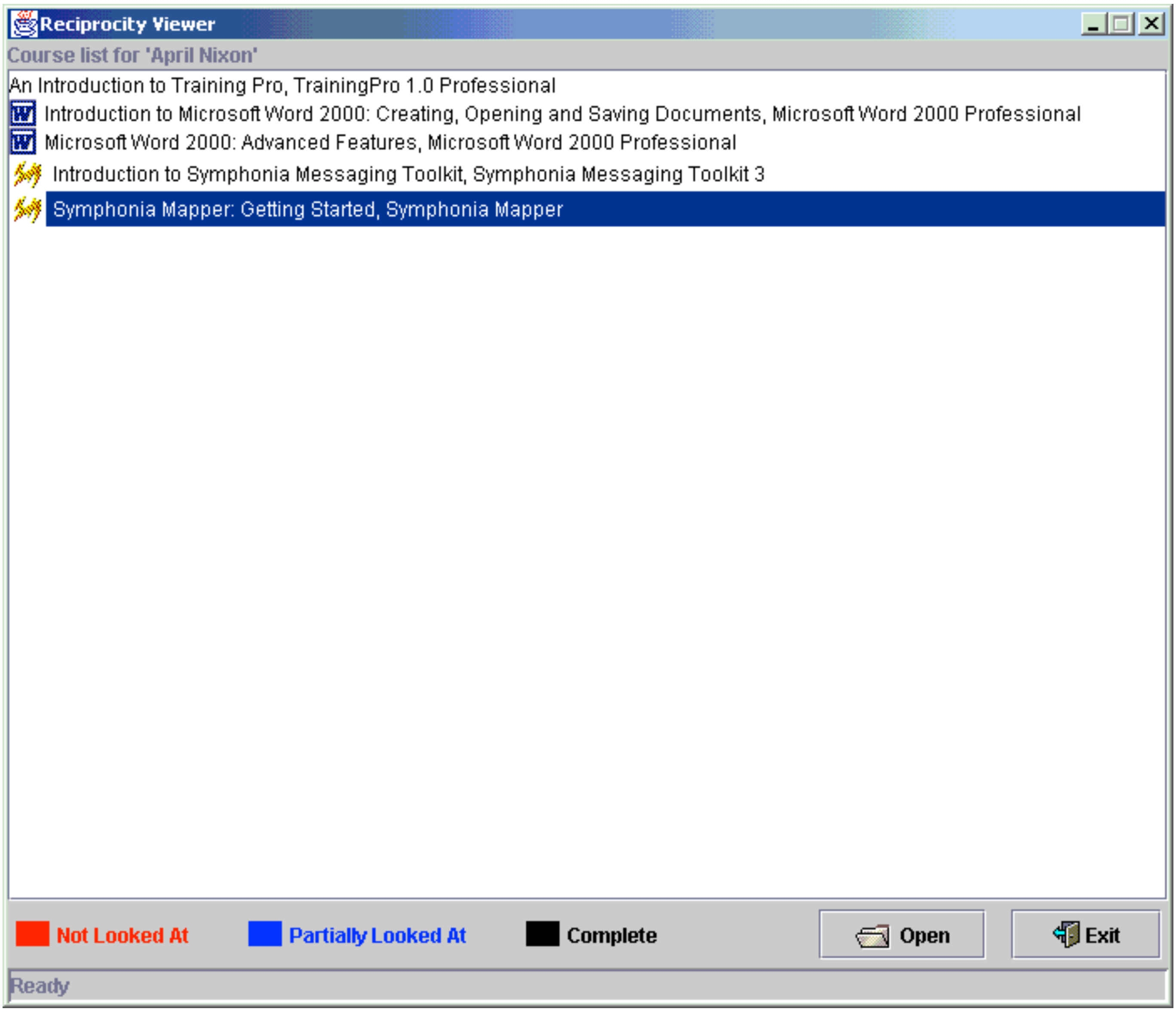
Figure 7.27: summary statistic design dialog

Once I had closed the dialog, I could then view the updated set of statistics by selecting the course node in the course structure tree. All statistics currently specified were displayed, with current values of 0, along with the course properties on the right hand side of the main screen. These statistics would now be available at all times and have their values updated every time this course is loaded into the authoring tool.

7.4 Viewing the Course

This section describes a typical usage of the training course authored in the previous section of this chapter.

The first action that a learner takes after launching the viewing application is to log in. Once I did this, I was presented with a personalised list of training courses. This is shown in Figure 7.28. Courses are included in the list based on the learner belonging to an organisation that has a current software license for the software the course teaches. In the example shown, five courses are listed, including the title of the course, the title of the software it teaches and optionally the software's icon, for faster visual recognition.

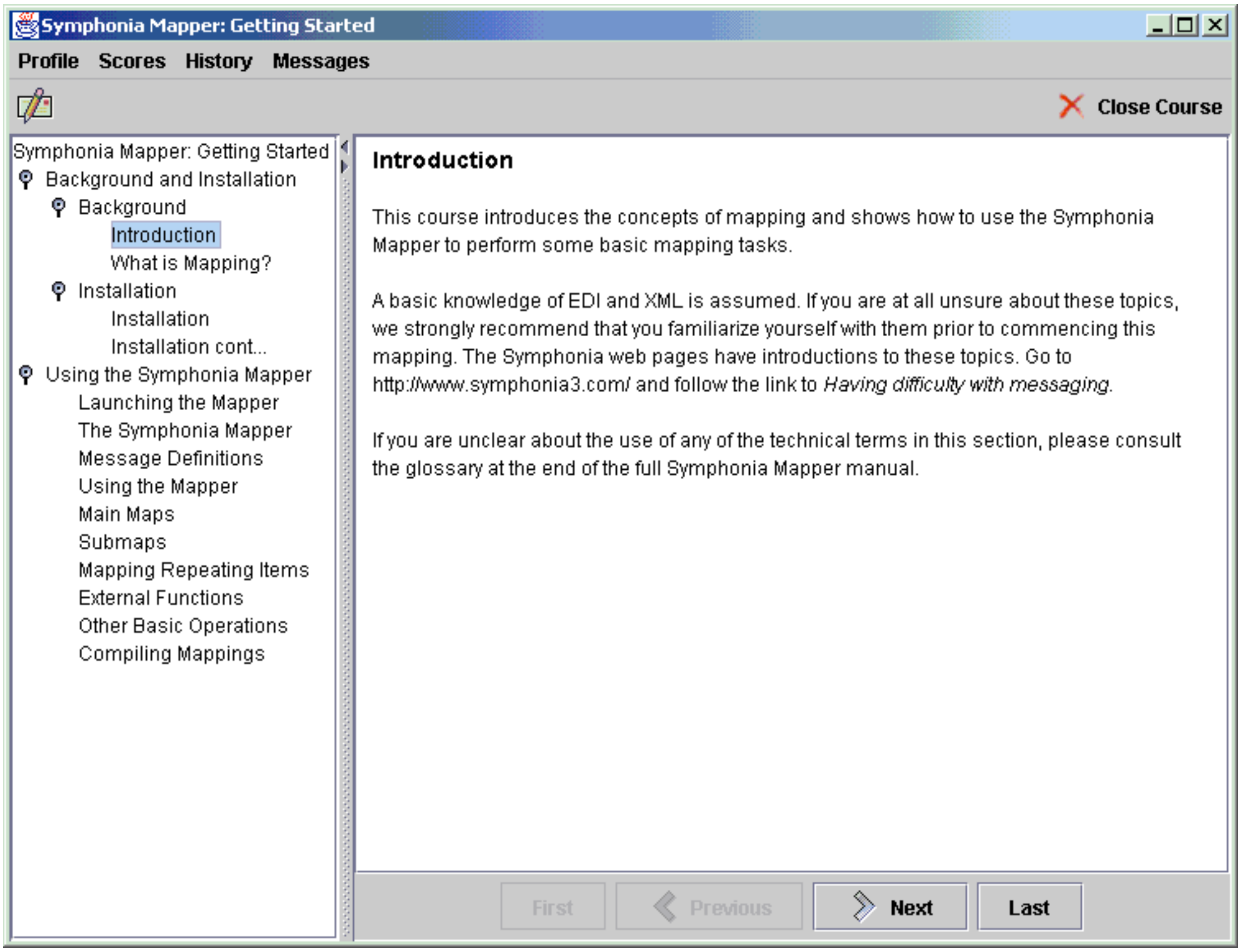


Course list for 'April Nixon'

An Introduction to Training Pro, TrainingPro 1.0 Professional
Introduction to Microsoft Word 2000: Creating, Opening and Saving Documents, Microsoft Word 2000 Professional
Microsoft Word 2000: Advanced Features, Microsoft Word 2000 Professional
Introduction to Symphonia Messaging Toolkit, Symphonia Messaging Toolkit 3
Symphonia Mapper: Getting Started, Symphonia Mapper

Figure 7.28: Personalised course menu

At the top of the user interface the learner's name is displayed, in this case 'April Nixon' to confirm I was logged in correctly, while at the bottom of the interface is a legend that describes the use of colour in the menu. This use of colour is to provide visitation progress feedback to learners. For example, if a learner has visited every page in every subject and lesson of a course, that course will be displayed in black, since the intention is not to draw the attention of the learner to material already visited. If a learner has only partially visited a course, it is displayed in blue and a course the learner has not visited at all will be displayed in red. Since I had previously completely visited all of the courses listed, they were all displayed in black.



Introduction

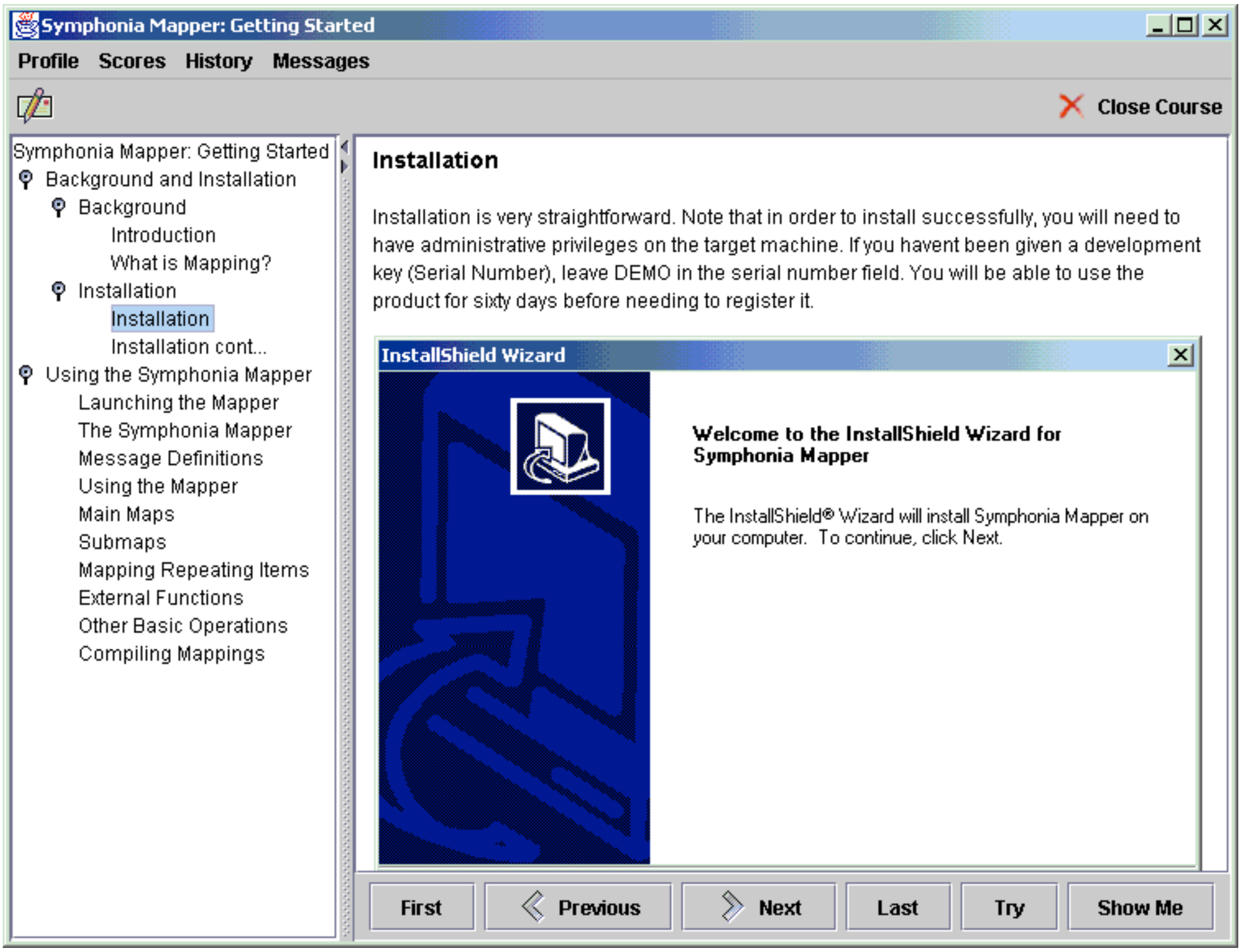
This course introduces the concepts of mapping and shows how to use the Symphonia Mapper to perform some basic mapping tasks.

A basic knowledge of EDI and XML is assumed. If you are at all unsure about these topics, we strongly recommend that you familiarize yourself with them prior to commencing this mapping. The Symphonia web pages have introductions to these topics. Go to http://www.symphonia3.com/ and follow the link to *Having difficulty with messaging*.

If you are unclear about the use of any of the technical terms in this section, please consult the glossary at the end of the full Symphonia Mapper manual.

Figure 7.29: Course viewer

I selected the course entitled "Symphonia Mapper: Getting Started" from the list and pressed 'Open' to view the course. The result of this is shown in Figure 7.29. The menu of courses is hidden and a new viewing window appears in its place. This window allows the learner to view the entire contents of a single course. It displays a table of contents on the left and the actual content of the course on the right. I decided I wanted to view the subject on installation, skipping the introductory subject. To do this I selected the page entitled 'Installation' from the table of contents by clicking on it. It immediately appeared on the right of the screen. Alternatively, I could have pressed the 'Next' button two times to move from the first page to the 'Installation' page. When the 'Installation' page was loaded, the navigation buttons at the bottom of the screen changed to include two more buttons: 'Try' and 'Show Me', as can be seen in Figure 7.30. The reason for this was that the page contained an animation. The image that can be seen on the page is the base image of that animation. I decided to try carrying out the task of installing the software so pressed the 'Try' button. This opened a separate dialog as described previously in this chapter. I completed the task and closed the dialog, which returned to the course viewer.



Installation

Installation is very straightforward. Note that in order to install successfully, you will need to have administrative privileges on the target machine. If you havent been given a development key (Serial Number), leave DEMO in the serial number field. You will be able to use the product for sixty days before needing to register it.

Welcome to the InstallShield Wizard for Symphonia Mapper

The InstallShield® Wizard will install Symphonia Mapper on your computer. To continue, click Next.

Figure 7.30: Course viewer displaying a page containing an animation

Once I had finished viewing the course, I closed the viewer by pressing the 'Close Course' button in the top right hand corner of the screen. This returned me to the original list of courses. Alternatively, I could have selected the 'close course' function from the File menu.

7.5 Summary

This chapter presented an in-depth description of building and viewing a course using the authoring and viewing tool prototypes. The section on the authoring tool described how to first create a new course, then build its structure, create content using all of the available types of media and finally to set up summary statistics to allow viewing of reporting information gathered from monitoring the course's usage. The section on the viewing tool discussed selecting and opening a specific course, the different ways to navigate a course as well as how to run and participate in an interactive software demonstration. Finally, it showed how to provide feedback to authors through rating and commenting on courses viewed.

Chapter 8 Evaluation

8.1 Introduction

This chapter discusses the methods used to evaluate the prototype tools and the results of the evaluation. First, we present the survey process, where subjects were enlisted to carry out specific tasks with the tools, then answer questions and provide general comments. This was carried out in two phases, the first phase making use of generic subjects and the tools in a partially completed state, and the second focusing on support staff from Orion Systems Ltd with the finished tools and a comprehensive training course for the Symphonia Messaging Toolkit, a suite of software developed by Orion Systems Ltd. We include summarised results and an analysis of these. Secondly, we present a summative evaluation of the tools using two heuristic approaches, the first a set of heuristics outlined by Squires [Squires 99] which focuses on the integration of learning and usability when evaluating educational multimedia software and the second presented by Nielsen [Nielsen 94] to comprehensively evaluate the usability of software in general.

The reason for conducting these two distinct types of evaluation was to assess the tools both as software objects in their own right and to evaluate the actual and perceived use of the tools and the training material produced by them.

8.2 Survey

A survey of user experience with the tools was carried out at two distinct times during this research. The first survey was conducted with the tools partially implemented using a range of subjects, all of whom qualified by having some past experience using software and electronic software training. The purpose of this survey was to canvas opinion on the concept of the tools, the state of those features already implemented and to help determine the direction of further implementation. The second survey was carried out on site at Orion Systems Ltd, using the finished tools and a comprehensive introductory training course for the Symphonia Messaging Toolkit. Software support staff at Orion were asked to carry out specific tasks with the tools, view the training course and then answer questions and provide general comments. The purpose of this survey was to assess the suitability of the concept of the tools for authoring and distributing software training.

8.2.1 First Round

8.2.1.1 Overview

To familiarise subjects with the prototype tools, each subject was provided with both a task sheet and a tutorial for each of the two tools. The task sheet consisted of instructions for accessing and carrying out the main functions of the tools. Each set of instructions consisted of a series of steps to be carried out.

For example,

"This task involves setting up reporting statistics for the course as a whole.

- Select the course in structural view by clicking on it. Notice that the information displayed in detail view (the right hand side of the screen) includes a frame entitled "Summary Statistics". Once at least one statistic has been set up, a table will display these. You are now going to set up a reporting statistic.
- From the Tools menu, select Summary Statistic Designer. You will notice that there are no statistics currently listed.
- Press "New" to create a new statistic.
- From the top drop-down box, select "Average Course Rating" and from the second drop-down box, select "in the last month".
- Press OK to save this. You will now notice that the statistic you just created has been added to the list.
- Press Close to close the Summary Statistic Designer.
- To see the result of creating the new statistic, select any other component in structural view and then select the course again, or minimise the window and then maximise it, to refresh the screen."

The tutorials contained a description of how to carry out the most common functions of the tools and were provided to subjects to aid in carrying out the prescribed tasks.

Once subjects were familiarised with the tools they were asked to fill out a questionnaire. The format of the survey questionnaire was designed to gather opinion of both the current state of the tools and possible future direction. As such, the questionnaire was divided into two sections. The first section concentrated on current features of the tools, requesting that a subject rate each current feature on three criteria: usefulness, user friendliness and importance. The second section focused on future features of the tools. A list of possible features that could be added to the tools was presented and subjects were again asked to rate each of these using just two criteria: perceived usefulness and importance. The ratings were then used in the following ways:

- Ratings on usefulness and user friendliness of current features, as well as average time taken to complete each task, were used to assess the usability of the current prototype.

- Ratings on importance of current features were used to determine whether it is worthwhile improving existing features, or even including them at all. Comments on low ratings given to current features were used to provide some context to any decisions made.

- Ratings on usefulness and importance of possible future features were used to formulate a plan for future implementation, with a priority given to those features that rated highly. Comments provided on particularly high ratings given to possible features were used to provide context for any decisions made.

The complete tutorials, tasks sheets and questionnaires for both tools can be found in Appendix A.

8.2.1.2 Authoring Tool Summarised Results

The current features surveyed at this stage in the authoring tool's implementation were:

1. The imposed hierarchical course structure
2. The available media from which to construct course pages – text and images
3. The properties display in the page design area of the detail view
4. The page content (layer) tree, also in the page design area of the detail view
5. The structural view
6. The detail view
7. The table view of all summary statistics and their values
8. The summary statistic graphical designer
9. The messaging inbox
10. The message composer
11. The ability to send unicast messages (reply)
12. The ability to send broadcast messages to all subscribers of a course

The features surveyed for possible future implementation in the authoring tool were:

1. A flexible course structure, rather than the current imposed hierarchical structure.
2. Existing media loading as content, such as PDF files, Microsoft Word files, RTF or plain text files.
3. A template creation facility, allowing pre-definition of both structure and aesthetics.
4. A course content printing facility, for alternative viewing.
5. A storyboard view, which would display small versions of the pages of a course in order.
6. The ability to embed sound.
7. The ability to include animation.
8. The ability to add hyperlinks from one point in a course to another.

9. The ability to add hyperlinks to an external web address, to be loaded in a separate browser window.
10. The pre-definition of all summary statistics, so that values for all statistics would be shown, removing the need and also the ability for authors to choose an individual subset of statistics.
11. The addition of a graphical display of the data behind summary statistics, such as values over time or the raw values behind an average.
12. The ability to send messages to external email addresses.
13. An auto-retrieve facility that checks for messages on a regular basis.
14. A message printing facility, for alternative viewing.

The collated results of the user survey with respect to the questions on the current features of the authoring tool are shown in Figure 8.1. The results from the questions relating to possible future features of the authoring tool are shown in Figure 8.2.

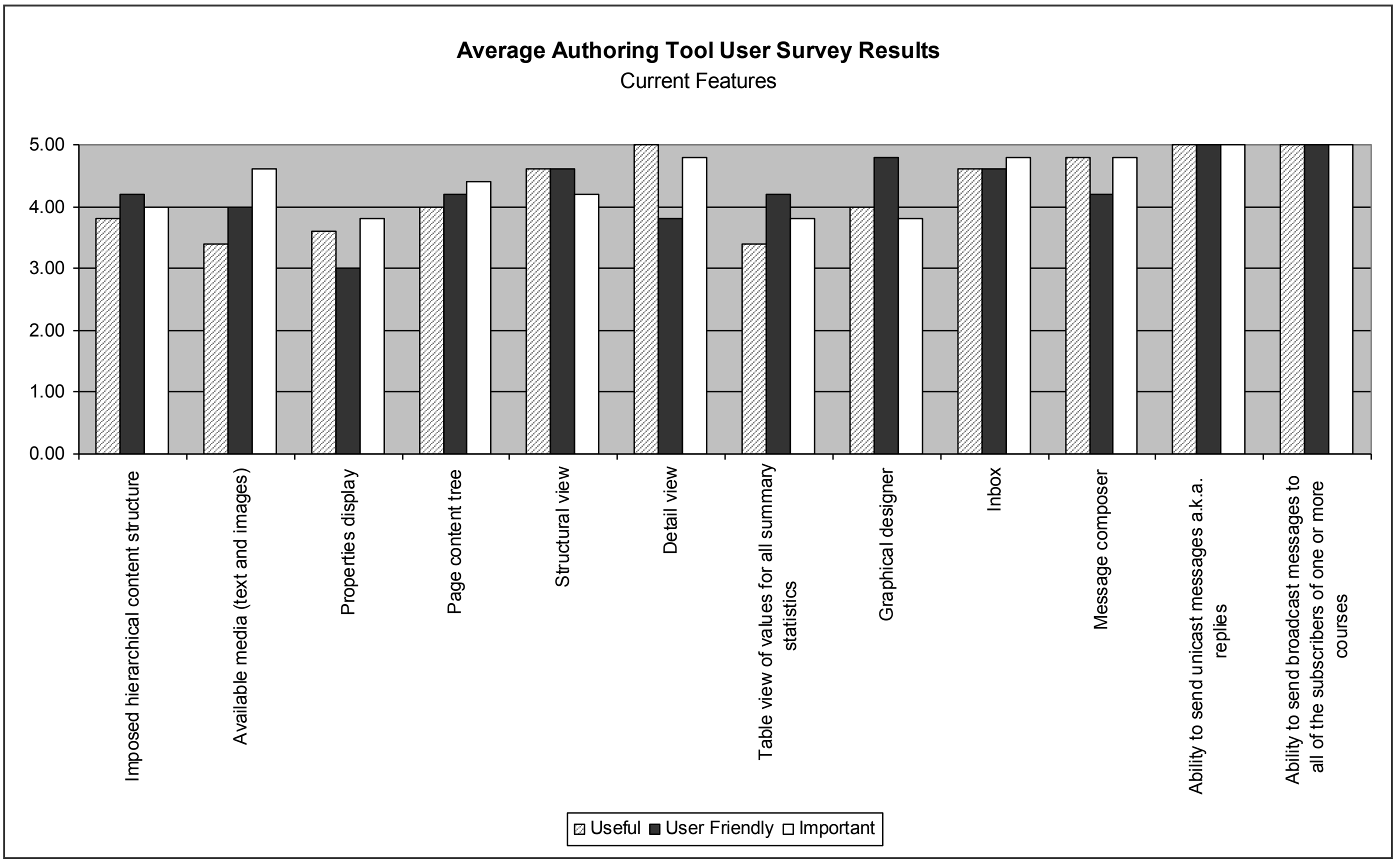


Figure 8.1: User testing results of current features implemented in authoring tool prototype

Of note in Figure 8.1 is the result for the available media authors may use to construct course content. At the time of this user survey, the only available media were text and images. Users felt that while the ability to use text and images was very important, the usefulness of just these two options was not high and the ease with which these media could be used to construct content was about average among the features of the tool. Also of particular interest in Figure 8.1 is the result of the question on the detail view of the tool. This is the display in the right hand side of the screen, which shows the detail of whatever component is selected from the course structure, such as the design of a page. While the importance and usefulness of this view are extremely high, the user-friendliness is relatively low, indicating potential to greatly improve the usability of the tool with work in this area. Some of the reasons given by users for the low usability of the detail view were the difficulty in manoeuvring images on a page, the lack of standard keyboard shortcuts for text editing functions and the difficulty of the page layers concept.

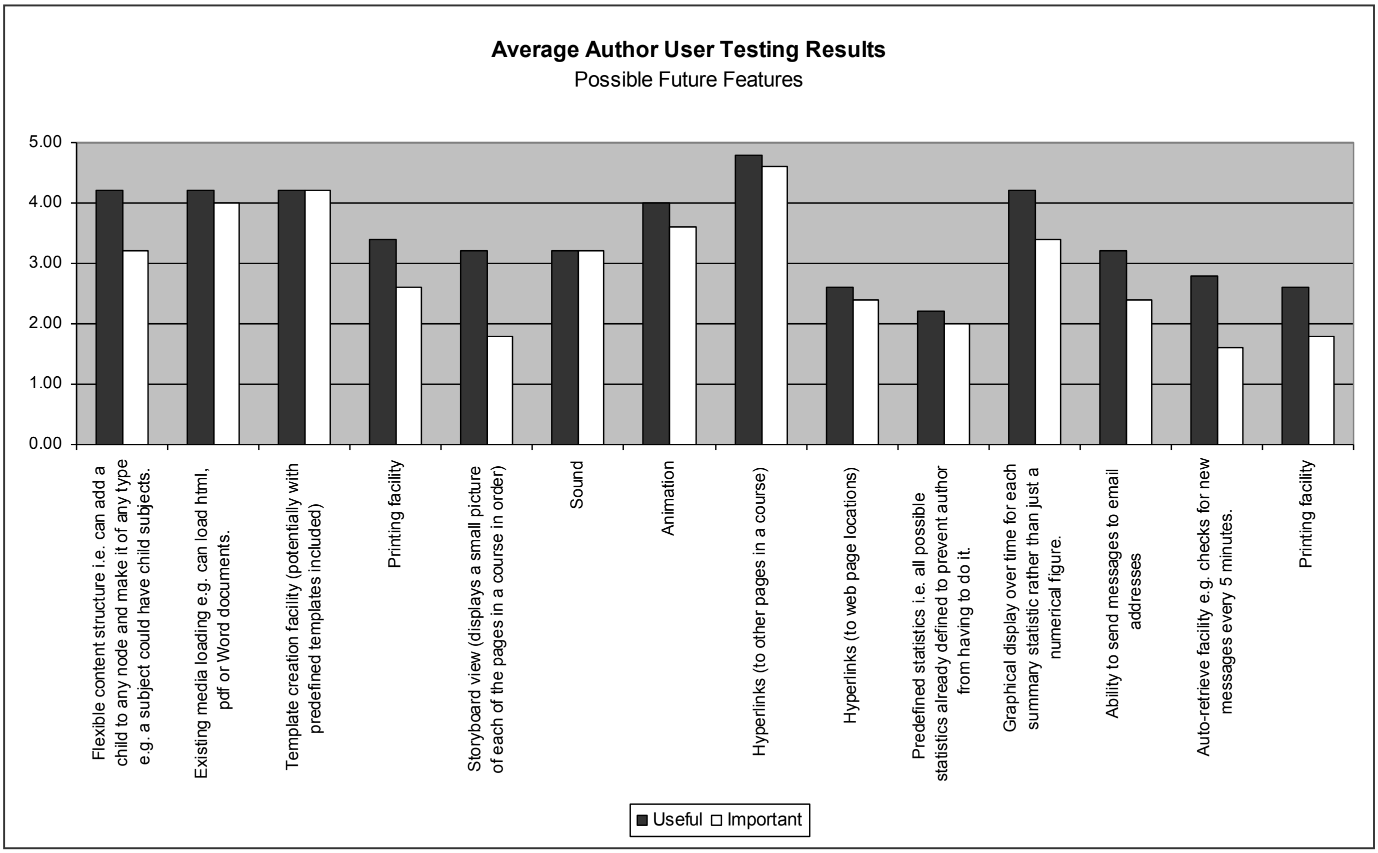


Figure 8.2: User testing results of possible future features of the authoring tool

In comparison with the results of the current imposed hierarchical structure for a training course, the allowance of a more flexible course structure, that would allow say a subject to contain other subjects or a page to be added directly to a lesson, was seen as slightly more useful but not very important to the ability of the tool to create adequate training courses. Among the possible future media suggested, the ability to load pre-existing media such as PDF documents, Microsoft Word documents or HTML as content was seen as the most useful along with the ability to create and use course templates, which could include structure and aesthetics. Of the other forms of media suggested, the ability to include animation in courses was seen as useful but not quite as important, and the ability to embed sound was seen as neither particularly useful nor important to training courses produced. As another form of navigation, users strongly felt that hyperlinks between two points in a course would be useful as well as important to the usability of training courses produced. Of the two possible future enhancements suggested for the summary statistic reporting facility, the ability to view the data behind a value such as an average or a count in a graphical form was seen as highly useful but not quite as important to the main function of the tool. The pre-definition of all possible statistics, removing the need and ability for users to set up individual sets of statistics they choose to view was seen as neither useful nor important to the tool. Users commented that they would rather have the ability to customise statistics shown as in the current implementation of the tool. Aside from the features already mentioned, the general response to future features was that they would be useful if implemented but not important to the core function of the authoring tool.

8.2.1.3 Viewing Tool Summarised Results

The current features surveyed at this stage in the viewing tool's implementation were:

1. The personalised menu of courses.
2. Hierarchical table of contents.
3. First, previous, next and last navigation buttons.
4. Text
5. Images
6. The ability to directly communicate with course authors through messaging.
7. The ability to rate and comment on courses.
8. Status bar.
9. Colour coding to show visitation progress within a course.
10. The use of text on buttons rather than icons.
11. The aesthetics of the colours used in the tool's interface.
12. The readability of text within the tool (appropriateness of font and size).

The features surveyed for possible future implementation in the viewing tool were:

1. Having the user's current location within the course displayed at all times in the form of a path.
2. The presence of hyperlinks within course content that link related sections of material.
3. The presence of hyperlinks to external web addresses.
4. The presence of links to the software application that is the focus of a training course for the purpose of demonstration and to allow users to try tasks for themselves.

5. The presence of sound within training courses.
6. The presence of animation within training courses.
7. The ability to customise the aesthetics of the viewing tool, such as colour theme.
8. A greater use of icons on button and menu functions.
9. The ability to increase or decrease the size of all fonts for readability.
10. A printing facility.
11. The ability to record personal information regarding users' current ability with the software being learned to form a profile, which would then be used to influence the level of material presented.
12. The ability to sit a test on material learned from a training course and have feedback returned.

The collated results of the user survey with respect to the questions on the current features of the viewing tool are shown in Figure 8.3. The results from the questions relating to possible future features of the viewing tool are shown in Figure 8.4.

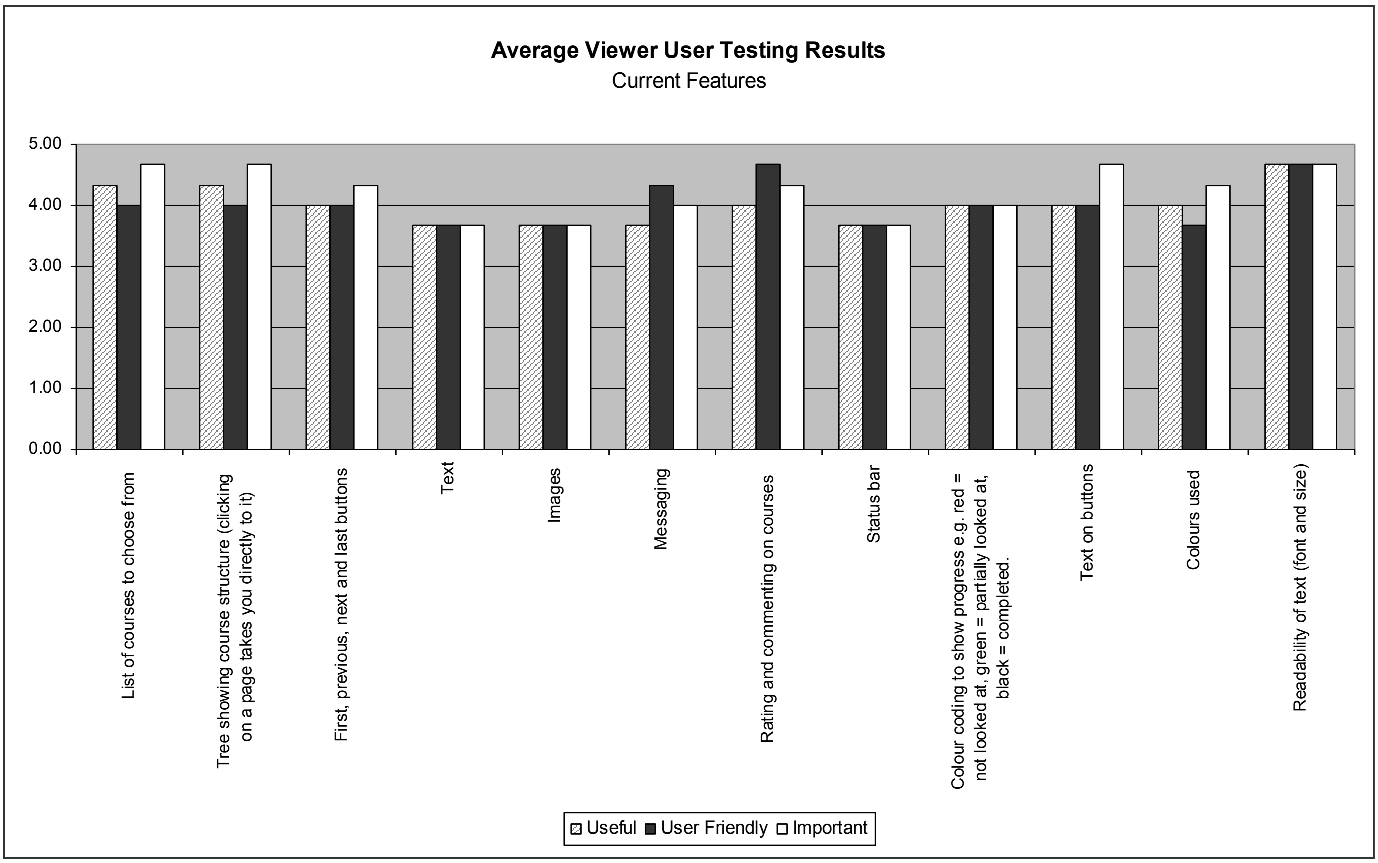


Figure 8.3: User testing results of current features implemented in viewing tool prototype

Of the current forms of navigation provided, users particularly liked the table of contents, giving it a very high importance and only slightly less high usefulness and user-friendliness. Users also appreciated the navigational buttons provided. In addition to these forms of navigation, users felt that the presence of hyperlinks with course content to link related sections of material would be very useful and important to the function of the viewing tool, allowing more explorative navigation. Also of interest was the very high rating of usefulness and above average importance given to the ability to view and interact with software demonstrations. Users also particularly liked the ability to directly communicate with training course authors through messaging, as well as to give feedback through ratings and comments. Although some users commented that they might not use the rating facility they also said that they appreciated the fact that they could because it showed an interest from the authors in their opinion of the material and gave them a greater sense of involvement and influence in the training material presented to them. Users also found the colour-coded visitation progress feedback useful and important to their effective use of the tool.

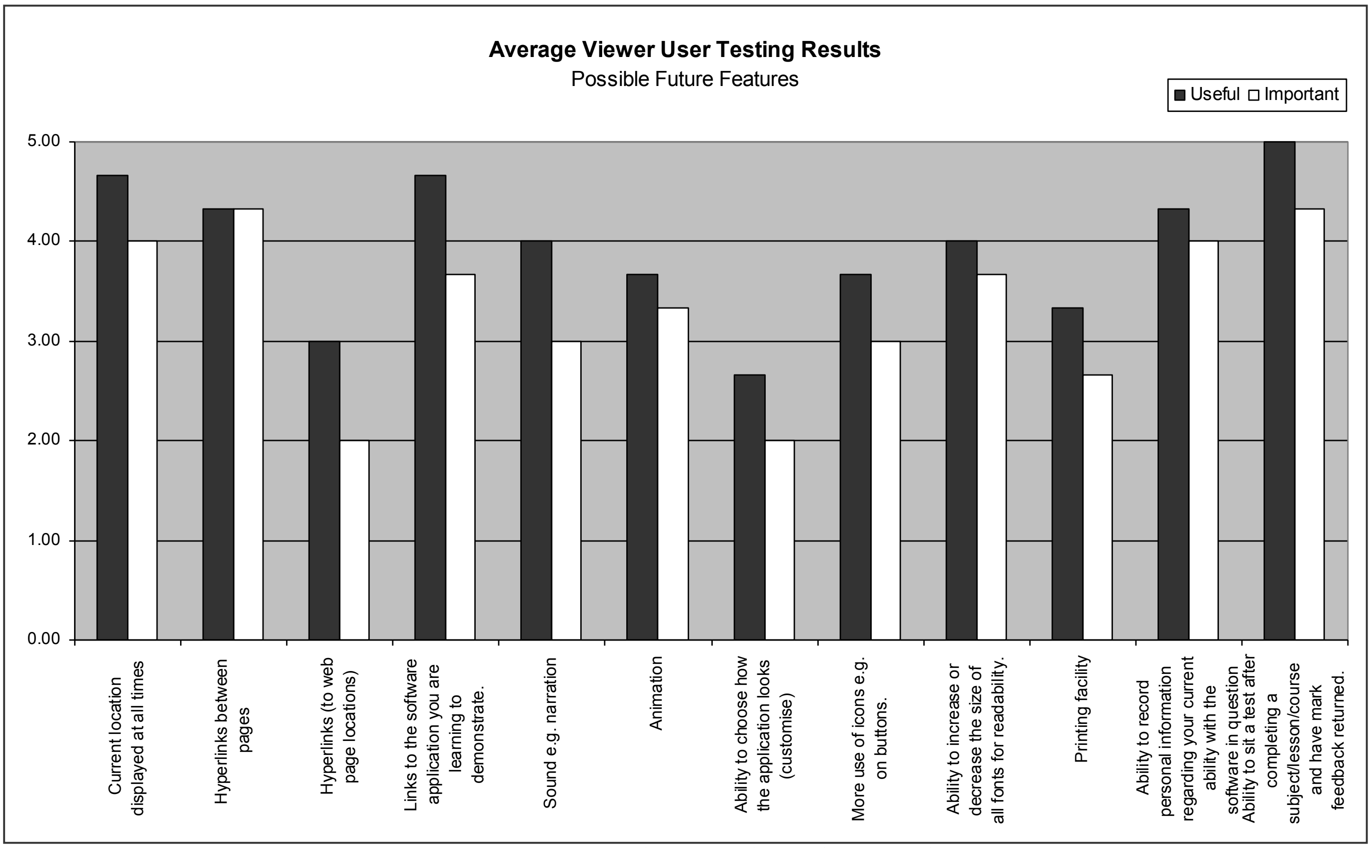


Figure 8.4: User testing results of possible future features of the viewing tool

To improve the usability of the viewing tool, most users felt that displaying the path of their current location within the course at all times, in addition to highlighting the current page in the table of contents would help them maintain their orientation within a course. To improve the learning experience, users felt strongly about two possible future features, the ability to record personal information to form a skill/knowledge profile that would influence the level of difficulty of material presented and the ability to sit a test after completing all or part of a course and have feedback returned. Profiling was seen by users as an opportunity to have greater control over the training material they had access to and felt it would increase the likelihood of finding content appropriate to their needs. The testing function was seen as very important by users to confirm their recollection and understanding of training material presented and as a positive reinforcement of the effort they had spent in learning it. We believe that a testing function would indeed provide much needed extrinsic motivation for users and assume an important part in the learning process.

8.2.2 Second Round

8.2.2.1 Overview

As mentioned above, the purpose of the second survey was to assess the suitability of the concept of the tools for authoring and distributing software training. To achieve this, the survey was carried out on site at Orion Systems Ltd with software support staff providing their experience and insight. Again, subjects were provided with a tutorial for each of the tools and given task sheets to familiarise them with the tools. As with the first questionnaire, this questionnaire asked the subjects to rate the current features of the tools but also asked more open-ended questions, aimed at determining the perceived suitability of the tools to Orion's software support environment, their willingness to use such tools and general comments about their experience using the tools.

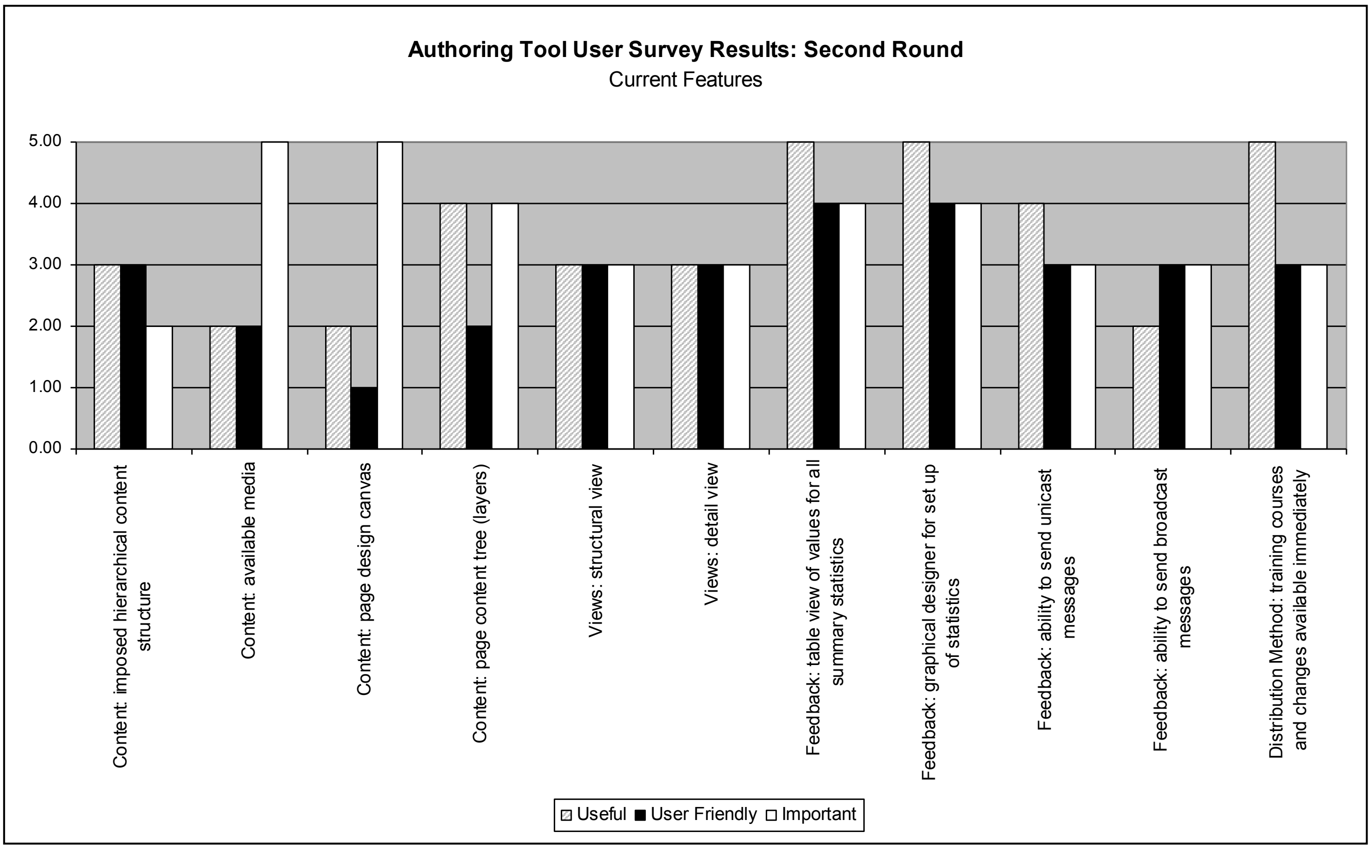


Figure 8.5: Authoring tool user survey results – second round: current features

8.2.2.2 Authoring Tool Summarised Results

The current feedback features of the authoring tool were very well received, with high ratings for their usefulness, user-friendliness in the prototype implementation and importance to the overall concept of the tool, as shown in Figure 8.5. This included the reporting of logged usage and rating information in the form of summary statistics and their graphical set up. The feedback features suggested for possible future implementation, as shown in Figure 8.6, were not rated high in terms of their usefulness or important in relation to the features already implemented, which suggests the users were pleased with the feedback options already included. It was seen as useful however to allow the export of the raw data behind the summarised statistics, to Microsoft Excel for example, for further analysis.

In terms of the distribution of training courses, users found the immediate availability of updates or changes to courses very useful. The distribution of courses is addressed more fully in the viewing tool summarised results section.

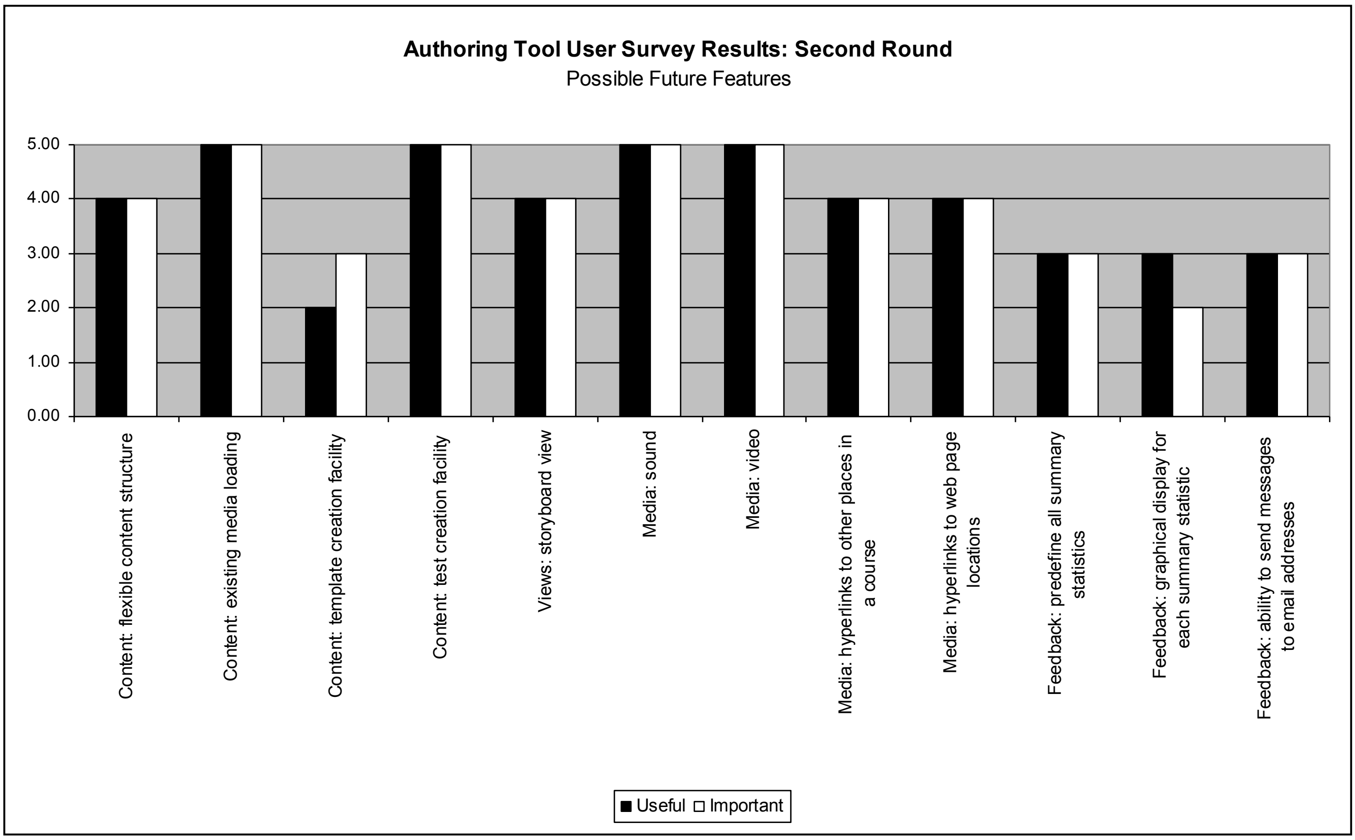


Figure 8.6: Authoring tool user survey results – second round: possible future features

Although the creation of course content was not the focus of this work, we still evaluated the current state of the content creation facilities and suggestions for its improvement in the future. Users found the content authoring functionality, including the available media, basic and somewhat restrictive, as shown by the fairly low ratings for these features in Figure 8.4. From the future features suggested to improve content creation, users felt that the addition of a facility that allowed the loading of existing media as course content and the addition of video and sound as available media would be both very useful and important. A flexible course structure was also seen as more useful and important than the imposed hierarchical course structure currently implemented in the prototype.

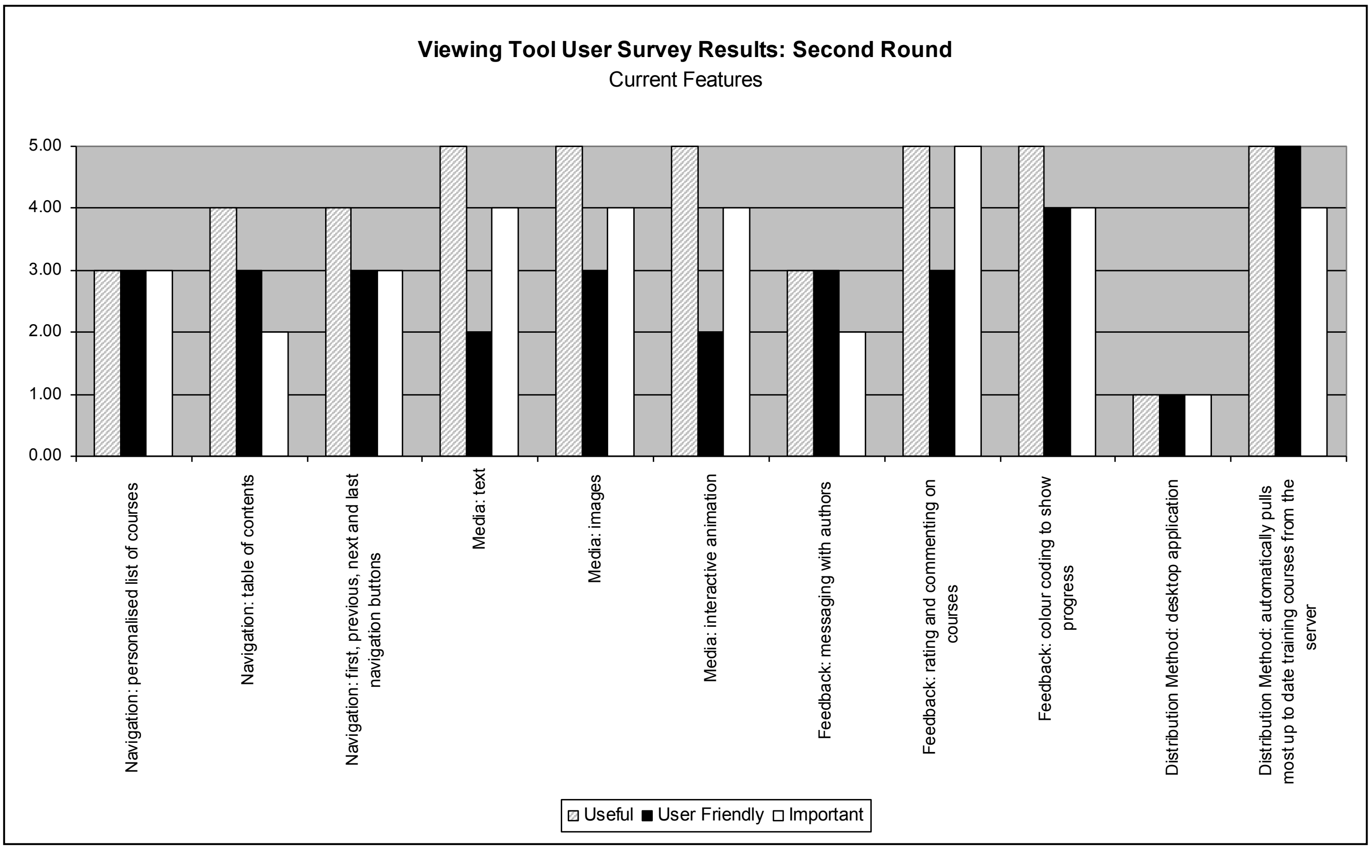


Figure 8.7: Viewing tool user survey results – second round: current features

8.2.2.3 Viewing Tool Summarised Results

Average ratings given to the navigation facilities currently provided in the viewing tool prototype, as shown in Figure 8.7. At the same time, users gave very high ratings to all of the suggested future navigation features in both usefulness and importance to course viewing, in particular hyperlinks and links to the actual software application being instructed for real time, interactive demonstrations.

The media currently viewable in the viewing tool were all rated as very useful and important, however not particularly user-friendly. This is shown in Figure 8.7. The reasons given for this are based on the training context of the material, with users feeling that more than text, images and interactive demonstrations would be required to keep their attention during training. This is reflected by the high ratings given for the future inclusion of sound for narration and video for demonstration, as shown in Figure 8.8.

In terms of the distribution method implemented, users rated the method of access to training courses through a desktop application very low in usefulness and user-friendliness and stated the reason for this was the need to access training immediately, such as from within the application they want to learn or on a website. This is reflected by the very high rating given to the suggestion of distributing the training courses through an Applet, viewable in any web browser.

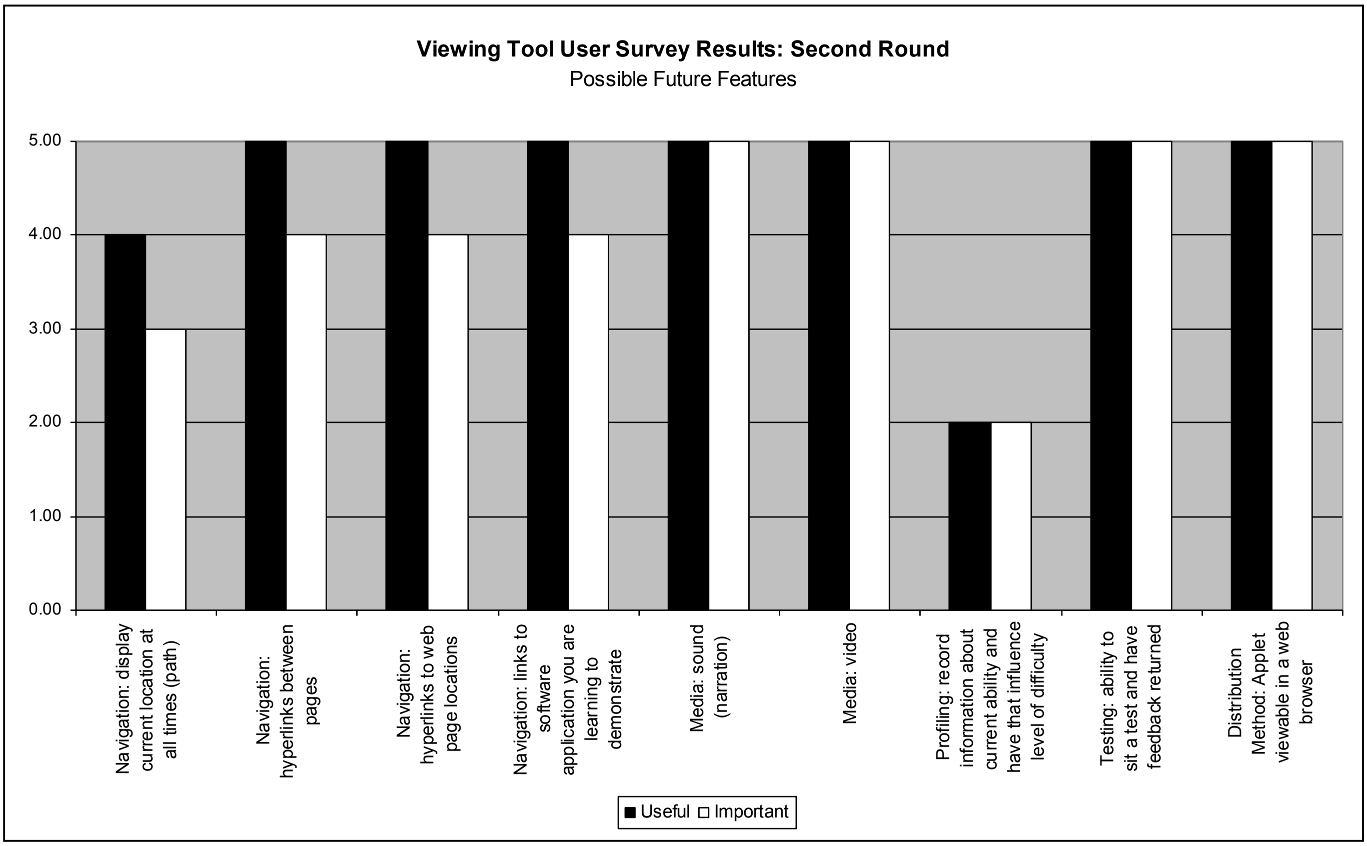


Figure 8.8: Authoring tool user survey results – second round: possible future features

8.3 Summative Evaluation

In addition to the evaluation of the system based on a user survey, we feel a summative self-evaluation of the system is appropriate. The reason for this is to provide a holistic evaluation of the system from the point of view of knowing the overall goals of the system and the main issues being investigated through the implementation of the prototype. The study of the evaluation of educational software has been around for as long as educational software itself. Currently, the context of the evaluation is under debate. It is the opinion of [Squires et al 99], [Squires et al

96] and [Reeves 92] that a meaningful evaluation can only be carried out when educational software is considered in the context of the intended learning environment and not merely as an object in its own right. This line of thinking certainly applies to training courses created using the authoring tool when viewed in the viewing tool, as this situation is consistent with the definition of 'educational software' in the previously referred to literature. However, it does not apply to the authoring tool itself, as this is more consistent with electronic publishing software. For this reason, the viewing and authoring tools are evaluated separately.

A review of methods of evaluation of educational software suggests several approaches that could be applied to evaluate our prototype system. These include evaluation frameworks, checklists and heuristics. For the evaluation of both the viewing and authoring tools, we have chosen a heuristics-based approach. The viewing tool, as it is used to view the complete training course presented in the chapter entitled 'Case Study' is evaluated using a set of heuristics suggested by [Squires 99], which take into consideration the combination of learning and usability. [Squires 99] points out that the heuristics are not comprehensive, due to the infancy of this style of heuristic, however they are suitable for the purpose of evaluating our prototype. The authoring tool is evaluated using Nielsen's usability heuristics [Nielsen 94], which may be used to comprehensively evaluate the usability of any software application.

8.3.1 Integrated Learning and Usability

The following section presents the self-evaluation of the viewing tool as it is used to view the training course presented in the chapter entitled 'Case Study' using a set of heuristics that evaluate both the learning and usability of educational software.

Is the complexity of the multimedia appropriate?
This heuristic evaluates an application on the appropriate use of complex multimedia, possibly including text, graphics, video and sound, for the purpose of enhancing and supporting conceptual development.

The viewing tool is capable of displaying text, graphics and animation. While any images may be turned into an animation, the purpose of implementing animation functionality was to create accurate replicas of software situations so that users may learn by interacting with the software in a safe, guided manner. Since this is the recommended use of the animation facility, we feel that this is an appropriate use of this type of multimedia. Lacking from the viewing tool prototype is the ability to play sound embedded in training courses. The inclusion of this facility for the purpose of creating more accurate software demonstrations would certainly be an appropriate use of sound and should be considered for future work.

Is the learner active?

This heuristic is concerned with users' development of a sense of involvement and ownership through actively controlling the learning environment, rather than becoming absorbed in it, as can often be the case with multimedia environments.

The most interactive part of a training course displayed in the viewing tool is the software demonstration, referred to as an animation. Users interact with these software demonstrations in the same way they would interact with a real version of the software they are learning. Users are given instructions and it is then up to them to complete the task set for them by carrying out such primitive actions as clicking a mouse button, pressing a key or typing a string of characters. The demonstration responds according to the appropriateness of the user's actions. It is also possible for a user to have the task automatically demonstrated for them if they do not feel confident enough to try it themselves. This software demonstration capability was seen as very important to the learning process supported by the viewing tool, and to actively involving users in learning, rather than limiting methods of learning to comprehension through reading and memory.

Is fantasy used in an appropriate way?

This heuristic evaluates educational software on the appropriate use of fantastically realistic visual environments or virtual reality. The viewing tool is capable of displaying accurate replicas of software situations but does not extend this to the representation of a three dimensional environment that might absorb the user and detract from the task or concept being taught.

How appropriate is the content to the curriculum?

Since the content of the training course outlined in the 'Case Study' chapter was extracted from pre-existing training material in a different form, the evaluation of this content is not appropriate for the overall evaluation of the viewing tool. However, the animations have been added to the pre-existing content. Since these animations are based on instructions for tasks provided in the pre-existing training material, we feel they are entirely appropriate and that they enhance the written explanations and instructions given by offering a dimension to the course that actively encourages learner participation.

How navigable is the software?

The ability to quickly and easily find content and move around within a training course was seen as one of the most important facilities of the viewing tool. To accomplish this, navigation has been kept as simple as possible and multiple methods for finding content have been provided. When a user initially opens the viewing tool, they are presented with a personalised list of training courses available, from which they may select a course and open it. From this point, there are two methods of navigation available. The first allows linear navigation through the course's pages from start to finish using a set of buttons consisting of 'first', 'previous', 'next' and 'last'. The second method uses the table of contents visible in the left hand side of the viewing tool. This shows the entire contents of the course, with every item listed an active link that displays the corresponding page in the display area on the right hand side of the tool. This method enables users to search the table of contents for a specific topic or task they wish to learn and skip directly to the corresponding part of the course. Users can switch from using on form of navigation to the other at any time. For example, once a desired subject has been located in the table of contents such as 'How to define messages' and the user clicks on this subject the first page would be displayed. From this point the user may wish to view the entire subject linearly, so would use the 'next' button to move from page to page until the end of the subject. The navigation of the viewing tool could possibly be improved by the addition of a 'back' function, similar to that found in common web browsers. This function would allow users to move to previous material viewed without having to recall where that material was located, which would be particularly useful as training course size became large.

What form of learner feedback is provided?

Users are provided with feedback of their visitation progress through training courses with the use of colour. Rather than simply showing visited and not visited, as with a common HTML hyperlink, the viewing tool makes a distinction between those parts of a course that have been completely visited, partially visited and not visited at all. While the smallest element of a course, a page, cannot be partially visited, subjects, lessons and the course itself are partially visited if a user has been to at least one but not all of the pages within them. This facility keeps users always informed of those parts of a course they have already seen, to prevent unnecessary repeat visits, and those parts of a course that have yet to be seen. This is particularly important if the user is navigating using the table of contents rather than in a linear fashion, as it may not be obvious when a course has been completed.

Another form of feedback provided to learners is given during interaction with software demonstrations. Feedback is provided on each action a user takes. If an action is appropriate the viewing tool will inform the user with a positive statement and progress the demonstration to the next stage. If a user action is inappropriate the user is presented with a negative but encouraging statement and allowed to attempt the action again. Care was taken to provide more attractive and rewarding feedback for a correct action than an incorrect action, to prevent motivational confusion. While the conceptual design of the viewing tool recommended responding to incorrect user actions by giving the user a choice between trying again, receiving a hint and trying again and being shown how to perform the action by the tool, this was not implemented in the viewing tool prototype. We feel the usability of the software demonstration function suffers because of this difference.

What is the level of learner control?

This heuristic is concerned with providing an adequate level of user control over the learning environment to enable users to develop a sense of involvement and ownership.

A fairly low level of user control has been allowed for in the viewing tool prototype. While users can choose their own path through the material presented they cannot choose what is presented to them, nor can they personalise the look and feel or colour theme of the tool. The conceptual design allowed for the specification of a level of learning, such as novice, intermediate or advanced, through profiling of individual user abilities. This was not implemented in the viewing tool prototype but we feel would be a very good way to provide greater learner control within the tool.

Another way have discovered to increase learner control within the viewing tool through the user survey is to change the way the software demonstration function works to allow for exploration. For example, the user could be presented with an image of the user interface as seen in the real version of the software they are learning. The user could then choose to what to explore by moving to a particular part of the interface or by selecting a particular function. The demonstration could then branch off to a demonstration focused on that part of the software and return to the main user interface image upon completion, from which the user could select another area to focus on. This would not require much of a change to the existing prototype as it could function as a collection of software demonstrations as they work now, collected together and made accessible by an initial image of the software's user interface.

Are learners motivated when they use the software?

The viewing tool prototype provides mainly intrinsic motivation. Intrinsic motivation is experienced through completion of the interactive software demonstrations, helping learners to complete a task just as they would in the real version of the software. Extrinsic motivation, where learners are provided with a reward for learning, was recommended in the conceptual design through testing. By placing a test at the end of a logical unit of learning, such as a lesson or the entire course, users would have an opportunity to recall the information they viewed throughout the course to test their memory and to apply this information to construct knowledge. Both forms of motivation were included in the design of the tool so as not to limit the choices possible for authors when designing training courses. The testing function was not implemented in the viewing tool prototype, which we feel lessens its teaching capability since learners do not have the opportunity to recall the information they have viewed.

8.3.2 Usability

The following section discusses the usability of the authoring tool based on Nielsen's usability heuristics [Nielsen 94]. While usability is only a small consideration at this stage, since the tool is a proof of concept prototype, a usability evaluation is still a useful indicator of the current usability of the tool and of future work that may improve its usability.

Visibility of system status

This heuristic looks at the way an application keeps users informed of what is going on, to see if it provides appropriate and helpful feedback to users in a reasonable amount of time.

The authoring tool provides a status bar that is always present and visible at the bottom of the main screen. Initially the status bar reads "Ready" to indicate it is not currently carrying out any tasks and is ready for user input. The status bar is used to inform users of all tasks that take control of the application away from the user, in particular all requests to the server since these may take some time. For example, when a user chooses to browse the image catalogue for the first time since logging in, the client will request from the server all images currently held in the database. During this process, that status bar will read "Retrieving available mages…" until it is completed when it reverts back to "Ready".

The first time a user selects a page from the course structure tree, which has a visual representation to be displayed, the application takes on average about two seconds to load and display the page, due to the need to construct the page design area. This delay assumes control of the application and because the user interface cannot be updated until the delay is over, the status of the system cannot be reported in the status bar. Ideally, the status bar should read "Loading page…" or a progress bar should appear in a dialog to show how the loading of the page is progressing. For every subsequent page that is selected during the session, there is no delay.

The user is also implicitly informed of the status of persistency of the course they are currently working on y the activation and deactivation of the 'save' function. Initially, the save function is not activated. After a user makes a change that is not cancelled or undone, the save function is activated to show the user that their view of the course is not consistent with the persisted version. Once the user carries out the save function, it is again deactivated.

Since the presence of an active connection between an author client and the server is very important to the operation of the authoring tool, the user is immediately informed of a cease in operation of the server through the appearance of a dialog box. The user may choose to work offline or close the authoring tool at this point.

Match between the system and the real world

This heuristic evaluates an application's ability to speak the user's language, by using terms and concepts that the user will be familiar with rather than system defined concepts, which are unlikely to be provided at a suitable level of abstraction, as well as following real world conventions.

When designing the allowable structures for a training course, we felt that the terminology used to describe course elements should match that used in paper-based training courses or in-person training courses. The reason for this was to take advantage of any end-user familiarity with previous training courses or general learning they have experienced in these forms. To this end, a complete training course is represented by the term 'course', which trains end-users how to use a single piece of software. Each course consists of 'lessons', which teach for example a functional section of the software such as would be used to complete an important task, or perhaps a physical section of the software such as a particular part of the user interface. Each lesson is then made up of 'subjects', which focus on a small part of a lesson, such as one step in carrying out a complex task or a simple concept. Finally, each subject consists of one or more 'pages'. This is where the visual representation of the training course is stored, just as it is in a paper-based training manual, where courses, lessons and subjects are really just used to organise and contain pages.

User control and freedom

This heuristic evaluates an application on its provision of clearly marked exits from undesirable system states. Since users often choose functions by mistake, they should always be provided with exits at every stage of a function and should also be provided with undo and redo capabilities.

In an attempt to provide sufficient user control and freedom, a 'cancel' function has been provided at every step in every dialog within the authoring tool. This allows users to change their minds about continuing with a function and reverts the system back to the state just prior to the appearance of the dialog. In addition to this, users are prompted for confirmation before carrying out any action that may cause part of a course design to be lost, such as deleting any part of a course or exiting without saving changes. This was included in an effort to prevent irreversible loss of work. The ability to exit an application without saving changes can also be useful should a user not wish to save the effect of carrying out an undesirable function. Undo and redo functionality has been implemented for text editing only at this stage. The extension of this functionality over the entire application would definitely enhance its usability and should be considered for future work.

Consistency and standards

This heuristic evaluates an application on its adherence to standards and general consistency, such as following platform conventions.

The use of the Java programming language for implementation of the authoring tool, which is supposed to be platform independent, removes the need to adhere to any particular platform conventions. However, the user interface of the tool strictly adheres to Java Look and Feel standards, including the names of commonly used functions and iconic graphics used to represent functions on toolbar buttons. The lack of commonly accepted shortcuts for common functions, such s ctrl-c for copy and ctrl-v for paste does detract from the consistency of the tool, as it could easily be off-putting for a user to attempt to use such a shortcut and find the system does not respond in the way they expect.

Error prevention

Error prevention is concerned with preventing user errors from occurring rather than allowing errors and then informing users that an error has occurred. This can be done by disabling functions when not appropriate or by hiding information that is not relevant to the situation.

The authoring tool has made an attempt to prevent user errors in the majority of its functionality. The practice of disabling irrelevant functions or functions that are not available for any reason to prevent them from being chosen from menus or otherwise has been widely used. A disabled function is still visible but may not be selected and is represented visually as a greyed-out button or menu item. For example, the save function mentioned above is only enabled when changes have been made to prevent users from carrying out the function, which may cause some delay to the application, when it is unnecessary. Also, several functions that rely on a selection from a list such as ‘open’ used to open courses, ‘delete’ used to delete summary statistics and ‘update’ used to update rating criteria, are only enabled when an item is selected from the list. In addition to the enabling and disabling of functions, some functions are made completely invisible when they are not appropriate to prevent them from being selected. This is the case with the popup menus available when nodes are selected from the course structure tree. The set of functions available when working with the structure of a course includes adding a lesson, adding a subject, adding a page, adding a rating page, and deleting all of these components. However, only a subset of these functions appears in the popup menu for each type of node in the tree. For example, the popup menu of a course node includes ‘add lesson’, ‘add rating page’ and ‘delete course’ but does not include ‘add subject’ because it would violate the allowable structure to add a subject directly to a course.

Recognition rather than recall

This heuristic considers the visibility of system objects, actions and options. Based on the idea that an application is less taxing on user memory and therefore more usable if information can easily be looked up rather than having to be remembered, users should not have to remember information that is relevant to what they are currently doing, but should be able to see the information or have fast, easy access to it. An example of this is making visible information entered in a previous step of a dialog. Instructions on how to use the application should also be readily available.

The authoring tool does allow for the idea of recognition rather than recall in its implementation but has not been implemented in a way that fully supports it or takes advantage of its benefits. An example of this is the visibility of information past information. When working in a multiple step dialog where previous information entered is relevant to current decisions, such as the dialog used to set up summary statistics, the user may view the previous step at the same time as the current step by moving the current step's window alongside that of the previous step. If a previous step is relevant to the current step, it is not completely hidden from view but sent behind the current step so that the information it contains may be viewed. A previous step cannot be edited however, unless the current step is closed. A better design would have relevant past information displayed in a non-editable form on the current step in the dialog, allowing users to view it without having to look at the more than one window at once. Also, at this stage there is no searchable help or documentation within the authoring tool to allow users to look up instructions on its use.

Flexibility and efficiency of use

This heuristic focuses on the presence of accelerators – shortcuts both pre-existing in the application and able to be created by users to increase the efficiency of using the application by experienced users.

The usability of the authoring tool is lessened by the omission of any accelerators or shortcuts, as well as a facility that allows users to customise their own shortcuts. The inclusion of these would definitely improve usability, especially for experienced users, and is recommended for any future work on this prototype or a future version of the system. For example, shortcuts to menu items such as ctrl-s for 'save', ctrl-c for 'copy' and ctrl-v for 'paste'. Also the use of defaults is recommended, for example having default values entered in all user input user interface components to possibly reduce user input required.

Aesthetic and minimalist design

This heuristic evaluates an application on what information is presented to users and how it is presented. Only information relevant to a user's current task should be displayed in the area they are working and should be presented in a clear, easily understood manner.

Because of the large amount of information that can be required when working with electronic media and the unlimited potential size of training courses, it was very important when designing the authoring tool to consider what information to display to users, how to display it to them and when it was appropriate to display it. To this end, the main screen of the authoring tool was designed as a split screen with course structure displayed on the left and detail displayed on the right. At any time, the detail section of the window only contains information relevant to the component selected from the course structure. For example, when the course itself is selected, the detail view shows all of the properties of the course as well as functions for updating these properties. When a page is selected however, the detail view shows the page design area, which includes the design of the page itself, as it would be seen in the viewing tool and a separate representation of the layered structure of the page. The page design area also reflects the need to display only relevant information. The layered structure of the page is displayed as a tree, with every media component of every layer listed in the tree. When one of these components is selected, for example an image, the image's properties are displayed alongside the layer tree. Should no component be selected, the properties of the page being worked on are displayed.

Help users recognise, diagnose and recover from errors

This heuristic is concerned with the error messages presented to users. Good error messages should appear only at appropriate times, should be expressed in plain language at a level users will be able to understand, should precisely state the problem and suggest at least one constructive solution.

Although efforts have been made to prevent users errors, as detailed earlier, errors will still occur. The authoring tool's error messages have been designed with a focus on what the user was doing when the error occurred. An example of such an error message is that which appears when a user attempts to view and animation that has not been completely specified. If the starting image of the animation has not been set the error message will read "The animation cannot be displayed because the base image has not been set. Please set the base image and try again." This style of error message has been used throughout the authoring tool.

Help and documentation

This heuristic evaluates an application on the presence of suitable help and system documentation, which is easy to search, task-oriented and of a suitable size.

At this stage, the authoring tool contains no searchable help or documentation to instruct users on its proper use. During user testing of the prototype, subjects were provided with a tutorial to help them carry out the tasks asked of them. The absence of embedded help certainly detracts from the tools usability and is recommended for any further work on this prototype of on a future version of this system.

8.4 Summary

In this chapter, we presented the summative self-evaluation carried out on the authoring and viewing prototype tools. We used a different heuristic approach for the evaluation of each tool. The viewing tool's learning potential and usability were evaluated in the context of a learner viewing the training course described in the ‘Case Study’ chapter within the viewing tool. The authoring tool was evaluated solely on its usability. We also included the summarised results and analysis of two user surveys we conducted at different stages of tool implementation.

In summary, the strengths of the approach implemented in the prototype system, as shown by the user surveys and heuristic evaluation, include the following:

- *Distribution method*
    - The viewing tool automatically pulling the most up to date training material from the server. This is done only when necessary, which is when a learner logs into the system.
    - Changes made to training material immediately available to learners.
- *Feedback*
    - An author being able to specify the information they want reported to them on a course-by-course basis. This is also very easy to set up.
    - The presence of a direct communication channel, allowing authors and learners to communicate one a one-to-one basis within the authoring and viewing environment.

The perceived weaknesses of the prototype system include:

- *Content Design*
    - Limited media available as course content.
    - Inability to load existing media as course content, such as video or sound.
- *Distribution Method*
    - Training courses must be accessed through a separate application, rather than being available within the software a learner wishes to learn about.
- *Feedback*
    - Inability to design tests and have these available for learners to carry out.
- *Navigation*
    - Lack of hyperlinks between related content, to allow a less structured form of navigation.

Chapter 9 Conclusion

9.1 Introduction

This chapter concludes the work presented in this thesis. It presents a summary of the key contributions of this thesis to the research area. It then outlines the areas we see as having the most promise for future work and finally presents a general summary of the thesis.

9.2 Contributions

- *A continuous service approach to software training provision.* This approach is a move away from the single delivery of a paper or electronic user manual with the distribution of software, toward the provision of a tool available on a user's desktop, which is capable of downloading dynamic, up to date training material at any time. This could also be achieved in a similar way by replacing the Java 2 based graphical user interface of the viewing tool with an Applet, as discussed in the future work section below, which would be just as capable of displaying dynamic material.

- *An integrated environment for software training authoring and delivery, aimed at non-authoring specialists, such as may be found in a software development environment.* This was developed in order to prove the concept of the continuous service approach and an integrated authoring and delivery environment that supported this through encouraging iteration. The implemented environment included a comprehensive software training authoring tool, with the ability to create and maintain interactive, demonstrative training courses, a viewing tool capable of displaying the training courses produced with the authoring tool, and a training course design store all running upon a client-server based communications infrastructure.

- *The incorporation of reciprocal feedback between learners and authors.* Reciprocal feedback is an important part of any iterative process and as such it was important to include this in the system prototype. Feedback from learners to authors was gathered both implicitly and explicitly. Implicitly, learners usage was monitored and logged in order to be available for reporting to authors should they choose to view this information. Explicitly, learners could provide ratings and comments for training courses they viewed. Authors could then use any of this information to make appropriate changes or additions to training courses, which are then fed back to learners. In addition to this, direct communication through messages between learners and authors was supported. Learners may send a unicast message directly to the author of a course if they have a specific query and authors may send unicast replies as well as broadcast messages should they wish to communicate with all users of a course. The hope was that through iteration supported by reciprocal feedback, both the quality of service received by learners and the quality, relevance and comprehensiveness of the training courses produced could be improved.

- *Information on the value of a service-oriented model for software training provision for both training authors and end users.* This information could only be ascertained through the implementation and evaluation of a software training development environment that supported a service-oriented model. The user survey component of our evaluation of the authoring tool prototype showed a general acceptance of the concept of the system and an appreciation of the ease with which completed training courses could be distributed to end users, by simply making a finished training course available for its intended user group which would then automatically download the course upon running the viewing tool. There was also interest in the ability to viewing summarised statistical data on learner usage and ratings given to courses. From the evaluation of the viewing tool prototype, users showed a positive reception for the ability to view software demonstrations and interact with these to get hands on experience in a safe, guided environment.

9.3 Future Work

The future work identified through the user survey component of the evaluation as having the most potential to improve the usability and usefulness of the authoring and viewing tools are outlined below. We believe this work would need to be included in a commercialisation of this concept, to provide a complete learning process within the viewing tool, as well as an effective, usable multimedia-authoring tool.

- *Incorporating the ability to design and carry out learner testing.* As discussed in the system design, the ability to author a test and have this available for learners to test their knowledge of material they have viewed would be an important aspect of the system. The design specified the creation of a test from one or more questions, either multi-choice, true/false or performance questions, where a learner would carry out a task and be assessed on the correctness of their actions. The implementation of this would provide much needed extrinsic motivation, and support a more complete learning process by giving learners the opportunity to recall information they have read or viewed and apply this information to form knowledge.

- *Incorporating the ability to specify user profiles within the viewing tool and have this influence the material presented*. This could include the difficulty of tasks, examples and questions given in the training courses distributed by the server. This would also require the ability to specify different levels of material within the authoring tool and possibly even to specify what the appropriate levels of difficulty should be, on an individual course basis.

- *Enable existing media loading as training course content.* The benefit of this for authors could be twofold. First, existing training material could be transferred into training courses, removing the need to re-enter information and second, supporting more forms of media could allow a richer and more varied learning experienced. For example, much training material already exists for the software the training course developed in the case study was based on, including an audio visual demonstration of using the software, which would have been ideal to include in such a training course. Formats of media we believe would be most useful to be able to import include MPEG, .wav files for audio, PDF documents, Microsoft Word documents, RTF, HTML and plain text.

- *Incorporating a template creation facility.* We believe the addition of a template creation facility could greatly improve the usability of the authoring tool by shortening the time taken to build a course and by enabling a consistent look and feel across multiple related courses. We believe there is also potential for supporting different learning strategies through providing pre-defined templates. This may provide the necessary guidance for non-specialist authors to construct more appropriate training, based on specific learning requirements.

- *Have the viewing tool interact with real versions of software for demonstrations*. By integrating the viewing tool with the event hierarchy of real software applications, it would be possible for the tool to drive situations within this software for the purposes of demonstration. This could provide for much more complex demonstrations allowing more varied and accurate user interaction than are currently capable with the completely internal 'animation' facility. Demonstrations would also likely be easier to author than with the existing animation facility.

Although the viewing tool prototype implemented contained a Java 2 based graphical user interface and ran as a separate application on a user's desktop, this is certainly not the only option available and suitable for displaying training courses to end users. We believe the following options for future work on displaying the training courses hold much potential and would be interesting to investigate.

- *Integrate/embed viewing tool as a component within existing software.* By integrating the viewing tool within the software it is teaching, the tool could become an internal trainer, accessible within help for example, that would then be readily available when needed by a user and would still connect to the server as soon as it is opened to deliver the most up to date material. The viewing tool could then be distributed with the software and would be responsible for downloading the most up to date material, rather than containing static material as is common in internal application help systems.

- *Convert the graphical user interface of the currently Java 2 based viewing tool into an Applet.* This would enable delivery of training courses using the Internet and common web browsers, allowing for much greater reach and ease of distribution, since a custom viewing application would not have to be installed at remote client sites.

Before the commercialisation of any of the concepts developed in the prototype implementation, more data on the value of the tools would be very useful in determining the applicability and acceptance of the concepts and any necessary usability improvements.

- *Testing the tools on client sites.* The best way to gather this information would be to carry out user acceptance testing within a software development environment and at a substantial number of real client sites simultaneously. Users in the software development environment could produce a training course and make this available for clients at sites where that software is used. A study then over time of course usage, viewing the reporting of this information, communication habits and updates or changes made to the training course would provide very useful information on the potential for commercialisation.

9.4 General Summary

This thesis has investigated the main issues in the subject area of the provision of software training related to our focus on an integrated authoring, delivery and monitoring environment that aims to serve dynamic training material. We began with a literature review of previous and current research related to this subject area. From this we identified the issues we would most like to investigate through this work, and those that were outside the scope of this work, and developed high-level requirements for an integrated authoring, delivery and monitoring system. We then developed a complete specification and design for such a system, including conceptual object models, suitable system architecture, database and user interface design. Next we described the implementation of the system, focusing on the authoring and viewing tools, including a discussion of implementation decisions made and how these positively and negatively affected the end result. We implemented following as closely as possible the conceptual design presented, however some deviations were necessary and these were also outlined, including the reasons for them. We then presented a comprehensive evaluation of the tools, including both the results of a user survey and a summative self-evaluation. Finally we concluded the key contributions of this thesis and suggested areas we feel hold the most potential for future work on the ideas presented.

Bibliography

[Brewerton 2001] David Brewerton, “Symphonia 3 Training Manual”, Orion Systems New Zealand Limited.

[Buck et al 1995] Buck, G.; Hunka, S. *Development of the IBM 1500 computer-assisted instructional system*, IEEE Annals of the History of Computing, Volume: 17 Issue: 1, Spring 1995 Pages 19 –31.

[Demetry et al 1992] Demetry, J.S.; Black, B.; Voltmer, D.; Nahvi, M.; Jones, J. *Computer-Assisted Interactive Instruction: Results from a Developmental Effort*, Frontiers in Education, 1992. Proceedings. Twenty-Second Annual Conference, Pages 662 – 667.

[Gould et al 1999] Gould, Daniel L., Simpson, Rosemary M., and van Dam, Andries. *Granularity in the Design of Interactive Illustrations*, Proceedings of ACM SIGCSE 1999.

[Hagler et al 1995] Hagler, M.; Wetzel, K.C.; Marcy, W.M. SIMPLE in practice [educational software]. Frontiers in Education Conference, 1995. Proceedings, 1995, Volume: 2, 1995 Page(s): 3b3.1 -3b3.4.

[Hagler et al 1996] Marcy, W.M.; Hagler, M.O. Implementation issues in SIMPLE learning environments, IEEE Transactions on Education, Volume: 39 Issue: 3, Aug. 1996 Page(s): 423 –429.

[Huston et al 1994] Huston, J.C.; Gillette, J.C.; Hiemcke, C.; Johnson, R.M.; Eversden, M.R.; Pletka, R.J, *Prototype educational software for integrating the engineering curriculum.* Frontiers in Education Conference, 1994. Twenty-fourth Annual Conference. Proceedings, 1994 Page(s): 633 –637.

[IBM 1966] *IBM announces computer system for education*, Educational Technology, Vol. 6, No. 9, 1966, Pages 16-19.

[Java 2 API] Java 2 Platform, Standard Edition, v 1.3, API specification, available online at, http://java.sun.com/products/jdk/1.3/docs/api/index.html

[Java 2 SDK] Java 2 SDK, Standard Edition, v 1.3, available from Sun Microsystems online at http://java.sun.com/products/jdk/1.3/

[Munro et al] *A Tool for Building Simulation-Based Learning Environments*, Short Paper, Behavioural Technology Laboratories, University of Southern California.

[Murray 1991] Murray, T. Facilitating Teacher Participation in Intelligent Computer Tutor Design: Tools and Design Methods, Ed. D. Dissertation, University of Massachusetts, Computer Science Tech. Report 91-95. [Nielsen 94] Jakob Nielsen, "Usability inspection methods", in: J. Nielsen, R.L. Mack (Eds), *Usability Inspection Methods*, John Wiley, New York, 1994, p. 30.

[Murray 1998] Murray, T. *Authoring Knowledge Based Tutors: Tools for Content, Instructional Strategy, Student Model, and Interface Design,* Journal of the Learning Sciences, 1998. Vol. 7, No.1, pp. 5-64.

[Reeves 92] T. C. Reeves, "Evaluating interactive multimedia", Educational Technology, May p. 47-52.

[Spalter et al 2000] Spalter AM, Simpson RM, Legrand M, Taichi S. *Considering a full range of teaching techniques for use in interactive educational software: a practical guide and brainstorming session*. 30th Annual Frontiers in Education Conference. Building on A Century of Progress in Engineering Education. Conference Proceedings (IEEE Cat. No.00CH37135). Stripes Publishing. Part Vol.2, 2000, pp.S1D/19-24 vol.2. Champaign, IL, USA.

[Squires 99] David Squires, "An heuristic approach to the evaluation of educational multimedia software" CAL 97, Available online http://www.dcs.ex.ac.uk/~masoud/cal-97/papers/index.htm

[Squires et al 99] David Squires, Jenny Preece, "Predicting quality in educational software: Evaluating for learning, usability and the synergy between them", Interacting with Computer 11 (1999) p. 467-483.

[Symphonia Mapper 2001] "Symphonia Mapper User Guide", Orion Systems New Zealand Limited.

[Symphonia Mapper Guide] “Symphonia Mapper Getting Started Guide”, Orion Systems New Zealand Limited,

[Symphonia User Guide 2001] “Symphonia Toolkit User Guide: EDI”, Reference Manual, Orion Systems New Zealand Limited.

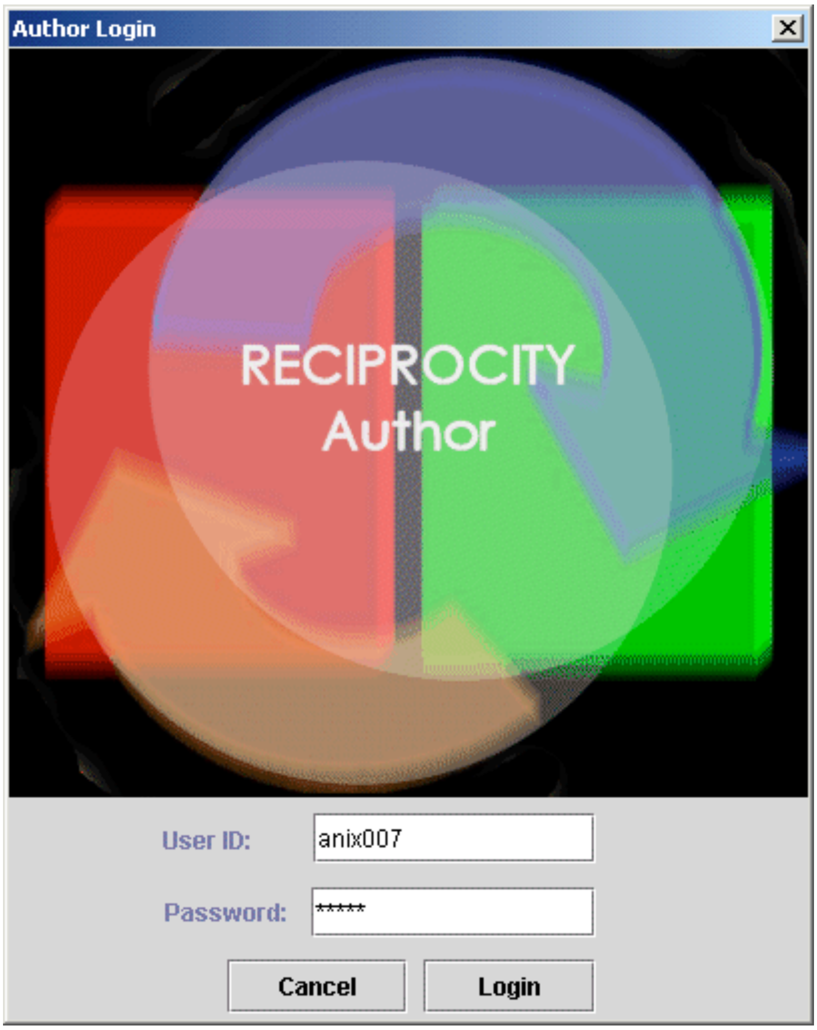
[van Dam 1999] Andries van Dam, *Education: the Unfinished Revolution,* ACM Computing Surveys (CSUR) December 1999.

Appendix A Tool Tutorials and Evaluation Material

Reciprocity Author Tutorial

Login

This first screen you will be presented with when you open the Author application is a login screen. To log in, simply enter your user ID and password and press “Login’. If you decide not to log in, pressing “Cancel” will close the login screen.



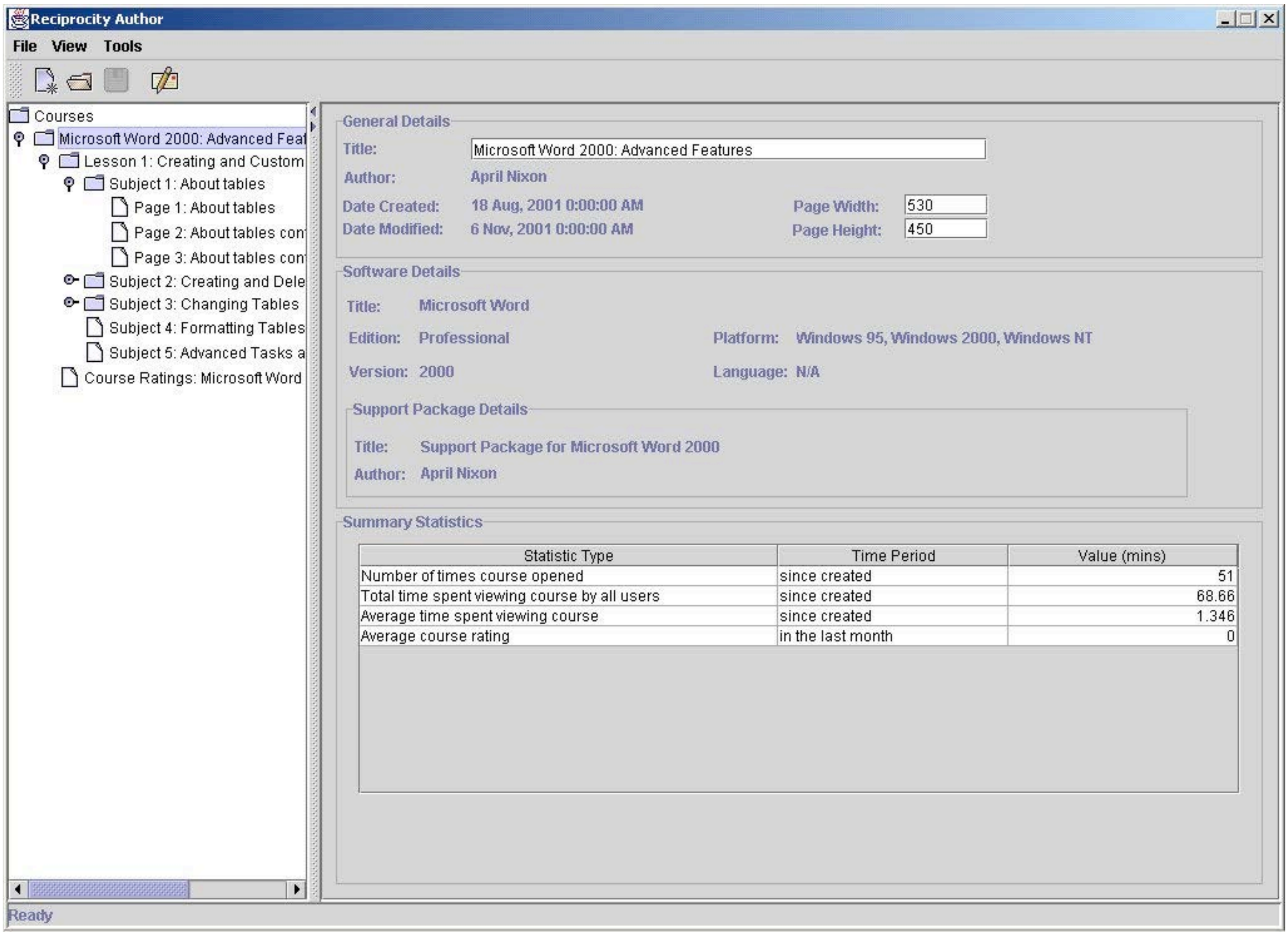
Author Login

User ID: anix007

Password: *****

Main Screen

Once you have successfully logged in, the main screen will appear. It will take up your entire screen and cannot be resized, although it can be minimised and maximised. The main screen consists of a menu bar with a series of menus in it, a toolbar, a split screen and a status bar. The left hand side of the split screen contains the "Structural View" of a course and the right hand side contains the "Detail View". These are described further on in the tutorial. The position of the split is movable, so you can make each side of the split screen as wide as you like.


Reciprocity Author
File
View
Tools
Courses
Microsoft Word 2000: Advanced Feat
Lesson 1: Creating and Custom
Subject 1: About tables
Page 1: About tables
Page 2: About tables con
Page 3: About tables con
Subject 2: Creating and Dele
Subject 3: Changing Tables
Subject 4: Formatting Tables
Subject 5: Advanced Tasks a
Course Ratings: Microsoft Word
Ready

General Details

Title: Microsoft Word 2000: Advanced Features

Author: April Nixon

Date Created: 18 Aug, 2001 0:00:00 AM

Date Modified: 6 Nov, 2001 0:00:00 AM

Page Width: 530

Page Height: 450

Software Details

Title: Microsoft Word

Edition: Professional

Platform: Windows 95, Windows 2000, Windows NT

Version: 2000

Language: N/A

Support Package Details

Title: Support Package for Microsoft Word 2000

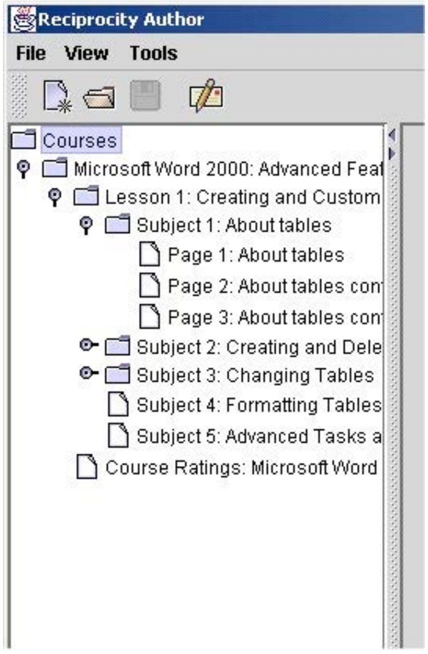
Author: April Nixon

Summary Statistics

Statistic Type	Time Period	Value (mins)
Number of times course opened	since created	51
Total time spent viewing course by all users	since created	68.66
Average time spent viewing course	since created	1.346
Average course rating	in the last month	0

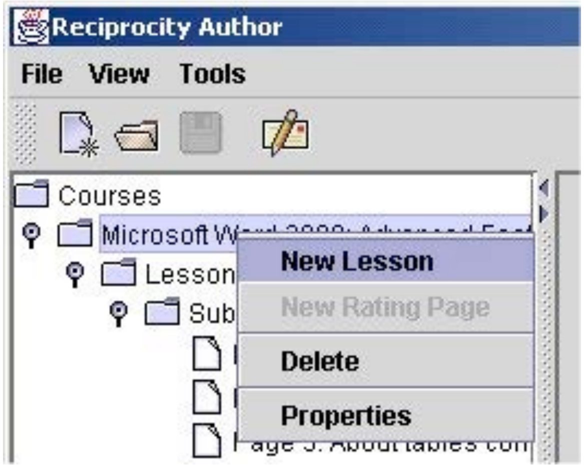
Structural View

The structural view is a tree-like representation of the structure of the course that is currently open. If no course is open, the tree will be empty. If a course is open, it will show the hierarchical structure of the course, including lessons, subjects and pages.


Reciprocity Author
File View Tools
Courses
Microsoft Word 2000: Advanced Feat
Lesson 1: Creating and Custom
Subject 1: About tables
Page 1: About tables
Page 2: About tables con
Page 3: About tables con
Subject 2: Creating and Dele
Subject 3: Changing Tables
Subject 4: Formatting Tables
Subject 5: Advanced Tasks a
Course Ratings: Microsoft Word

Popup Menus

There are several functions that are specific to each type of component that is displayed in the structural view. These functions are available in a popup menu that is displayed by right clicking on the component you want to perform the function on. For example, a function you may wish to perform on a course is adding a new lesson. This can be done by choosing "New Lesson" from the menu that pops up when you right click on the course. All components' popup menus include "Delete" and "Properties…" functions, for deleting and viewing the properties of a component.

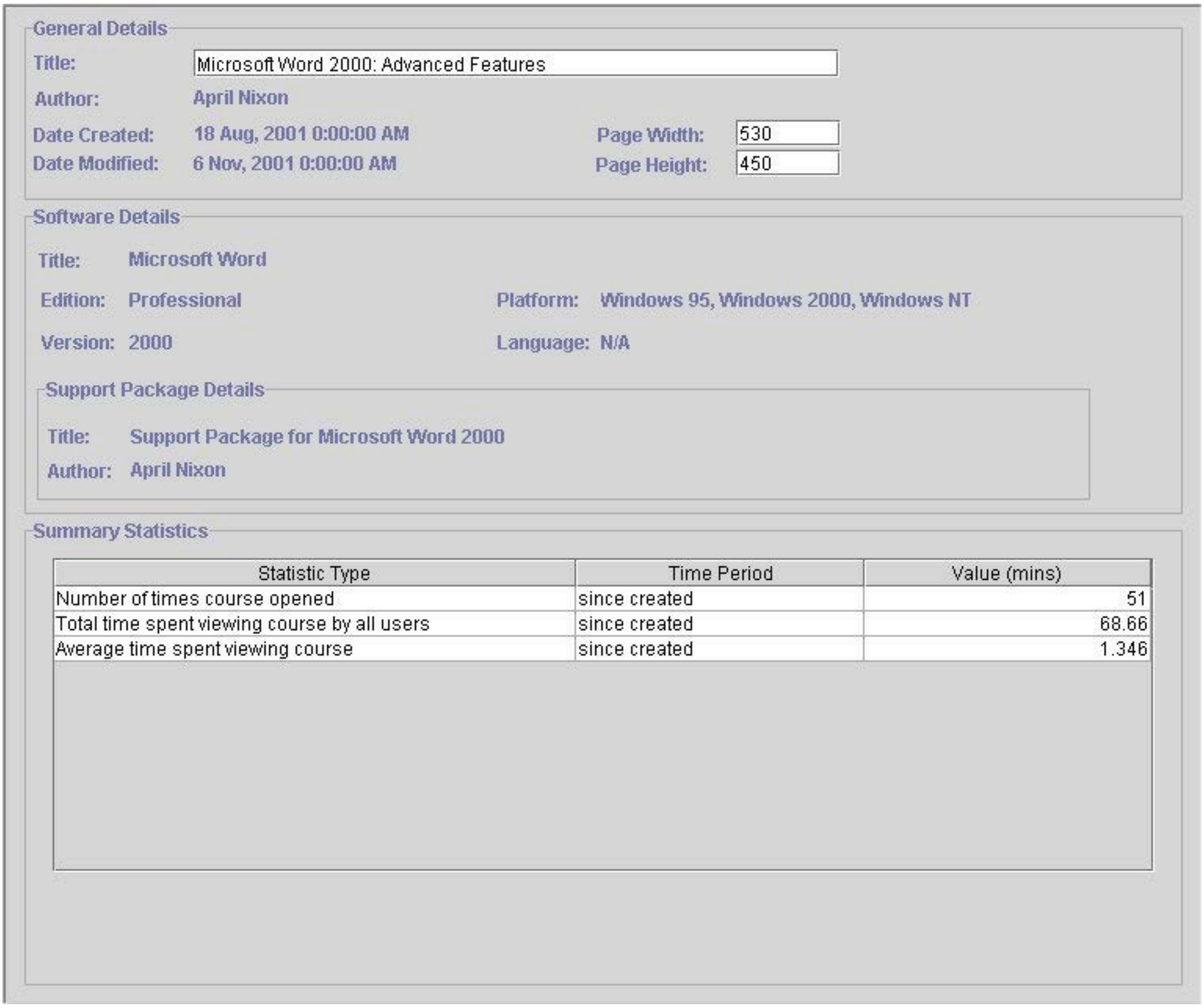


Changing the Order of Components

Another function that is available in structural view is changing the order of components. For instance, you may be working on a course that has five lessons and you realise that lessons four and five should really be swapped around. To do this, press the mouse down on the lesson you would like to move, drag it to the position you would like it to be in and release the mouse. The lesson you moved will be inserted in that position and all other lessons will be moved down to compensate. The lessons will also be automatically renumbered.

Detail View

The detail view takes up the right hand side of the screen and shows detail of whatever component is selected in the structural view. For instance, if a course is selected in structural view, the detail view will show the properties of the course, which includes the title, the author's name, the dates it was created and last modified, the default page size, as well as the details of the software it was developed to teach and a list of summary statistics that the current author has set up to monitor the course's usage.



General Details

Title: Microsoft Word 2000: Advanced Features

Author: April Nixon

Date Created: 18 Aug, 2001 0:00:00 AM

Date Modified: 6 Nov, 2001 0:00:00 AM

Page Width: 530

Page Height: 450

Software Details

Title: Microsoft Word

Edition: Professional

Version: 2000

Platform: Windows 95, Windows 2000, Windows NT

Language: N/A

Support Package Details

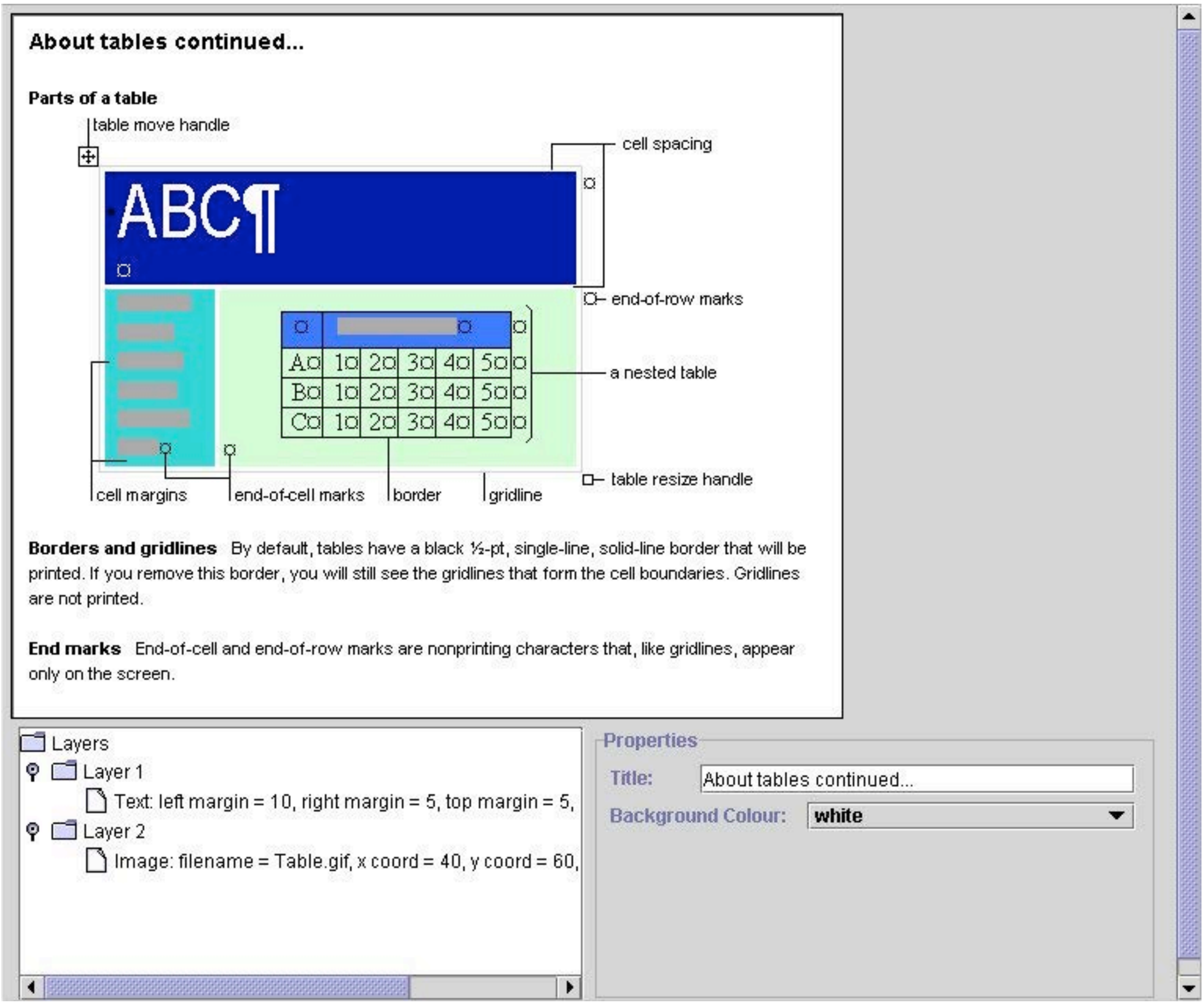
Title: Support Package for Microsoft Word 2000

Author: April Nixon

Summary Statistics

Statistic Type	Time Period	Value (mins)
Number of times course opened	since created	51
Total time spent viewing course by all users	since created	68.66
Average time spent viewing course	since created	1.346

The detail view is very similar for lessons and subjects, showing just their properties. However, the detail view becomes very important when a page is selected. It then becomes the page design area, which allows you to create training course pages by adding text and images as well as setting the properties of the page, including its title, size and background colour.



About tables continued...

Parts of a table

Borders and gridlines By default, tables have a black ½-pt, single-line, solid-line border that will be printed. If you remove this border, you will still see the gridlines that form the cell boundaries. Gridlines are not printed.

End marks End-of-cell and end-of-row marks are nonprinting characters that, like gridlines, appear only on the screen.

Layers
- Layer 1
  - Text: left margin = 10, right margin = 5, top margin = 5,
- Layer 2
  - Image: filename = Table.gif, x coord = 40, y coord = 60,

Properties

Title: About tables continued...

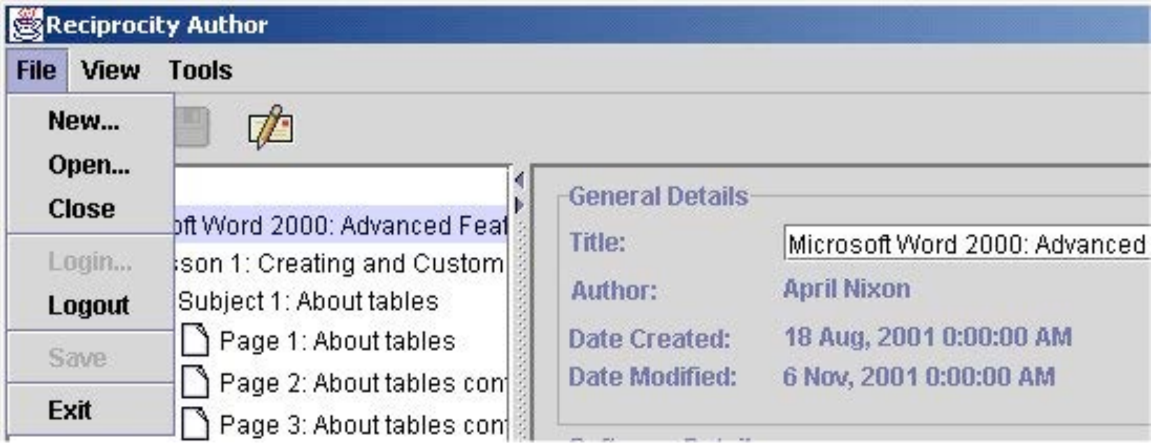
Background Colour: white

Below the actual page area are two panels. The left panel displays the layer information for the current page being displayed. Each page can have as many layers as you like and the layers are displayed on the page so that the higher numbered layers are placed on top of the lower numbered layers. Layers may contain text and/or images. The right panel displays the relevant information for whatever is selected in the layer tree. If nothing is selected, the properties of the page are shown.

Menus and Toolbar

The menu bar is located across the top of the main screen. Initially, the menus that are available are File, View and Tools.

The File menu contains general functions including New, Open, Close, Login, Logout and Exit.



General Details

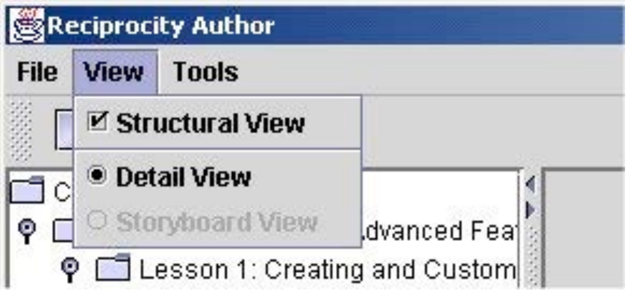
Title: Microsoft Word 2000: Advanced

Author: April Nixon

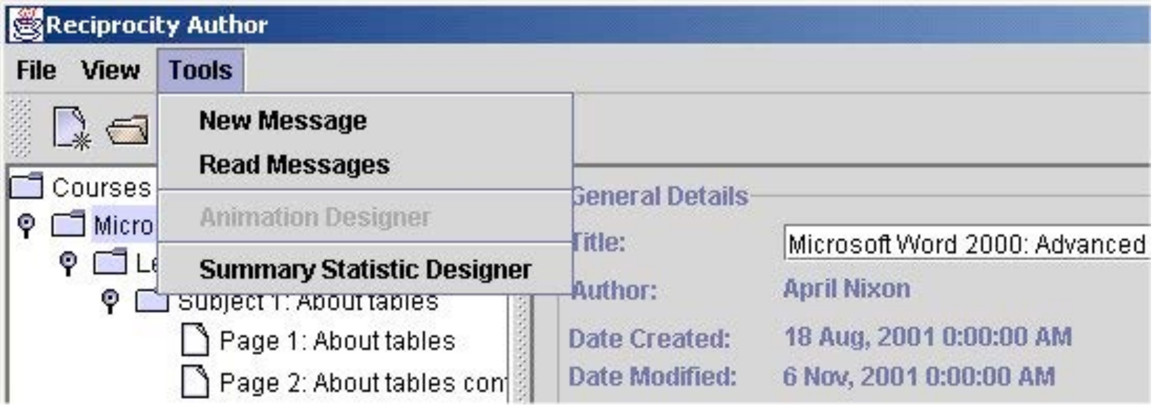
Date Created: 18 Aug, 2001 0:00:00 AM

Date Modified: 6 Nov, 2001 0:00:00 AM

The View menu allows you to choose between the three available views: Structural View, Detail View and Storyboard View. You can choose either to have the structural view there or not there and you can choose between having either the detail or the storyboard view on the right hand side.



The Tools menu contains optional functions including messaging functions, and items that launch the Animation Designer and the Summary Statistic Designer.



General Details

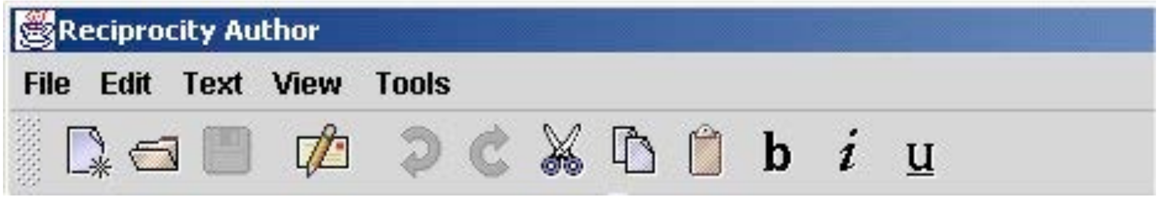
Title: Microsoft Word 2000: Advanced

Author: April Nixon

Date Created: 18 Aug, 2001 0:00:00 AM

Date Modified: 6 Nov, 2001 0:00:00 AM

Once you have selected a page from the structural view, the menus and toolbar will change, with the addition of text formatting functions. This includes the addition of Edit and Text menus a well as several of the most common functions from these menus on the toolbar.



The Edit menu contains typical editing functions, including Cut, Copy, Paste and Select All. The Text menu contains four submenus: Font, Style, Size and Colour.

Status Bar

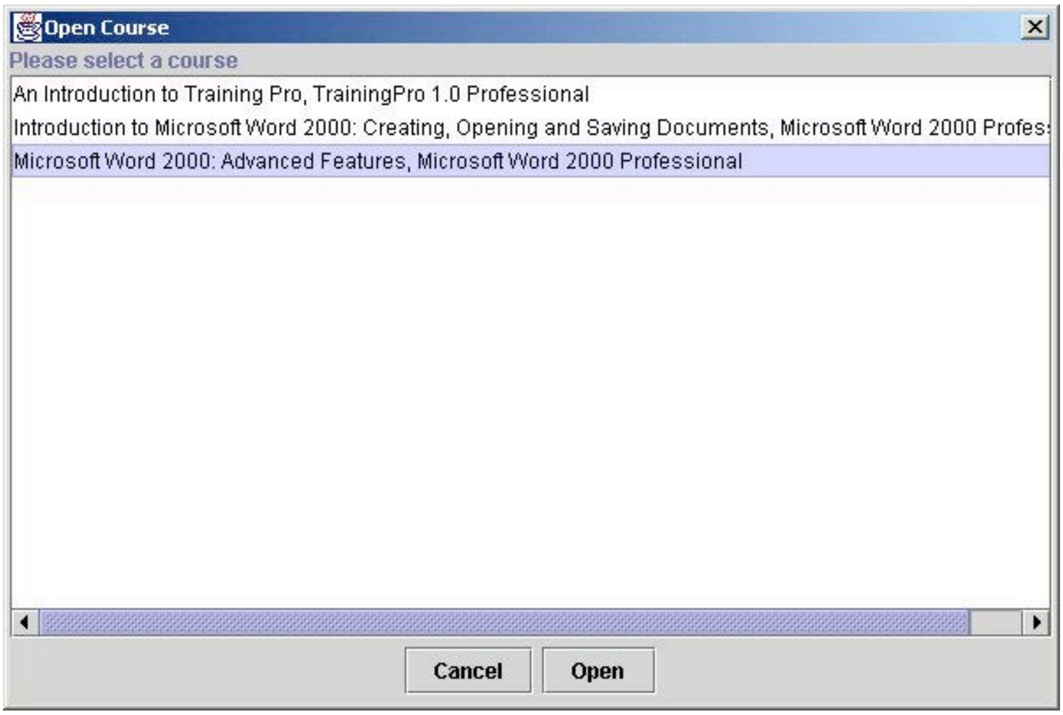
The status bar appears across the bottom of the main screen and contains feedback as to what the application is currently doing. Most of the time the application status is "Ready" but when you perform an action such as opening a course or saving changes, it will show that the operation is taking place and return to "Ready" when it is complete.

Opening an Existing Course

The following steps should be followed to open an existing course:

1. From the File menu, select "Open…". A dialog will be displayed that contains a list of all courses you are permitted to open. Alternatively, you can press the "Open Course" button 🗀 on the toolbar.
2. Select the course you wish to open by clicking on it once.
3. Press "Open". Alternatively, if you decide not to open a course, you can press "Cancel" at any time and the dialog will disappear.

If there was already a course open when you went through this procedure, that course would be closed and the new selected course would be opened.



An Introduction to Training Pro, TrainingPro 1.0 Professional

Introduction to Microsoft Word 2000: Creating, Opening and Saving Documents, Microsoft Word 2000 Profes

Microsoft Word 2000: Advanced Features, Microsoft Word 2000 Professional

Creating a New Component

There are several ways of creating a new component. The standard way is as follows:

1. Choose "New…" from the File menu. Alternatively, you can press the new button on the toolbar.
2. Select the type of new component you would like to create. This can be a support package, course, lesson, subject or page.
3. Depending on the type of component you may or may not be asked to select the parent of the component, but you will need to enter the title of the component.
4. Press "OK".

Alternatively, there are shortcuts for creating new components, which are specific to each type of component. For details of these, see each component type below.

The type of new components you are allowed to create will depend on whether you currently have a course open. If you do not, you will only be able to create a new support package or a new course. If you do, you can create any of the following new components.

Support Package

A support package is a unit that encapsulates all of the training material for a single piece of software. For instance, there may be several courses produced for a single piece of software, and these would all be kept in a single support package.

There is no shortcut to create a new support package because this is a rare activity. To create a new support package:

1. Choose "New…" from the File menu.
2. Select "Support Package" as the component type and press "OK".
3. Select the software that you would like this support package to be for by clicking on it once, and then press "OK".
4. Enter the title of the support package, and then press "Finish".

Course

A course is a single training course for a piece of software that belongs in that software's support package. A course does not have to cover every training topic for a piece of software because there can be multiple training courses for a single piece of software.

There is no shortcut to create a new course because this is also not a common action. To create a new course:

1. Choose "New…" from the File menu.
2. Select "Course" as the component type and press "OK".
3. Select the support package that you would like this course to be put in by clicking on it once, and then press "OK".
4. Enter the title of the course, and then press "Finish".

Lesson

A lesson is the next smallest unit after the course. A course is therefore made up of a set of one or more lessons.
To create a new lesson:

1. Right click on the course in structural view.
2. Select "New lesson" from the popup menu.
3. Enter the title of the lesson and press "OK".

Subject

A subject is the next smallest unit after the lesson. A lesson is therefore made up of a set of one or more subjects. To create a new lesson:

1. Right click on the lesson you want to add the subject to in structural view.
2. Select “New subject” from the popup menu.
3. Enter the title of the subject and press “OK”.

Page

A page is the next smallest unit after the subject. A subject is therefore made up of a set of one or more pages. The page is the part of the course that end users will actually see and may contain media including text and images.

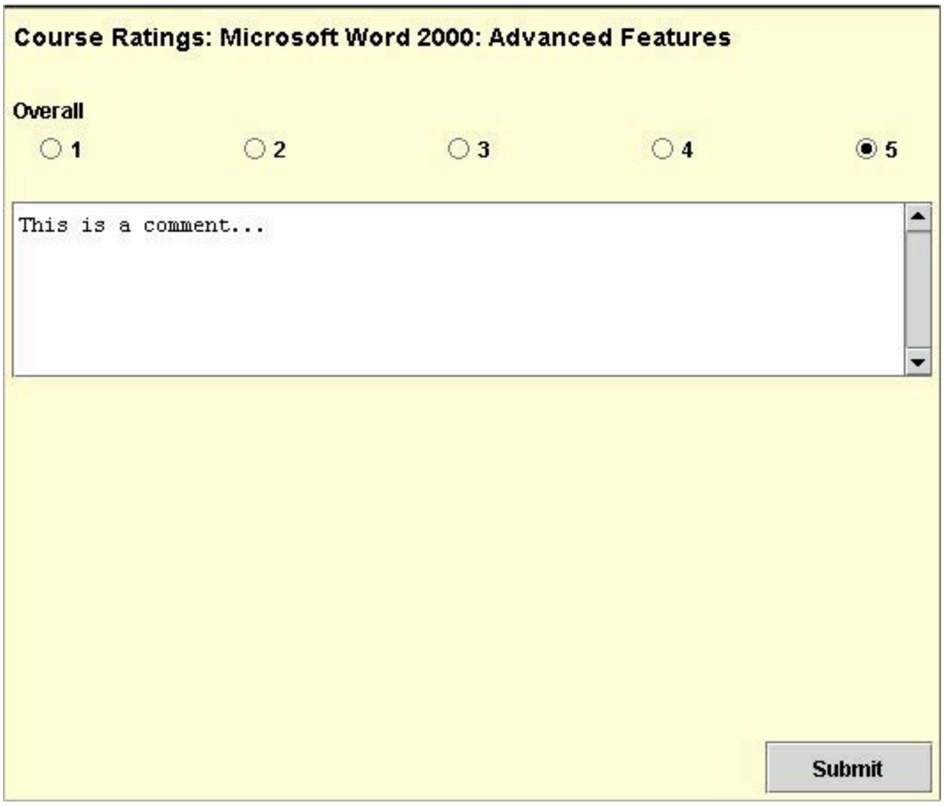
To create a new page:

4. Right click on the subject you want to add the page to in structural view.
5. Select “New page” from the popup menu.
6. Enter the title of the page and press “OK”.

The page will be added as the last page for the subject. If this is not where you intend the page to be, see “Changing the Order of Components” on page 4.

Rating Page

A rating page may be generated for a course, a lesson or a subject. It contains sets of radio buttons that allow an end user to select a rating for criteria that you specify. It may also contain a comments box. You may only create one rating page per course, lesson or subject, so while there may be many rating pages in a course, each lesson or subject may have a maximum of one rating page.



Course Ratings: Microsoft Word 2000: Advanced Features

Overall

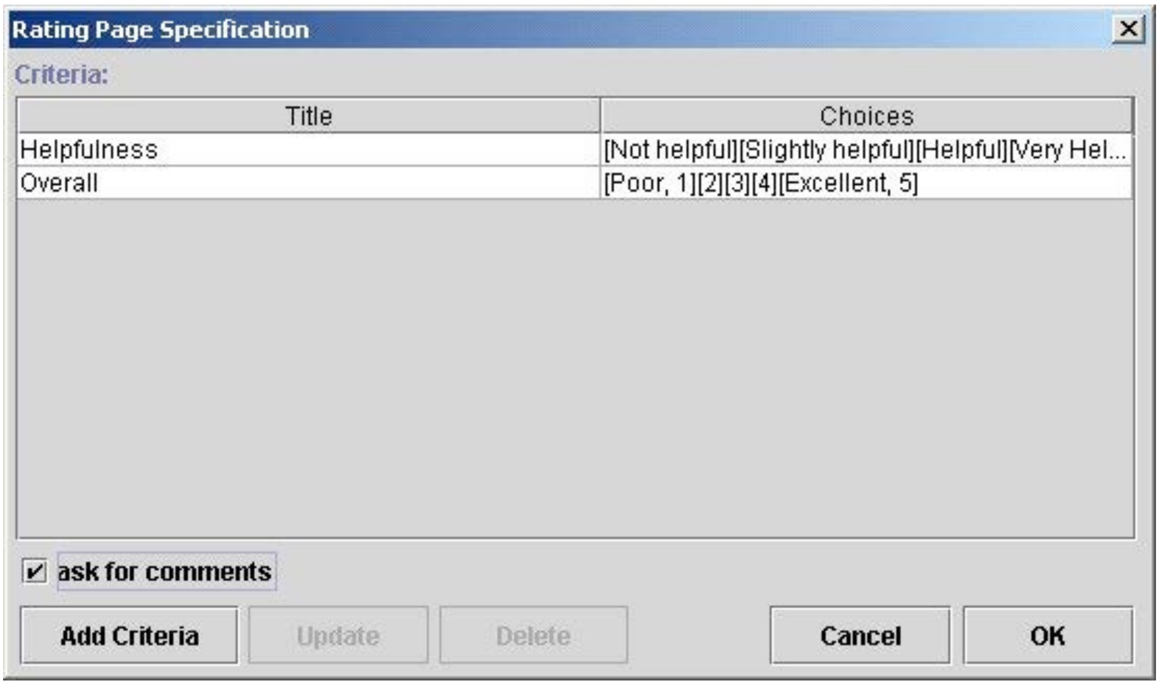
○ 1 ○ 2 ○ 3 ○ 4 ● 5

This is a comment...

To create a new rating page:

1. Right click on the component you wish to add the rating page to in structural view.
2. Select "New rating page" from the popup menu.
3. A dialog will appear that contains a table that lists all of the criteria you have specified for rating the component. You will notice that it is currently empty because as yet, you have not added any criteria.
4. Press "Add Criteria". Another dialog will appear.
5. At the top of the dialog, enter the name of the criteria. "Helpfulness", "Relevance" and "Overall" and examples of possible criteria names.
6. This dialog also has a table, which displays all of the possible choices you want to let an end user choose from when they are rating on this criterion. It should also be empty at this time, because you have not created any. Use the sub-form to add choices, by filling in the value field, the category field or both and pressing "Add".
7. If you wish to delete any of the choices you have created, select the choice in the table by clicking on it once and press "Delete".
8. Once you are satisfied with the choices you have created, press "OK". If you decide you did not want to add this criterion, press "Cancel".

9. The original rating page dialog will again be displayed. The criteria you just specified will now be visible in the table. Repeat steps 4-8 as many times as necessary to add all of the criteria you wish the component to be rated on.
10. If you want to change or delete any of the criteria you have specified, first select the criteria from the table by clicking on it once and then press “Update” or “Delete”.
11. Finally, if you would like a comments box to appear on the rating page, select the “Ask For Comments” checkbox.
12. Once you have finished specifying the rating page, press “OK”. If you decide not to create a rating page, press “Cancel”.

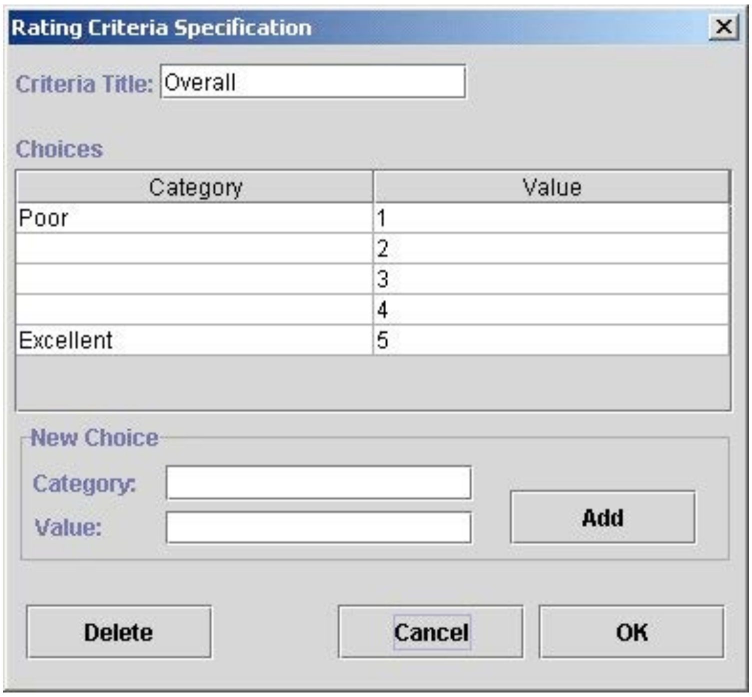


Rating Page Specification

Criteria:

Title	Choices
Helpfulness	[Not helpful][Slightly helpful][Helpful][Very Hel...
Overall	[Poor, 1][2][3][4][Excellent, 5]

☑ ask for comments



Rating Criteria Specification

Criteria Title: Overall

Choices

Category	Value
Poor	1
	2
	3
	4
Excellent	5

New Choice

Category:

Value:

Page Construction

The page is the unit of any training course that end users will actually see. When each page is created, the title given to the page is automatically displayed across the top of the page in detail view. This can be changed or deleted if you wish.

Adding Text

To add text to a page, simply click once on the page to get the I cursor, and then type. The page is just like a Word document when it comes to entering text so if you would like to enter text at the bottom, you would need to press “Enter” until you reach the point you want.

To make changes to text you have entered, there are several functions available in the “Edit” menu. These are:

- Undo ↶
- Redo ↷
- Cut ✂
- Copy ⧉
- Paste 📋

- Select All

All of these functions except "Select All" also have buttons on the toolbar for fast access.

Formatting Text

The options available for formatting text are typical for a simple word processor. This includes:

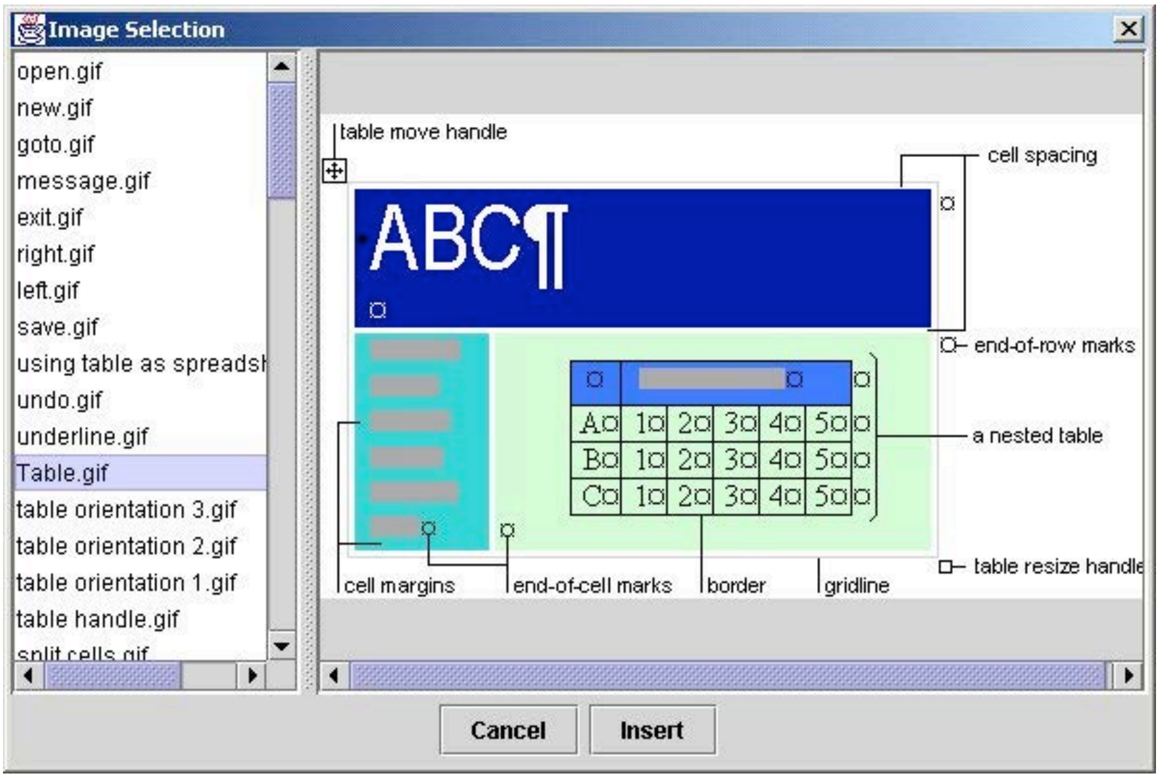
- Font
    - Serif (like writing)
    - Sanserif (like printing)
- Style
    - Bold **b**
    - Italic *i*
    - Underline u
- Size
    - 8-18
- Colour
    - Black
    - Red
    - Green
    - Blue

To access any of these functions, go to the "Text" menu, find the submenu you want and then select the formatting function you want. Some common functions also have buttons on the toolbar.

Adding Images

To add an image to a page, you must first select a layer from the layer tree in detail view.

1. Right click on the layer you wish to add the image to. A popup menu will appear.
2. Select "New Image" from the popup menu. A dialog will appear that displays a list of all available images on the left hand side. By clicking on the name of an image in the list, the image will be displayed on the right hand side.
3. Once you have found the image you wish to add, press "Insert".
4. If at any time you choose not to insert an image, press "Cancel".



Saving Changes

To save changes at any time, either choose “Save” from the File menu, or press the save button 💾 on the tool bar. These options will only be available if you have made any changes since you opened the course or since the last time the course was saved.

Summary Statistics

Summary statistics are statistics that you as an author may set up in order to view reporting information on the use of a course. The statistics you set up are solely for your viewing and will not appear for other authors who access the same course.

Setting up Summary Statistics

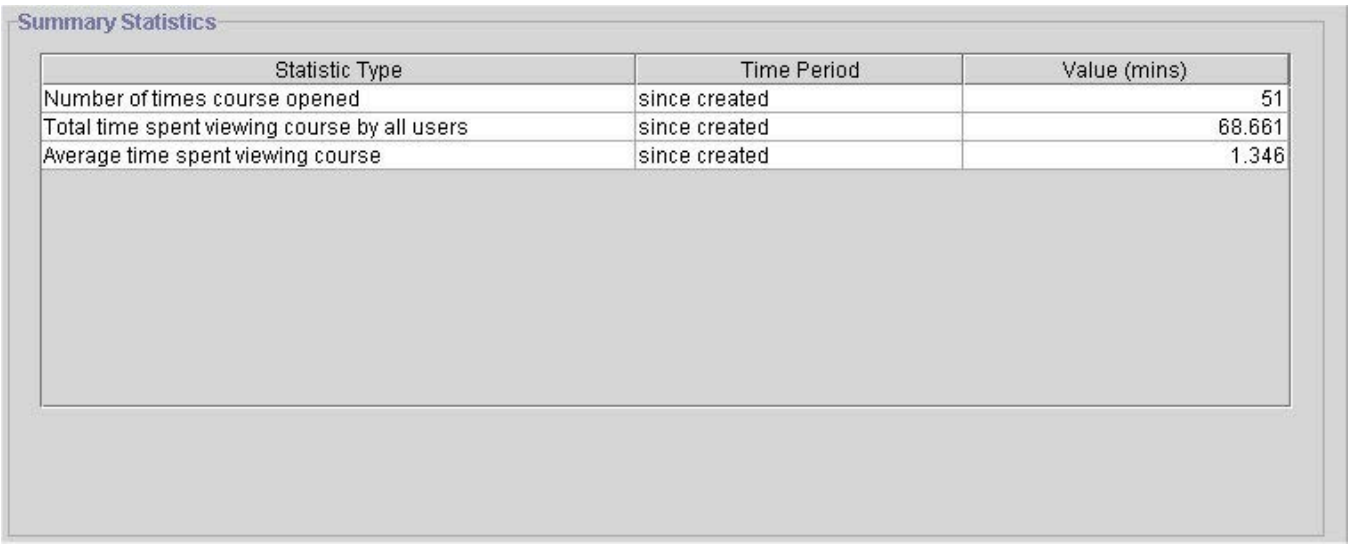
To set up a summary statistic:

1. Select “Summary Statistic Designer” from the Tools menu. A dialog will be displayed that contains a list of all of the current statistics you have set up for this course.
2. Press “Add”. A second dialog will appear.

3. From the top drop-down box, select the type of statistic you would like to create.
4. From the bottom drop-down box, select the time period over which you would like the statistic to report on.
5. Once you are satisfied with the new statistic, press "OK". If you decide not to create a new statistic, press "Cancel".
6. Repeat steps 2-4, as often as needed to create all of the summary statistics you would like.

Viewing the Results of Statistics

To view the results of the reporting statistics you have set up, simply select the course in structural view. The table that is displayed at the bottom of the detail view will show the values of all current summary statistics for this course.



Statistic Type	Time Period	Value (mins)
Number of times course opened	since created	51
Total time spent viewing course by all users	since created	68.661
Average time spent viewing course	since created	1.346

Messaging

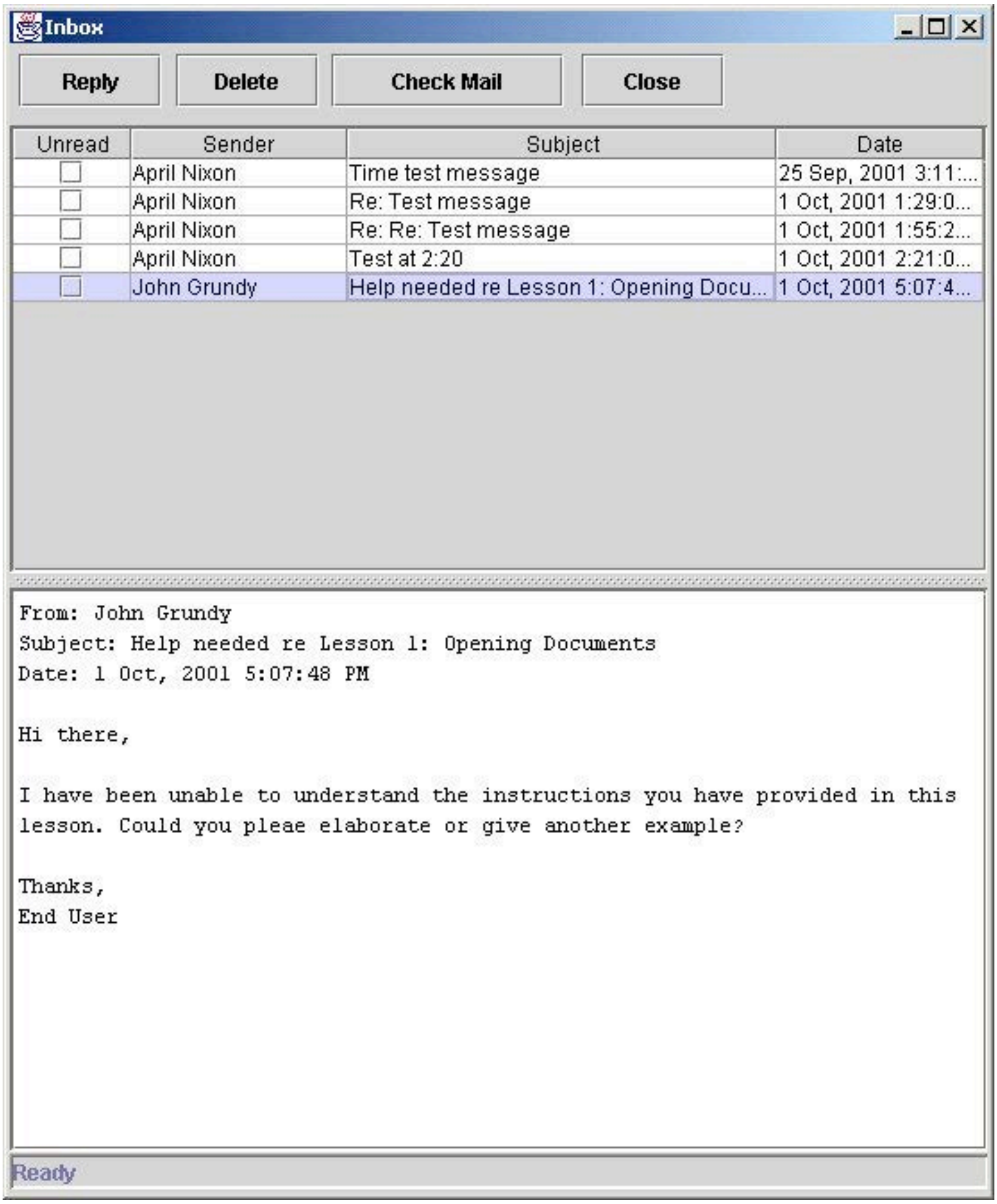
This application allows you to exchange messages with the viewers of your training courses. You are able to send broadcast messages to all of the viewers of one or more training courses, as well as to send unicast messages that are in response to a message you have received from a viewer.

Viewing Mail

To view the contents of your inbox and check for new mail:

1. Select "Read Messages" from the Tools menu. You inbox will be displayed.
2. To view any of the messages in your inbox, select the message from the list by clicking on it once. The contents will be displayed in the bottom pane of the dialog.

3. To delete a message, select the message from the list by clicking on it once and press “Delete”.
4. If you have had your inbox open for some time and would like to check you mail to see if you have received any since it was opened, press “Check Mail”. If you have received any new messages, they will be displayed at the bottom of the list. If not, a message indicating this will appear in the status bar.



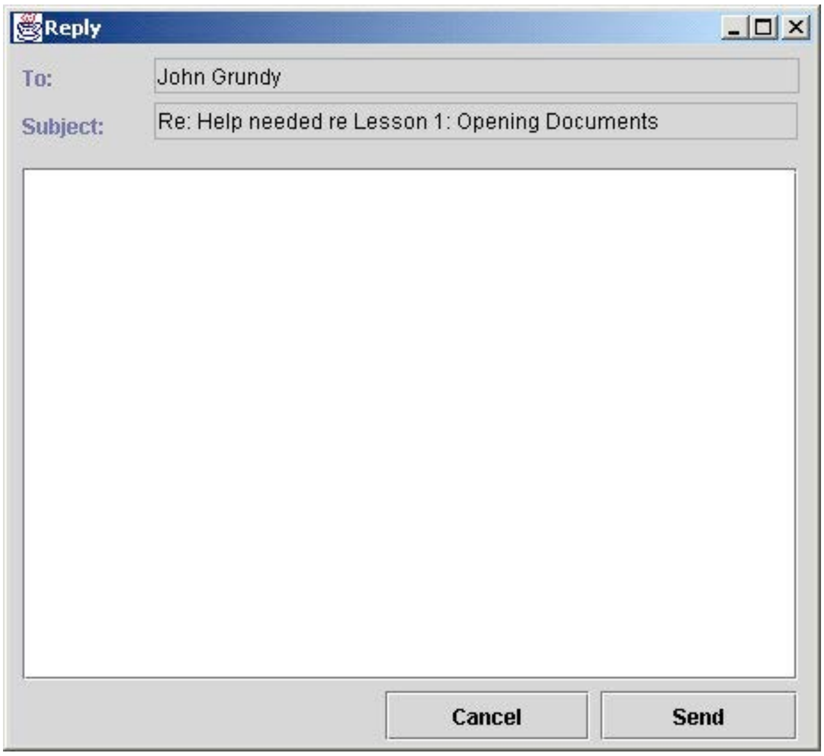
Unread	Sender	Subject	Date
□	April Nixon	Time test message	25 Sep, 2001 3:11:...
□	April Nixon	Re: Test message	1 Oct, 2001 1:29:0...
□	April Nixon	Re: Re: Test message	1 Oct, 2001 1:55:2...
□	April Nixon	Test at 2:20	1 Oct, 2001 2:21:0...
□	John Grundy	Help needed re Lesson 1: Opening Docu...	1 Oct, 2001 5:07:4...

```
From: John Grundy
Subject: Help needed re Lesson 1: Opening Documents
Date: 1 Oct, 2001 5:07:48 PM

Hi there,

I have been unable to understand the instructions you have provided in this
lesson. Could you pleae elaborate or give another example?

Thanks,
End User
```

Composing a Reply

To reply to a message you have received take the following steps:

1. Select the message you wish to reply to from the list by clicking on it once.
2. Press "Reply". A message composer will appear. The recipient and subject of the message will already be entered for you and may not be changed.
3. Enter the body of the message by typing in the text area.
4. Press "Send". If you decide not to send the reply, press "Cancel".



To: John Grundy

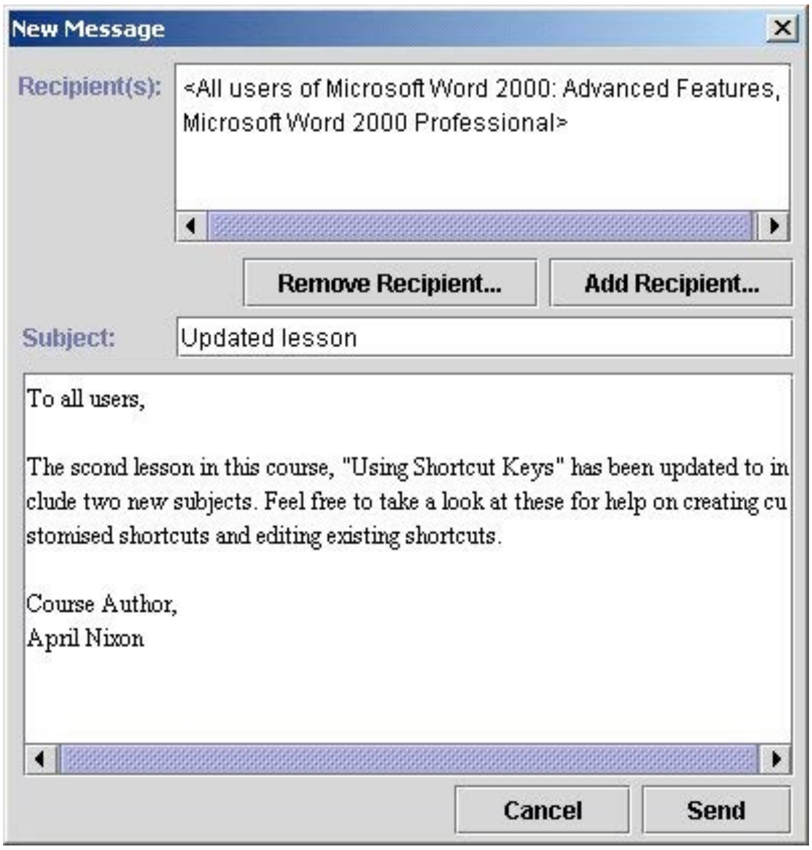
Subject: Re: Help needed re Lesson 1: Opening Documents

Composing a New Message

To compose a new broadcast message:

1. Select "New Message" from the Tools menu. Alternatively, you can click the new message button 📝 on the toolbar. A message composer will appear.
2. To add a group of recipients to the message, press "Add Recipient…". A second dialog will appear, containing a list of all courses you are able to view.
3. Select a course by clicking on it once and pressing "Add". You can add as many courses as you would like to the recipient list. Once you have finished press "Close".
4. If you decide you do not want to send the message to a particular group of recipients, press "Remove Recipient…", select the course name and press "Remove". When you have finished, press "Close".

5. Enter the subject of the message.
6. Enter the body of the message.
7. Press “Send”. If you decide not to send the message, press “Cancel”.



<All users of Microsoft Word 2000: Advanced Features, Microsoft Word 2000 Professional>

Updated lesson

To all users,

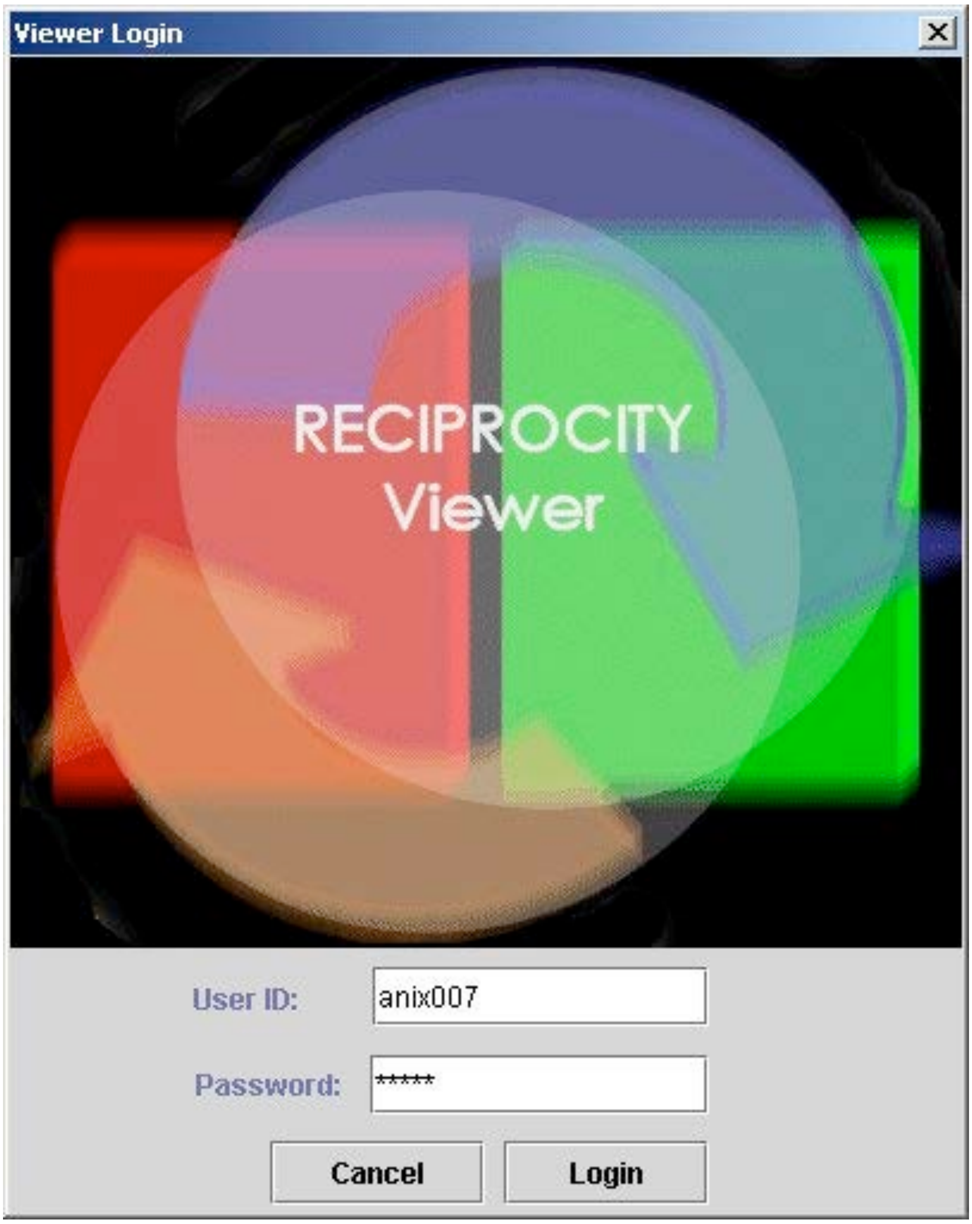
The scond lesson in this course, "Using Shortcut Keys" has been updated to in
clude two new subjects. Feel free to take a look at these for help on creating cu
stomised shortcuts and editing existing shortcuts.

Course Author,
April Nixon

Reciprocity Viewer Tutorial

Login

This first screen you will be presented with when you open the Author application is a login screen. To log in, simply enter your user ID and password and press “Login’. If you decide not to log in, pressing “Cancel” will close the login screen.



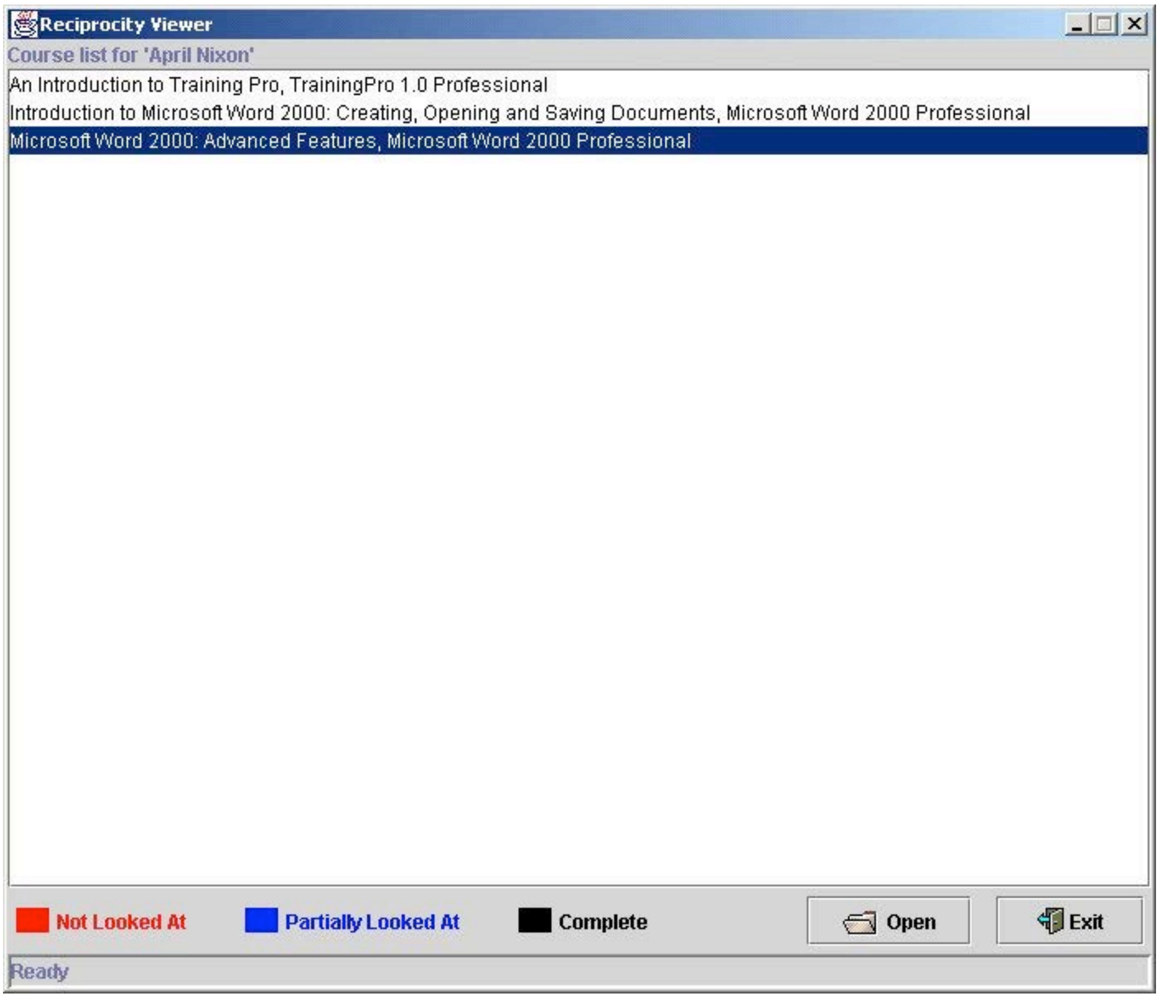
User ID: anix007

Password: *****

Figure 1: viewer login screen

Main Screen

Once you have successfully logged in, the main screen will appear. It will not take up your entire screen and can be resized, minimised and maximised. The main screen consists of a list of the courses you may view, a colour legend, two buttons and a status bar.



An Introduction to Training Pro, TrainingPro 1.0 Professional
Introduction to Microsoft Word 2000: Creating, Opening and Saving Documents, Microsoft Word 2000 Professional
Microsoft Word 2000: Advanced Features, Microsoft Word 2000 Professional

Figure 2: viewer main screen

Course Menu

This takes up most of the main screen and is simply a list of all of the courses you are allowed to view. To view a course:

1. Select the course from the list by clicking on it once.
2. Press “Open”. The main screen will disappear and be replaced by the course viewer.

To exit the application, press “Exit”. You will be prompted for confirmation that you want to exit.

Progress Feedback

Each course in the list is displayed in a specific colour, depending on how far you have progressed in viewing it. The purpose of this is to give you feedback on your progress, to remind you what you have and have not looked at and to inform you if changes or updates have been made to the course. If you have not looked at a course at all, the course will be displayed red. If you have partially viewed a course but not completed it, the course will be displayed blue, and finally if you have completely viewed a course, it will appear black.

Legend

The legend in the bottom left hand corner of the main screen explains the colours that appear in the course list.

Course Viewer

The course viewer allows you to navigate around a single course. Like to main screen, it does not take up the entire screen but can be resized, minimised and maximised. The screen contains a series of menus, a toolbar and a split screen. The left hand side of the screen displays the table of contents of the course you are viewing and the right hand side displays the pages of the course. The split can be moved left or right to change the width of the table of contents and the page being displayed.

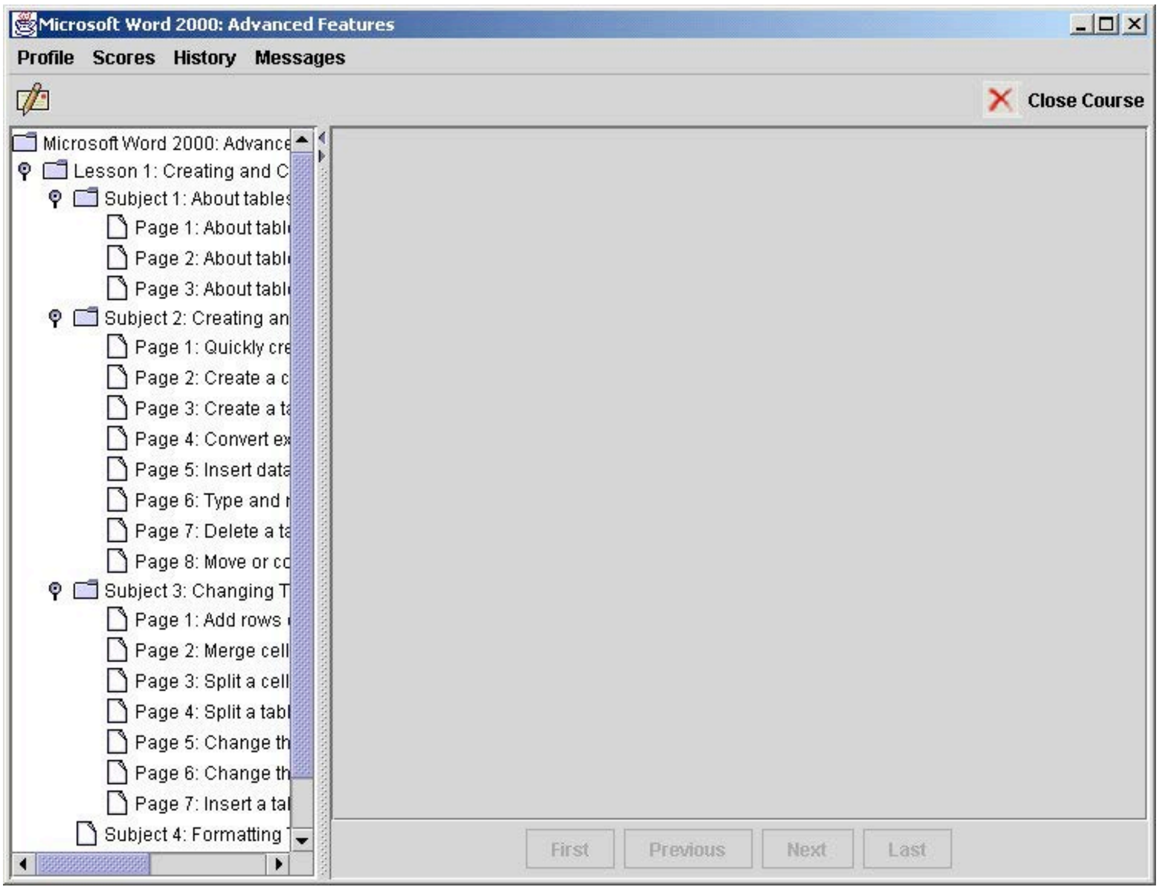


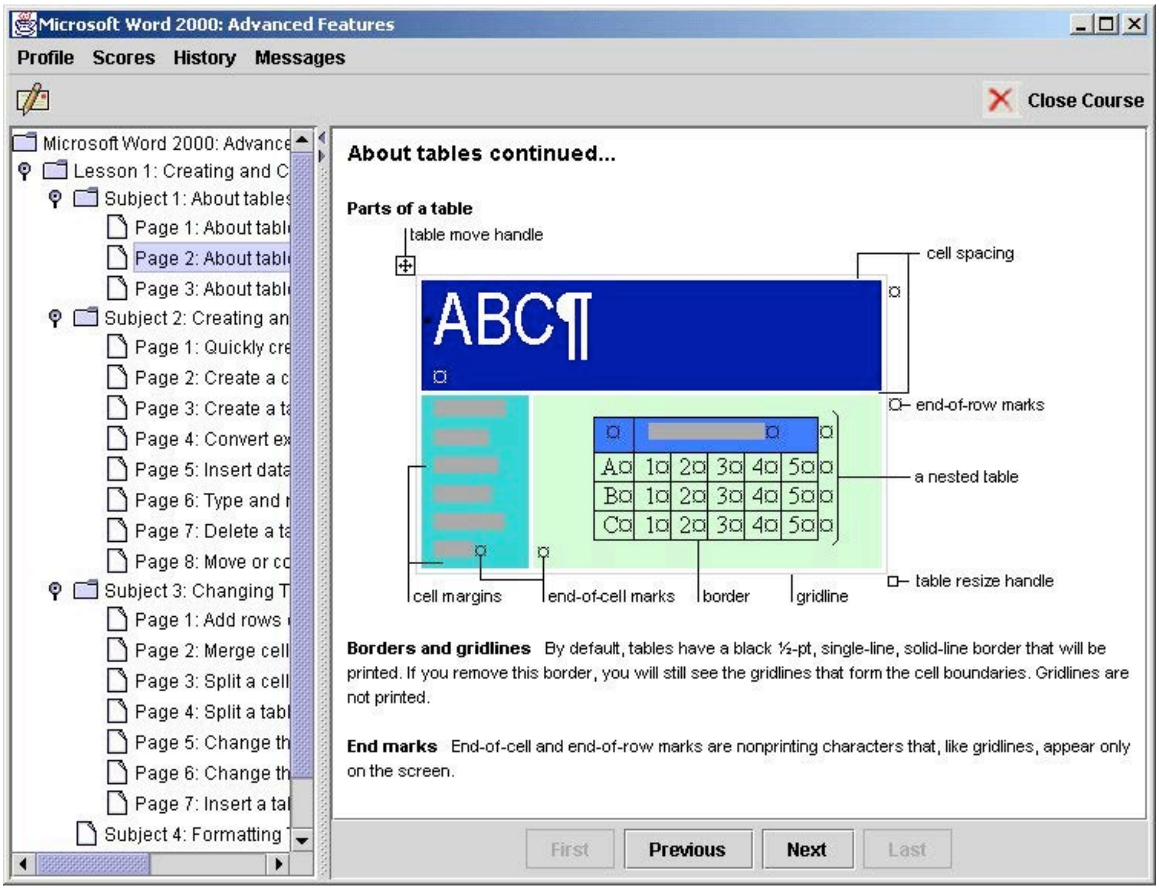
Figure 3: single course viewer initial screen

Table of Contents

The right hand side of the course viewer displays the table of contents for the course you are currently viewing. By clicking on any page that appears in the table of contents you will be taken directly to that page in the course (it will be displayed on the right hand side of the screen). This allows you to go directly to an area of the course that interests you.

Page Display Area

The page display area is on the right hand side of the course viewer screen. Initially this screen is blank, but once you select a page from the table of contents it will be displayed. Once a page is displayed, the navigation buttons at the bottom of the area will become enabled, allowing you to scroll through the pages of the course. These are discussed in detail below.



About tables continued...

Parts of a table

Borders and gridlines By default, tables have a black ½-pt, single-line, solid-line border that will be printed. If you remove this border, you will still see the gridlines that form the cell boundaries. Gridlines are not printed.

End marks End-of-cell and end-of-row marks are nonprinting characters that, like gridlines, appear only on the screen.

Figure 4: course viewer – displaying a selected page

Navigation Buttons

These buttons allow you to scroll backwards and forwards through the course. If you are at the very beginning of the course, the first and previous buttons will be disabled, and likewise, if you at the end of the course, the next and last buttons will be disabled. These buttons enable you to scroll through the course if you want to view the entire course or if you have not viewed it before and do not know what you are looking for.

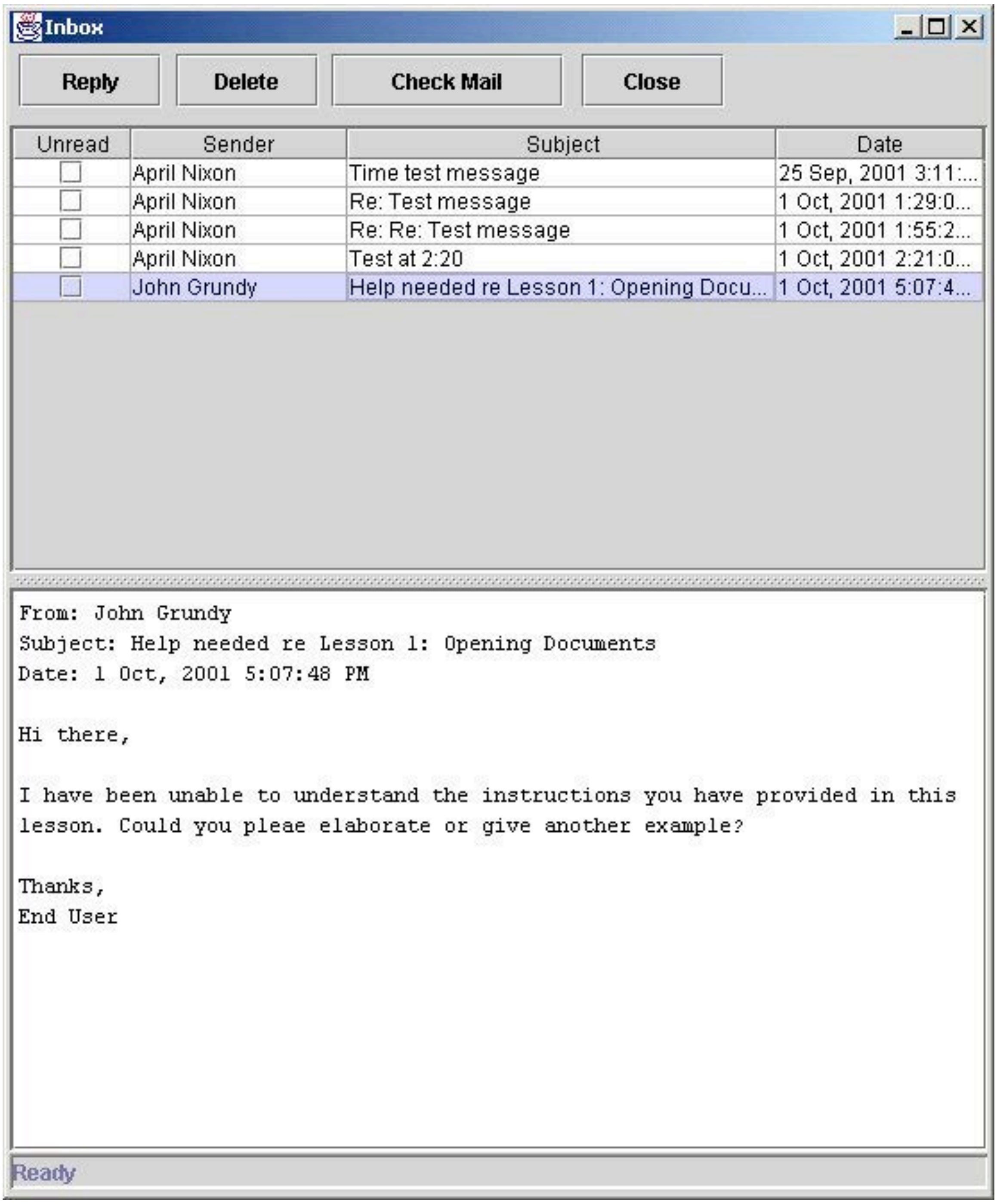
Messaging

This application allows you to exchange messages with the viewers of your training courses. You are able to send broadcast messages to all of the viewers of one or more training courses, as well as to send unicast messages that are in response to a message you have received from a viewer.

Viewing Mail

To view the contents of your inbox and check for new mail:

5. Select “Read Messages” from the Messages menu. You inbox will be displayed.
6. To view any of the messages in your inbox, select the message from the list by clicking on it once. The contents will be displayed in the bottom pane of the dialog.
7. To delete a message, select the message from the list by clicking on it once and press “Delete”.
8. If you have had your inbox open for some time and would like to check you mail to see if you have received any since it was opened, press “Check Mail”. If you have received any new messages, they will be displayed at the bottom of the list. If not, a message indicating this will appear in the status bar.



Unread	Sender	Subject	Date
□	April Nixon	Time test message	25 Sep, 2001 3:11:...
□	April Nixon	Re: Test message	1 Oct, 2001 1:29:0...
□	April Nixon	Re: Re: Test message	1 Oct, 2001 1:55:2...
□	April Nixon	Test at 2:20	1 Oct, 2001 2:21:0...
□	John Grundy	Help needed re Lesson 1: Opening Docu...	1 Oct, 2001 5:07:4...

```
From: John Grundy
Subject: Help needed re Lesson 1: Opening Documents
Date: 1 Oct, 2001 5:07:48 PM

Hi there,

I have been unable to understand the instructions you have provided in this
lesson. Could you pleae elaborate or give another example?

Thanks,
End User
```

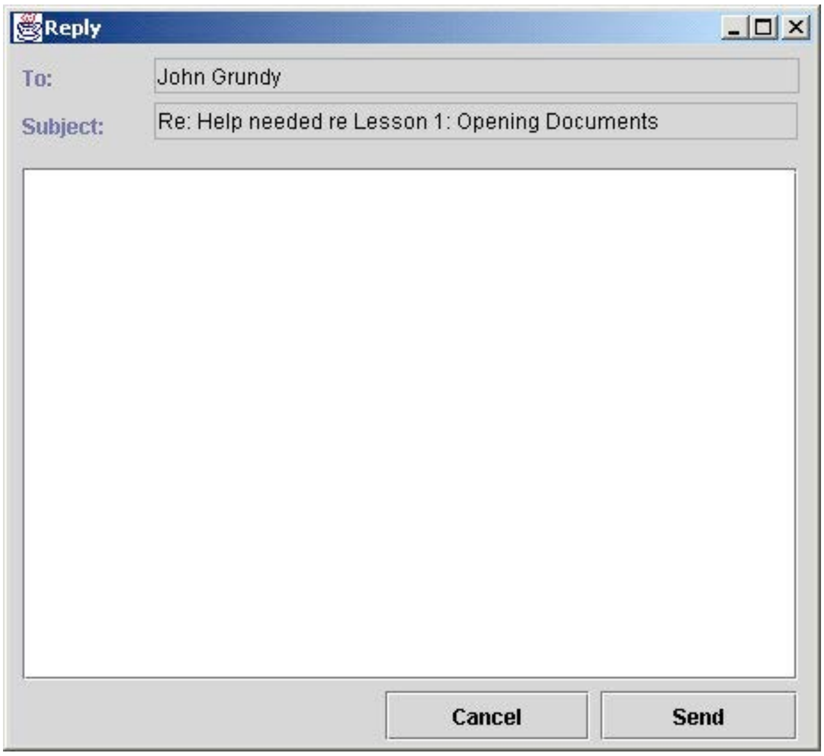
Figure 5: message inbox

Composing a Reply

To reply to a message you have received take the following steps:

5. Select the message you wish to reply to from the list by clicking on it once.

6. Press “Reply”. A message composer will appear. The recipient and subject of the message will already be entered for you and may not be changed.
7. Enter the body of the message by typing in the text area.
8. Press “Send”. If you decide not to send the reply, press “Cancel”.


Reply

To: John Grundy

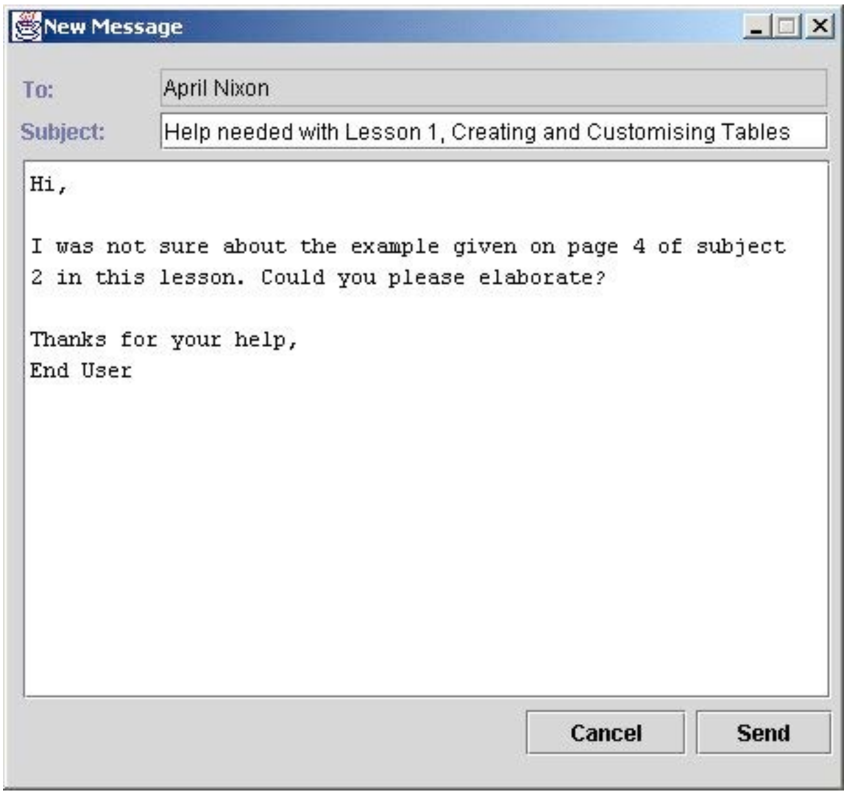
Subject: Re: Help needed re Lesson 1: Opening Documents

Figure 6: reply composer

Composing a New Message

To compose a new broadcast message:

8. Select “New Message” from the Messages menu. Alternatively, you can click the new message button on the toolbar. A message composer will appear. The “to” field will already be filled in as all messages go directly to the author of the course.
9. Enter the subject of the message.
10. Enter the body of the message.
11. Press “Send”. If you decide not to send the message, press “Cancel”.



To: April Nixon

Subject: Help needed with Lesson 1, Creating and Customising Tables

```
Hi,

I was not sure about the example given on page 4 of subject
2 in this lesson. Could you please elaborate?

Thanks for your help,
End User
```

Figure 7: new message composer

Reciprocity Author Task Sheet

Scenario

You are an employee at a software development company. Your company has just completed the development of a new piece of software, named Microsoft Word, and you are responsible for creating training material for all advanced users of the new software.

The section of training material you have been allocated to create is:

Using Shortcut Keys

Tasks

The following tasks will run through some of the typical actions that you would take in constructing this training material. You will by no means have to construct an entire training course ☺

1. Open the Reciprocity Author application. Your user ID = ……………………… and your password = ……………………………
2. Since you are contributing only part of the training course for advanced users, another author has already created the course. Open the course named "Microsoft Word 2000: Advanced Features". You will notice that the course already has a lesson in it, created by another author.
3. Add the following new components to the course:

   3.1. A lesson entitled "Using Shortcut Keys".

   3.2. A subject entitled "Introduction" to the lesson above.

   3.3. A second subject entitled "Assigning custom shortcut keys" to the lesson above.

   3.4. A page entitled "About using shortcut keys" to the Introduction subject.

   3.5. A second page entitled "Assign shortcut keys to a command or other item" to the Introduction subject.
4. To the page you just created entitled "About using shortcut keys", add the following media objects:

4.1. Text:

“**What are shortcut keys?**

You can quickly accomplish tasks you perform frequently by using shortcut keys — one or more keys you press on the keyboard to complete a task. For example, pressing CTRL+B changes the selected text to bold, just as clicking **Bold** on the **Formatting** toolbar or selecting **Bold** in the **Font** dialog box (**Format** menu) changes the selected text to bold.

Printing a list of shortcut keys

If you frequently refer to the Help topics that list shortcut keys, you may find it helpful to keep a printed copy of those topics. In Help, locate the topic you want and click **Print**. You can also print a list of all the shortcut key assignments in Word.”

4.2. Format the text so that it is in Sanserif font and is size 11.

4.3. Images:

4.3.1. Insert the “bold” icon **B** between the words “Bold” and “in”, in the second sentence of the first paragraph. You may have to make room for the image between the words by spacing to prevent them being obscured. The name of the image file is “word_bold.gif”.

4.3.2. Insert the “print” icon 🖨 after the word “Print” at the end of the second sentence of the second paragraph. The name of the image file is “print.gif”.

5. To the page entitled “Assign shortcut keys to a command or other item”, add the following media objects:

5.1. Text:

“You can assign a shortcut key to a command, macro, font, AutoText entry, style, or a commonly used symbol.

1. On the **Tools** menu, click **Customize**.
2. Click **Keyboard**.
3. In the **Save changes in** box, click the current document name or template in which you want to save the shortcut key changes.
4. In the **Categories** box, click the category that contains the command or other item.
5. In the box to the right, click the name of the command or other item. Any shortcut keys that are

currently assigned appear in the **Current keys** box.

6. In the **Press new shortcut key** box, type the shortcut key combination you want to assign.

7. Click **Assign**."

6. You have been told that the author who created the existing lesson, "Creating and Customising Tables", has some pages that are out of order. Expand the lesson and the subject named "Creating and Deleting Tables" in structural view until you can see the following pages:

Page 1: Quickly create a simple table
Page 2: Create a complex table
Page 3: Create a table inside another table
Page 4: Convert existing text to a table
Page 5: Insert data from a data source as a table
Page 6: Type and move around in a table
Page 7: Delete a table or delete items from a table
Page 8: Move or copy items in a table

The last two pages are out of order and should be swapped so that the pages are listed like this:

Page 1: Quickly create a simple table
Page 2: Create a complex table
Page 3: Create a table inside another table
Page 4: Convert existing text to a table
Page 5: Insert data from a data source as a table
Page 6: Type and move around in a table
Page 7: Move or copy items in a table
Page 8: Delete a table or delete items from a table

7. Now that you have constructed a course, you would like to be able to receive feedback from users about it, so you are going to generate a lesson ratings page.

7.1. Similar to the way you added a page in tasks 3.4 and 3.5, add a rating page to the lesson you created, "Using Shortcut Keys", by selecting "New Rating Page" from the lesson's popup menu that shows when you right click on the lesson. This will open a dialog where you can specify the criteria you want the lesson to be rated on and the possible choices users will have when they rate it. Notice that there are no current criteria listed in the table.

7.2. Press “Add Criteria” to add a single criterion to the rating page. This will open a second dialog where you can specify the possible choices users will have when rating on this criterion.

7.3. Enter the name of the criterion as “Helpfulness”.

7.4. Add four choices to this criterion with the categories Not Helpful, Slightly Helpful, Helpful and Very Helpful. As you type each choice and press “Add”, notice it appear in the table.

7.5. Press “OK” when you have completed this.

7.6. Add a second criterion and call it “Overall”.

7.7. Add five choices to this criterion with the values 1, 2, 3, 4 and 5.

7.8. Again, press “OK” when you have finished. You will now notice that two criteria appear in the table for this rating page.

7.9. The last task is to choose to ask for comments by clicking in the check box at the bottom of the left hand side of the dialog. Now that you have set up the lesson’s rating page, press “OK” to close the dialog.

7.10. If you now look under the lesson in structural view, you will notice that a page has been added called “Lesson Ratings: Using Shortcut Keys”. If you click on this page you will see that ratings page that has been generated.

8. This task involves setting up reporting statistics for the course as a whole.

8.1. Select the course in structural view by clicking on it.

Notice that the information displayed in detail view (the right hand side of the screen) includes a frame entitled “Summary Statistics”. Once at least one statistic has been set up, a table will display these. You are now going to set up a reporting statistic.

8.2. From the Tools menu, select Summary Statistic Designer. You will notice that there are no statistics currently listed.

8.3. Press “New” to create a new statistic.

8.4. From the top drop-down box, select "Average Course Rating" and from the second drop-down box, select "in the last month".

8.5. Press OK to save this. You will now notice that the statistic you just created has been added to the list.

8.6. Press Close to close the Summary Statistic Designer.

8.7. To see the result of creating the new statistic, select any other component in structural view and then select the course again, or minimise the window and then maximise it, to refresh the screen.

9. The last task involves messaging between you and the viewers of the training course you have created. Similar to an email client, this application includes an inbox and the ability to write and send new messages.

9.1. Open your inbox by choosing 'Read Messages" from the Tools menu.

To view the contents of any of the messages, click on the message, and the contents will be displayed in the pane at the bottom of the inbox.

9.2. There should be one message from April Nixon. Click on this message once to select and read it.

9.3. Construct a reply to this message by pressing "Reply' when the message is selected. You can write anything you want in the message ☺ Once you have finished, press "Send" to send the message and then press "Close" to close your inbox.

9.4. Now you want to write a message to all of the people who will view this course, perhaps simply to bring their attention to the fact that there is a new course that may be of assistance to them. To write a new message, choose "New Message" from the Tools menu, or alternatively, you can press the compose icon ✎ on the toolbar.

9.5. Add a single recipient to the message: all users of the course you have been creating, "Microsoft Word 2000: Advanced Features".

9.6. Enter the subject and content as anything you want and press "Send".

Reciprocity Viewer Task Sheet

Scenario

You are a user of the software product Microsoft Word 2000. You are a fairly advanced user of Word (don't worry if you are not ☺) and would like to know more about working with tables in Microsoft Word documents.

In order to learn more about tables, you will go through the "Microsoft Word 2000: Advanced Features" training course, focusing on the following lesson:

Creating and Customising Tables

Tasks

The following tasks will run through some of the typical actions that you would take in going through this training course.

1. Open the Reciprocity Viewer application. Your user ID = ……………………… and your password = ……………………………

2. Select the course entitled "Microsoft Word 2000: Advanced Features" by clicking on it once and then press "Open". You will notice that the main course menu disappears and is replaced by a new window. This window contains only the course you have chosen and allows you to navigate through it.

3. Scroll through the lesson entitled "Creating and Customising Tables" by first selecting the first page of the lesson and then using the navigation buttons.

4. Go to the first page of the lesson entitled "Using Shortcut Keys" and have a look at the pages you created earlier using the Author.

5. Go to the course ratings page, either by selecting it from the left hand side menu or by navigating your way to it using the right hand side navigation buttons.

   5.1. Assign any ratings you wish to the criteria listed.

   5.2. Enter any comments you wish and when finished press "Submit".

6. This task involves checking your messages, and is similar to the task you carried out in the Author.

6.1. Open your message inbox by choosing “Read Messages” from the Messages menu.

6.2. Read the contents of any message in your inbox by clicking on the message in the list.

6.3. Write a reply to any message that is in your inbox. When you have finished, press “Send”.

Reciprocity Author Questionnaire

1. Please rate each of the listed current features based on their **usefulness**, **user friendliness** and **importance** to the authoring process.

Possible Ratings:

1 = not useful at all / very difficult to use / unimportant
2 = slightly useful / difficult to use / slightly important
3 = useful / ok to use / important
4 = very useful / easy to use / very important
5 = highly useful / very easy to use / essential

Current Feature	**Useful**	**User Friendly**	**Important**
Content Designer			
Imposed hierarchical content structure			
Available media (text and images)			
Properties display			
Page content tree			
Views			
Structural view			
Detail view			
Summary Statistics			
Table view of values for all summary statistics			
Graphical designer			
Messaging			
Inbox			
Message composer			
Ability to send unicast messages a.k.a. replies			
Ability to send broadcast messages to all of the subscribers of one or more courses			

1.1 If you rated any of the current features with a 1 or 2, please comment on this:

2. Please rate each of the listed possible future features based on how **useful** you think they would be and how **important** they would be to the authoring process.

Possible Ratings:

1 = not useful at all / very difficult to use / unimportant
2 = slightly useful / difficult to use / slightly important
3 = useful / ok to use / important
4 = very useful / easy to use / very important
5 = highly useful / very easy to use / essential

Possible Future Feature	**Useful**	**Important**
Content Designer		
Flexible content structure i.e. can add a child to any node and make it of any type e.g. a subject could have child subjects.		
Existing media loading e.g. can load html, pdf or Word documents.		
Template creation facility (potentially with predefined templates included)		
Printing facility		
Views		
Storyboard view (displays a small picture of each of the pages in a course in order)		
Available media		

Sound		
Animation		
Hyperlinks (to other pages in a course)		
Hyperlinks (to web page locations)		
Summary Statistics		
Predefined statistics i.e. all possible statistics already defined to prevent author from having to do it.		
Graphical display over time for each summary statistic rather than just a numerical figure.		
Messaging		
Ability to send messages to email addresses		
Auto-retrieve facility e.g. checks for new messages every 5 minutes.		
Printing facility		
Other (optional, please specify)		

2.1 If you rated any of the potential future features with a 5, please comment on this:

2.2 If there are any other features you felt would be important or useful, and included in the "other" section in the above list, please outline your reasoning for each:

Reciprocity Viewer Questionnaire

1. Please rate each of the listed current features based on their **usefulness**, **user friendliness** and **importance** to the authoring process.

Possible Ratings:

1 = not useful at all / very difficult to use / unimportant
2 = slightly useful / difficult to use / slightly important
3 = useful / ok to use / important
4 = very useful / easy to use / very important
5 = highly useful / very easy to use / essential

Current Feature	**Useful**	**User Friendly**	**Important**
Navigation			
List of courses to choose from			
Tree showing course structure (clicking on a page takes you directly to it)			
First, previous, next and last buttons			
Media			
Text			
Images			
Communication with Author			
Messaging			
Rating and commenting on courses			
User Feedback			
Status bar			
Colour coding to show progress e.g. red = not looked at, green = partially looked at, black = completed.			
Aesthetics			
Text on buttons			
Colours used			

Readability of text (font and size)			

1.1 If you rated any of the current features with a 1 or 2, please comment on this:

2. Please rate each of the listed possible future features based on how **useful** you think they would be and how **important** they would be to the authoring process.

Possible Ratings:

1 = not useful at all / very difficult to use / unimportant
2 = slightly useful / difficult to use / slightly important
3 = useful / ok to use / important
4 = very useful / easy to use / very important
5 = highly useful / very easy to use / essential

Possible Future Feature	**Useful**	**Important**
Navigation		
Current location displayed at all times		
Hyperlinks between pages		
Hyperlinks (to web page locations)		
Links to the software application you are learning to demonstrate.		
Media		
Sound e.g. narration		
Animation		
Aesthetics		
Ability to choose how the application looks (customise)		
More use of icons e.g. on buttons.		

Accessibility		
Ability to increase or decrease the size of all fonts for readability.		
Printing facility		
Profiling		
Ability to record personal information regarding your current ability with the software in question and have that influence the difficulty of examples given, questions asked etc.		
Testing		
Ability to sit a test after completing a subject/lesson/course and have mark feedback returned.		
Other (optional, please specify)		

2.1 If you rated any of the potential future features with a 5, please comment on this:

2.2 If there are any other features you felt would be important or useful, and included in the “other” section in the above list, please outline your reasoning for each:

Reciprocity Author Questionnaire (Second Round)

1. Please rate each of the listed current features based on their **usefulness**, **user friendliness** and **importance** to the authoring process.

Possible Ratings:

1 = not useful at all / very difficult to use / unimportant

2 = slightly useful / difficult to use / slightly important

3 = useful / ok to use / important

4 = very useful / easy to use / very important

5 = highly useful / very easy to use / essential

Current Feature	**Useful**	**User Friendly**	**Important**
Content			
Imposed hierarchical content structure			
Available media (text, images and animations)			
Page design canvas			
Page content tree (layers)			
Views			
Structural view			
Detail view			
Feedback			
Table view of values for all summary statistics			
Graphical designer for set up of statistics			
Ability to send unicast messages a.k.a. replies			
Ability to send broadcast messages to all of the subscribers of one or more courses			
Distribution Method			
Training courses available immediately (assuming access rights present)			

1.2 If you rated any of the current features with a 1 or 2, please comment on this:

....

....

....

....

....

....

....

....

....

....

....

2. Please rate each of the listed possible future features based on how **useful** you think they would be and how **important** they would be to the authoring process.

Possible Ratings:

1 = not useful at all / very difficult to use / unimportant
2 = slightly useful / difficult to use / slightly important
3 = useful / ok to use / important
4 = very useful / easy to use / very important
5 = highly useful / very easy to use / essential

Possible Future Feature	**Useful**	**Important**
Content		
Flexible content structure i.e. can add a child to any node and make it of any type e.g. a subject could have child subjects.		
Existing media loading e.g. can load html, pdf or Word documents.		
Template creation facility (potentially with predefined templates included).		
Ability to create tests for end-users to sit and receive feedback from.		
Views		
Storyboard view (displays a small picture of each of the pages in a course in order)		

Available media		
Sound		
Video		
Hyperlinks (to other pages in a course)		
Hyperlinks (to web page locations)		
Feedback		
Predefined statistics i.e. all possible statistics already defined to prevent author from having to do it.		
Graphical display over time for each summary statistic rather than just a numerical figure.		
Ability to send messages to email addresses		
Other (optional, please specify)		

2.1. If you rated any of the potential future features with a 5, please comment on this:

....

....

....

....

....

....

....

....

....

....

....

2.2. If there are any other features you felt would be important or useful, and included in the "other" section in the above list, please outline your reasoning for each:

....

....

2.3. Is there any other information you would like gathered or any other viewing method you would like for information already gathered?

2.4. What do you think of the concept of an integrated authoring and delivery system for software training, where feedback is provided to authors to help support an iterative improvement process? Do you think it could be useful in a software support environment?

2.5. If you think it would be useful, what do you like most about the concept?

2.6. If you don’t think it would be useful, what if anything do you feel could be added/changed to make it useful?

Reciprocity Viewer Questionnaire (Second Round)

1. Please rate each of the listed current features based on their **usefulness**, **user friendliness** and **importance** to the authoring process.

Possible Ratings:

1 = not useful at all / very difficult to use / unimportant
2 = slightly useful / difficult to use / slightly important
3 = useful / ok to use / important
4 = very useful / easy to use / very important
5 = highly useful / very easy to use / essential

Current Feature	**Useful**	**User Friendly**	**Important**
Navigation			
Personalised list of courses to choose from			
Table of contents showing course structure (clicking on a page takes you directly to it)			
First, previous, next and last navigation buttons			
Media			
Text			
Images			
Animation (interactive software demonstrations)			
Feedback			
Messaging with authors			
Rating and commenting on courses			
Colour coding to show progress e.g. red = not looked at, green = partially looked at, black = completed.			
Distribution Method			
Desktop application			
Automatically pulls most up to date training courses from server			

1.3 If you rated any of the current features with a 1 or 2, please comment on this:

....

....

....

....

....

....

....

....

....

2. Please rate each of the listed possible future features based on how **useful** you think they would be and how **important** they would be to the authoring process.

Possible Ratings:

1 = not useful at all / very difficult to use / unimportant
2 = slightly useful / difficult to use / slightly important
3 = useful / ok to use / important
4 = very useful / easy to use / very important
5 = highly useful / very easy to use / essential

Possible Future Feature	Useful	Important
Navigation		
Current location displayed at all times (path)		
Hyperlinks between pages		
Hyperlinks (to web page locations)		
Links to the software application you are learning to demonstrate.		
Media		
Sound e.g. narration		
Video		
Profiling		
Ability to record personal information regarding your current ability with the software in question and have that influence the difficulty of examples given, questions asked etc.		

Testing		
Ability to sit a test after completing a subject/lesson/course and have mark feedback returned.		
Distribution Method		
Applet viewable in a web browser		
Other (optional, please specify)		

2.1. If you rated any of the potential future features with a 5, please comment on this:

2.2. If there are any other features you felt would be important or useful, and included in the “other” section in the above list, please outline your reasoning for each:

Appendix B Complete Use Case Descriptions

Author Use Cases

Administrative Task Use Cases

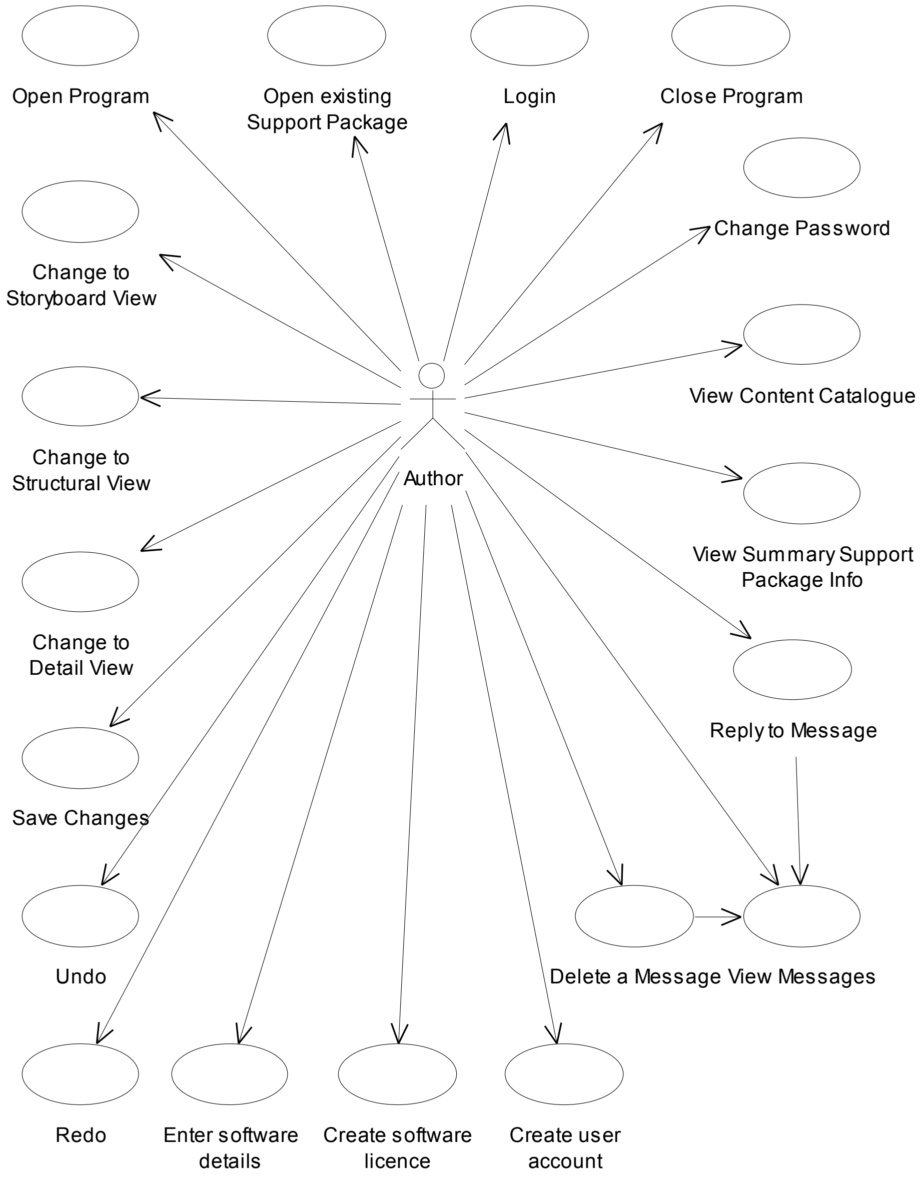


Figure 4.3 shows the use cases related to administrative tasks a user might wish to perform. These are tasks that do not directly involve the authoring of a training course but are required in order to make courses available to appropriate users and to effectively run the authoring tool. The details of four of these use cases are described below. The full description of all use cases can be found in Appendix B.

Login

Basic Flow:

Precondition - program has just been opened.

1. Enter user name.
2. Enter password.
3. Press enter or click login.

Open Existing Support Package

Basic Flow:

Precondition - authoring program has just been opened.

1. Select "open existing support package" from the initial dialog box.
2. Select the file from the file chooser that appears.
3. Press open.

Alternate Flow 1:

Precondition - authoring program is already open e.g. no initial dialog box open.

1. Select the file menu.
2. Choose open...
3. Select the support package file from the file chooser that appears.
4. Press open.

Alternate Flow 2:

Precondition - authoring program is already open e.g. no initial dialog box open.

1. Select the file menu.
2. Choose one of the recent support package titles from the list at the bottom of the menu.

Change Password

Basic Flow:

Precondition - user must already be logged into the program.

1. Select Tools menu.
2. Select change password.
3. Enter current password.
4. Enter new password twice.
5. Press OK (or cancel).

Change to Storyboard View

Basic Flow:

Precondition - a course must be open and expanded.

1. Select a lesson in storyboard view.
2. Select storyboard view from the View menu or press the storyboard view toolbar button.
3. Structural view is hidden and storyboard takes up the entire working area of the interface.

Change to Detail View

Basic Flow:

Precondition - currently in storyboard view.

1. Select detail view from the View menu or press the detail view toolbar button.
2. Structural view and detail view are displayed with the lesson page that was selected in storyboard view (default is the first page) displayed in detail view and highlighted in structural view.

Change to Structural View

Basic Flow:

Precondition - currently in storyboard view.

1. Select structural view from the View menu or press the structural view toolbar button.
2. Structural view is displayed with the lesson that was being viewed in storyboard view as the highlighted lesson in structural view and nothing in the detail view (because there is no page for a lesson node).

View Content Catalogue

Basic Flow:

Precondition - a support module must be open (only one can be open at a time).

1. Select the view menu.

2. Select content catalogue.
3. A pop-up window containing the content catalogue will appear.
4. To close, press the close window button.

Alternate Flow 1:

1. Press the course catalogue view toolbar button.
2. A pop-up window containing the content catalogue will appear.
3. To close press the close window button.

View Summary Support Package Info

Basic Flow:

Precondition - author has already designed the summary info.

1. Highlight the support package node in structural view.
2. The detail view will then be displayed which contains previously designed summary usage information.

View Messages

Basic Flow:

Precondition - support package has just been opened.

1. Message pops up that says "x new messages".
2. Select view messages.
3. Message viewing window appears (like an inbox). From here the author can also delete or reply to messages.
4. Close the message window.

Alternate Flow 1:

Precondition - support package is open but hasn't just been opened.

1. Select view menu.
2. Select messages.
3. Message viewing window appears (like an inbox). From here the author can also delete or reply to messages.
4. Close the message window.

Reply to Message

Basic Flow:

1. View messages.
2. Select the message to reply to.
3. Press the reply toolbar button or select reply menu option.
4. Enter message content.
5. Press send toolbar button or select send menu option.

Delete Message

Basic Flow:

1. View messages.
2. Select the message to delete.
3. Right click and choose delete or press the delete toolbar button.

Close Program

Basic Flow:

1. Select the File Menu.
2. Select Exit.
3. Answer yes to prompt to exit.
4. The application and the currently open support package closes.

Alternate Flow 1:

Precondition - changes have been made since last saving.

1. Select the File Menu.
2. Select Exit.
3. Answer yes to prompt to save changes.
4. The application and the currently open support package closes.

Alternate Flow 2:

1. Select the File Menu

2. Select Exit
3. Answer no to prompt to exit.
4. The application remains open.

Save Changes

Basic Flow:

1. Select the File Menu.
2. Select Save.

Alternate Flow 1:

1. Press the Save toolbar button.

Undo

Basic Flow:

Precondition - at least one change has been made since the support package was last saved.

1. Select the Edit menu.
2. Select Undo.

Alternate Flow 1:

1. Press the Undo toolbar button.

Undo can only be performed on the last action taken.

Redo

Basic Flow:

Precondition - the last action taken was undo.

1. Select the Edit menu.
2. Select Redo.

Alternate Flow 1:

1. Press the Redo toolbar button.

Redo can only be performed on the last action taken.

Create User Account

Basic Flow:

1. Select Tools menu.
2. Select Create User Account. The User Account Maintenance window appears.
3. Enter the personal details of the user, including name and role, as well as their user ID and initial password.
4. Select the software that the user requires an support account for. This is from a list of already entered software.
5. Select the software licence i.e. the licence for the company the user is employed by.
6. Select the user group that this user fits into e.g. viewer, administrator.
7. Press OK or cancel.

Alternate Flow 1:

1. Select Tools menu.
2. Select Create User Account. The User Account Maintenance window appears.
3. Enter the personal details of the user, including name and role, as well as their user ID and initial password.
4. Select create new software (because this software is not on the list).This calls the Enter Software Details use case.
5. Select create new software licence (because there is not currently a licence in the system for this company to use the software). This calls the Create Software Licence use case.
6. Select the user group that this user fits into e.g. viewer, administrator.
7. Press OK or cancel.

Enter Software Details

Basic Flow:

Precondition - the user is in the process of creating a user account but the relevant software has not already been recorded in the system.

1. Enter the details of the software, including name, version, author(s), platform, language, edition etc.
2. Choose Add or cancel.

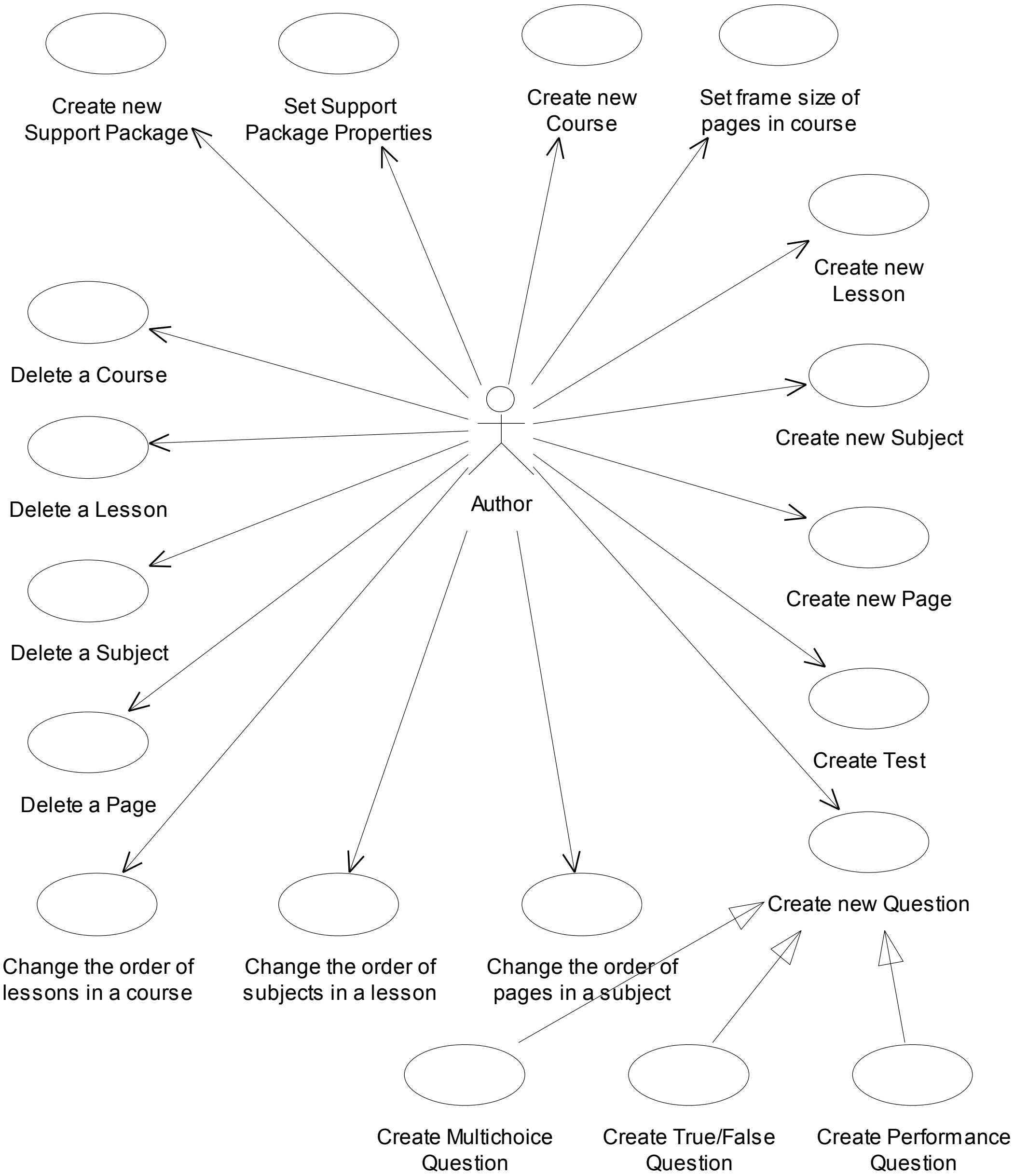
Create Software Licence

Basic Flow:

Precondition - the user is in the process of creating a user account but a software licence for the user's company has not already been recorded in the system.

1. Select the software from a pre-defined list.
2. Enter the company details and the licence terms e.g. time period valid, number of concurrent users etc.
3. Choose Add or cancel.

Course Structure Creation Task Use Cases


Create new Support Package
Set Support Package Properties
Create new Course
Set frame size of pages in course
Create new Lesson
Delete a Course
Create new Subject
Delete a Lesson
Author
Create new Page
Delete a Subject
Create Test
Delete a Page
Create new Question
Change the order of lessons in a course
Change the order of subjects in a lesson
Change the order of pages in a subject
Create Multichoice Question
Create True/False Question
Create Performance Question

Create New Support Package

Basic Flow:

1. Open the authoring program
2. Select "create new support package" from the options given in the initial dialog box.
3. Give the support package a file name and directory.
4. Give the support package a title.

Alternate Flow 1:

Precondition - the authoring program is already open.

1. Open the File Menu.
2. Select "new".
3. Select "support package".
4. Give the support package a file name and directory.
5. Give the support package a title.

Set Support Package Properties

Basic Flow:

Precondition - the support package must already exist and structural view must be visible.

1. Select the support package node in the structural view tree.
2. The properties sheet for this package will appear in the detail view (because there is no course page for this node).
3. Enter a value for a property, by typing or selecting.
4. Press enter in the control or move to another control.

Alternate Flow 1:

(For just the title property)

1. Select the support package node in the structural view tree.
2. Click one more time in the title in structural view.
3. Type in the new title.
4. Press enter or click anywhere else on the interface.

Create New Course

Basic Flow:

Precondition - the parent support package must already exist.

1. Select the parent support package in the structural view.
2. Choose the file menu and the new submenu.
3. Select "course".
4. Enter the title of the course.

Alternate Flow 1:

1. Select the parent support package in the structural view.
2. Right click on the selected node in the structural view.
3. Select new course from the pop-up menu.
4. Enter the title of the course.

Set Frame Size of Pages in Course

Basic Flow:

Precondition - the course must already exist.

1. Select the course node in structural view.
2. Select the page size control in detail view.
3. Enter the desired value (probably select from a pre-defined list).
4. Press enter or click somewhere else on the interface.

Create New Lesson

Basic Flow:

Precondition - the parent course must already exist.

1. Select the parent course in structural view.
2. Choose the file menu and the new submenu.
3. Select "lesson".
4. Enter the title of the lesson.

Alternate Flow 1:

1. Select the parent course in structural view.
2. Right click on the selected node.
3. Select new lesson from the pop-up menu.
4. Enter the title of the lesson.

Create New Subject

Basic Flow:

Precondition - the parent lesson must already exist.

1. Select the parent lesson in structural view.
2. Choose the file menu and the new submenu.
3. Select "subject".
4. Enter the title of the subject.

Alternate Flow 1:

1. Select the parent lesson in structural view.
2. Right click on the selected node.
3. Select new subject from the pop-up menu.
4. Enter the title of the subject.

Create New Page

Basic Flow:

Precondition - the parent lesson must already exist.

1. Select the parent lesson node in structural view.
2. Choose the file menu and the new submenu.
3. Select "page".
4. Enter the title of the page.

Alternate Flow 1:

1. Select the parent lesson node in structural view.
2. Right click on the highlighted lesson node.

3. Select new page from the pop-up menu.
4. Enter the title of the page.

Delete a Course

Basic Flow:

Precondition - the course must already exist.

1. Select the course node in structural view.
2. Select the file menu and select "delete".
3. Answer "yes" when prompted for confirmation to delete.

Delete a Lesson

Basic Flow:

Precondition - the lesson must already exist.

1. Select the lesson node in structural view.
2. Select the file menu and select "delete".
3. Answer "yes" when prompted for confirmation to delete.

Delete a Subject

Basic Flow:

Precondition - the subject must already exist.

1. Select the subject node in structural view.
2. Select the file menu and select "delete".
3. Answer "yes" when prompted for confirmation to delete.

Delete a Page

Basic Flow:

Precondition - the page must already exist.

1. Select the page node in structural view.
2. Select the file menu and select "delete".
3. Answer "yes" when prompted for confirmation to delete.

Change the Order of Lessons in a Course

Basic Flow:

Precondition - the course must be open and visible in structural view. It must also be expanded so that all lessons are visible.

1. Click down on the lesson that you want to move.
2. Drag the lesson node to the position where it is to go.
3. Drop the lesson node.
4. The titles of all lessons are recalculated e.g. change from "Lesson 1: <lesson name>" to "Lesson 3: <lesson name>" also causes lesson 2 -> lesson 1 and lesson 3 -> lesson 2.

Change the Order of Subject in a Lesson

Basic Flow:

Precondition - the course must be open and visible in structural view. The lesson must also be expanded so that all subjects in it are visible.

1. Click down on the subject that you want to move.
2. Drag the subject node to the position where it is to go.
3. Drop the subject node.

Change the Order of Pages in a Subject

Basic Flow:

Precondition - the course must be open and visible in structural view. The subject must also be expanded so that all pages in it are visible.

1. Click down on the page that you want to move.
2. Drag the page node to the position where it is to go.
3. Drop the page node.

Create New Test

Basic Flow:

Precondition - lesson must be selected in structural view.

1. Select File menu and new submenu.
2. Select Test. The Test Designer window will open and show that the test is for the highlighted lesson.

3. Choose to add a new question. This calls the Create new Questions use case.
4. Repeat step 3 any number of times.
5. Enter the threshold for passing the test.
6. Press OK or cancel.

Create New Question

Basic Flow:

Precondition - the Test Designer window must be open.

1. Choose to add a new question.
2. Choose the multi-choice question type. This calls the Create Multi-choice Question use case.
3. Enter the threshold for passing the test.
4. Press OK or cancel.

Alternate Flow 1:

2. Choose the true/false question type.

Alternate Flow 2:

2. Choose the performance question type.

Create Multi-choice Question

Basic Flow:

1. Enter the question text.
2. Add at least two possible answers to choose from, including the points scored for each answer (including 0).

Create True/False Question

Basic Flow:

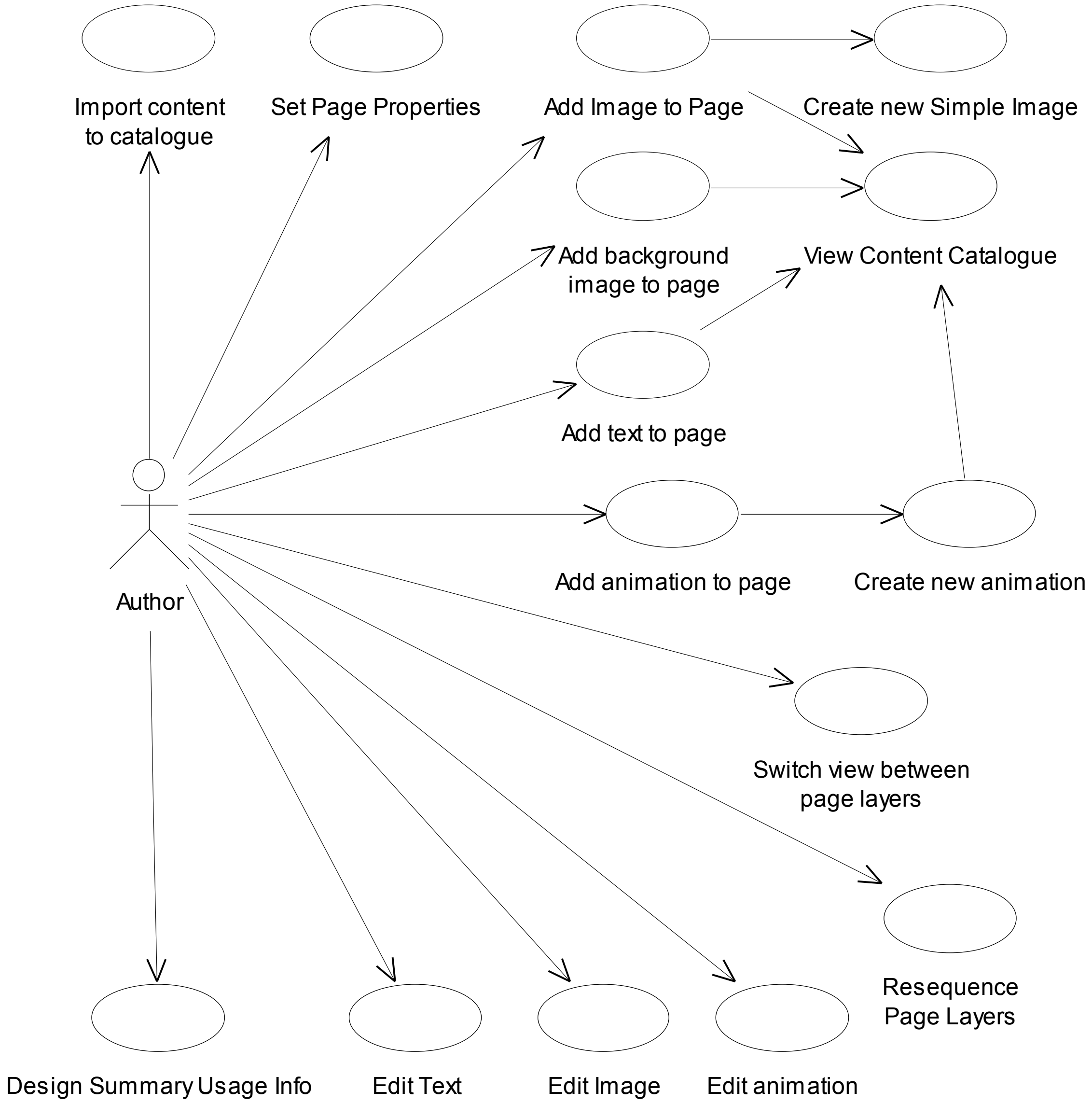
1. Enter the question text.
2. Enter the correct answer (true or false) and the points scored for both answers.

Create Performance Question

Basic Flow:

1. Enter the performance task text.
2. Choose to create the performance animation. This calls the Create New Animation use case.
3. Enter the points scored for correctly completing the task.

Author Content Design Task Use Cases



Import Content to Catalogue

[A catalogue is for a single support module]

Basic Flow:

1. Select the Import menu and the type of file to import: image, audio, text.
2. Select the file to import from the file chooser displayed and press "open".
3. Enter the name of the new catalogue component (default is the file name).
4. The component would be added to the catalogue as the last item. If it is an image, a small copy of the image would show in the catalogue. If it is text or audio, a letter or note would be displayed respectively.

Add Image to Page

Basic Flow:

Precondition - page node must be selected in structural view and therefore visible in detail view.

1. Select Insert menu.
2. Choose simple image.
3. Select "from catalogue".
4. Select the image wanted from the catalogue.
5. Press Insert.

Alternate Flow 1:

Precondition - page node must be selected in structural view and therefore visible in detail view.

2. Create new Simple Image

Create New Simple Image

Basic Flow:

Precondition - a page node is selected in structural view and therefore visible in detail view.

1. Select the Insert menu.
2. Choose simple image.
3. Choose new (as opposed to from catalogue).
4. Choose between the standard shapes available (circle, rectangle, line, arrow) and choose a colour.
5. Press Insert (the window doesn't close so that the author can easily select multiple shapes if they want).

6. The image appears on the page.
7. Author adjusts size and position and can set the order of objects on the page if they want.

Alternate Flow 1:

1. Right click on the page node in structural view.
2. Choose add simple image.
3. Choose new (as opposed to from catalogue).
4. Choose between the standard shapes available (circle, rectangle, line, arrow) and choose a colour.
5. Press Insert (the window doesn't close so that the author can easily select multiple shapes if they want).
6. The image appears on the page.
7. Author adjusts size and position and can set the order of objects on the page if they want.

<ins>Add Background Image to Page</ins>

Basic Flow:

Precondition - there is no current background image on the page and the image has already been imported into the content catalogue.

1. Select a page in structural view so that it shows in detail view.
2. Choose the Insert menu and select background image.
3. Select the image wanted from the content catalogue.
4. The image will be shrunk or stretched to fit exactly into the size of the page.

Alternate Flow 1:

1. Select a page in structural view so that it shows in detail view.
2. Press the insert background image toolbar button in the detail view toolbar.
3. Select the image wanted from the content catalogue.
4. The image will be shrunk or stretched to fit exactly into the size of the page.

Alternate Flow 2:

1. Select a page in structural view so that it shows in detail view.
2. Choose the Insert menu and select background image.
3. Select the image wanted from the content catalogue.

4. The existing background image is removed.
5. The new image will be shrunk or stretched to fit exactly into the size of the page.

Add Text to Page

Basic Flow:

Precondition - page node must be selected in structural view and therefore visible in detail view.

1. Select Insert menu.
2. Choose text.
3. Select "new".
4. An empty text box appears on the page.
5. Type desired text into text box.

Alternate Flow 1:

1. Select Insert menu.
2. Choose text.
3. Select "from catalogue".
4. Select the text wanted from the catalogue.
5. Press Insert.

Set Page Properties

Basic Flow:

Precondition - page is visible in detail view.

1. Select the control of the property to change.
2. Enter the new value.
3. Press enter or click elsewhere on the interface.

Alternate Flow 1:

1. Right click on the page node in structural view.
2. Select properties.
3. Enter the new value of any property.
4. Press OK.

Design Summary Usage Information

Basic Flow:

Precondition - the support package is open.

1. Select the edit menu.
2. Select summary usage info.
3. The Summary Usage Information Designer window appears.
4. Make changes from the defaults if new or from what is already there is updating.
5. Press save changes to save or cancel.
6. The window will close and this new summary info will appear in the detail view when the support package or course node in structural view is highlighted.

Add Animation to Page

Basic Flow:

Precondition - page must be selected in structural view and therefore visible in detail view.

1. Select Insert menu.
2. Select animation. The Animation Designer window opens.
3. <Use Case> create new animation.
4. Select OK or cancel.

Create New Animation

Basic Flow:

1. Choose to add an image to the animation. This uses the View Content Catalogue use case.
2. Select an image from the catalogue.
3. Repeat steps 1 and 2 as often as needed to include all desired images.
4. Drag and drop images to achieve desired sequence. Default sequence is the order the images were added.
5. Add trigger to each image in the sequence. This could be either a timer e.g. 5 seconds after previous image, or a user action e.g. click on page in certain place, right click, double click etc.

Edit Text

Basic Flow:

(For editing the content of the text itself)

1. Select the text box in detail view of the page.
2. Edit text.
3. Click elsewhere on the interface.

Alternate Flow 1:
(For editing the properties of the text, not the text itself)

1. Select the text box in detail view of the page.
2. Change any of the values in detail view using the Text Options controls e.g. font, style, size, colour.
3. Click elsewhere on the interface.

Edit Image
Basic Flow:
(For a simple image created using this program)

1. Select the image in detail view of the page.
2. Change any of the values in detail view using the Image Options controls e.g. size (width and height), colour.
3. Click elsewhere on the interface.

Alternate Flow 1:
(For a simple image created using this program or an imported image)

1. Select the image in detail view of the page.
2. Drag the image to a new position or stretch an edge or corner of the image to change its size.

Edit Animation
Basic Flow:

1. Select edit animation from the detail view Animation Options. This opens the Animation Designer with the current animation opened in it.
2. Make any desired changes.
3. Press OK or cancel.

Switch View Between Page Layers

Basic Flow:

Precondition - a page must be highlighted in structural view and visible in detail view.

1. From the list of layers in the Page Properties section of the detail view, select the layer to view.

Re-sequence Page Layers

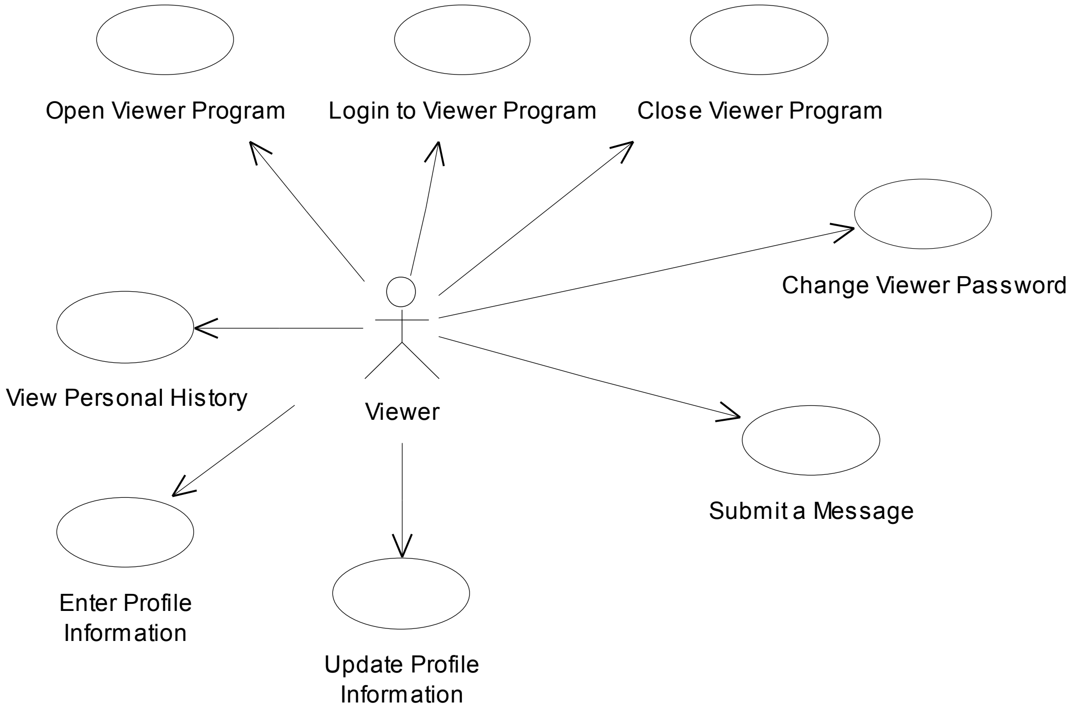
Basic Flow:

Precondition - a page must be highlighted in structural view and visible in detail view.

1. From the list of layers, select the layer to move by clicking down on it.
2. Drag the layer to the desired position in the list.
3. Drop the layer in its new position.
4. The layers would subsequently be renumbered.

Viewer Use Cases

Viewer Administrative Task Use Cases



Login to Viewer Program

Basic Flow:

Precondition - program has just been opened.

1. Enter user name.
2. Enter password.
3. Press enter or click login.

Close Viewer Program

Basic Flow:

1. Press exit
2. Answer yes to prompt to exit.
3. The application and the currently open support package closes.

Alternate Flow 1:

1. Press exit
2. Answer yes to prompt to exit.
3. The application remains open.

Change Viewer Password

Basic Flow:

Precondition - user must already be logged into the program.

1. Go to the personal information section.

3. Enter current password.

4. Enter new password twice.

5. Press OK (or cancel).

View Personal History

Basic Flow:

Precondition - already logged into the viewing program.

1. Open the course that you want the history for. This calls the Open a Course use case.

2. Select Personal History. This would replace the lesson menu of the course as the content of the main window. Personal History contains both lessons completed and test results.

Submit a Message

Basic Flow:

Precondition - user is on the last page of a lesson.

1. Select "submit a comment or question about this lesson".
2. Enter the message text.
3. Press submit or cancel.

Alternate Flow 1:

Precondition - user has just completed the last component of a course (not necessarily in order).

1. Choose to "submit a comment or question about this course" when prompted.
2. Enter the message text.
3. Press submit or cancel.

Alternate Flow 2:

Precondition - user is on the main course selection window or the main lesson selection window of a course.

1. Select "contact support staff".
2. Enter the message text.
3. Press submit or cancel.

Enter Profile Information

Basic Flow:

Precondition - it is the user's first time using the application.

1. Agree with the dialog prompting for profile information.
2. Enter values for all of the variables given.
3. Press OK or cancel.

Alternate Flow 1:

Precondition - not first time using the application (no dialog prompt is open).

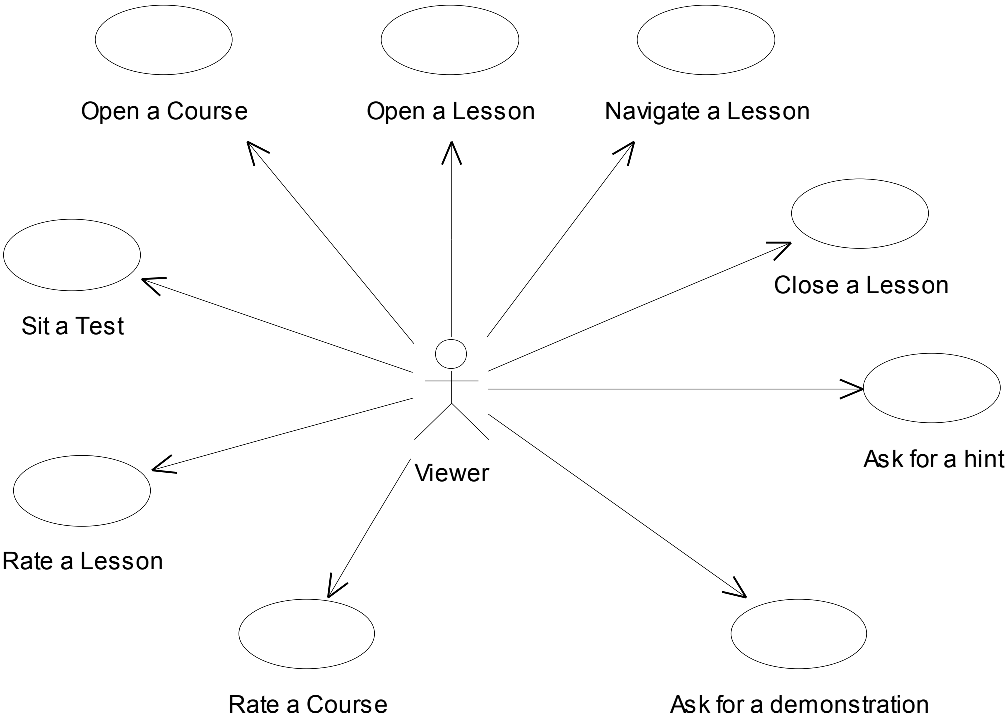
1. See Update Profile Information use case.

Update Profile Information

Basic Flow:

1. Select Personal Information.
2. Choose to enter profile information.
3. Enter values for all of the variables given.
4. Press OK or cancel.

Viewer Support Viewing Task Use Cases



Open a Course

Basic Flow:

Precondition - user is on the initial page - the software support package selection page.

1. From the list of software that the user is registered for, select the desired title.
2. The list of software will be replaced by the menu for that support package e.g. course, help, docs etc.

3. Select the course component of the package from the menu.
4. The package menu will be replaced by the course menu e.g. lessons.

Open a Lesson

Basic Flow:

Precondition - the user has already opened a course.

1. Select a lesson. Lessons do not have to be completed in order and can be run any number of times. The lesson will appear in a separate window and the main window will be minimised.

Navigate a Lesson

Basic Flow:

Precondition - a lesson is open.

1. Read the content on the current page.
2. Choose "next page".

Alternate Flow 1:

Precondition - the current page contains an animated demonstration or an audio component.

1. Read the content on the current page.
2. Choose "replay".

Alternate Flow 2:

Precondition - the current page is the last in the lesson.

1. Read the content on the current page.
2. Choose "exit lesson".

Close a Lesson

Basic Flow:

Precondition - a lesson is currently open.

1. Press the close window button or select Close Lesson.
2. Answer yes to prompt to exit lesson.

3. If the lesson was completed, this is reflected on the main window by a change in colour of the lesson title.

Sit a Test

Basic Flow:

Precondition - the course is open and the lesson menu is visible.

1. Select a test from the menu.
2. Answer each question.
3. Submit the test.

Alternate Flow 1:

Precondition - the user is at the personal history section.

1. Select the test from user's personal history (e.g. one that has already been sat).
2. Choose "re-sit".
3. Answer each question.
4. Submit the test.

Alternate Flow 2:

Precondition - user reaches a test page at the end of a lesson.

1. Answer each question.
2. Submit the test.

Rate a Lesson

Basic Flow:

Precondition - the user has just finished a lesson.

1. Choose "rate this lesson".
2. Choose between the offered ratings.
3. Choose submit or cancel.

Rate a Course

Basic Flow:

Precondition - user has just completed the last component of a course (not necessarily in order).

1. Choose to "rate this course" when prompted.
2. Choose between the offered ratings.
3. Choose submit or cancel.

Ask for a Hint

Basic Flow:

Precondition - user is on a demonstration page.

1. Select "give me a hint".
2. Press OK to close the hint.

Ask for a Demonstration

Basic Flow:

Precondition - the user must be on a demonstration page.

1. Choose "show me".
2. Press "play".
3. Close the demonstration.

Table of Contents

List of Figures