User-Centered Design and User Interface Prototyping
Conceptual Design and High Fidelity Prototyping

Project

Title	***Space Shooter Game Map Editor***
Project Web Page	`http://whitedwarf.sf.net/`

Team Members and Roles

Name	**ID**	**Email**	**Role**
Christopher Jensen	5936012	jensen@cs.concordia.ca	Project Manager
Etienne Clement	5924561	clement@cs.concordia.ca	Usability Engineer
Benoit Nadeau	5923352	nadeau@cs.concordia.ca	Direct User
Gaspard Petit	5934877	petit@cs.concordia.ca	Software Developer

Project Deliverables

Deliverable	**Date**	**Comments**
Project Proposal	02/01/02	
Tasks Analysis, User Requirements and Low-Fidelity Prototyping	02/25/02	
Conceptual Design and High Fidelity Prototyping	04/02/02	
Usability Testing and Final Recommendations		

Space Shooter Map Editor

Table of Contents

List of Tables

List of Figures

Project Proposal

Space Shooter Map Editor

1 Project Summary

A Cooperative Multiplayer Sidescrolling Space Shooter game is currently being designed and implemented by our team for the SOEN 390 project. Our project for this course will consist of creating an editor for the game. This editor will allow users to create their own scenarios by editing the map, units, and even sound effects.

The editor is intended to make the game more pleasant for the players since they will be able to create custom scenarios that can be exchanged from system to system. The scenarios can be created with any degree of difficulty to suit the needs of any player in order to reach a broader audience. Furthermore, the editor permits the extension of the game to the user's content and thus provides unlimited gameplay.

The editor has two main uses. The developers will see it as a tool to build levels to test and deliver the game with, while the end-users will see it as a way to expand the game once they are bored with the levels provided to them. Thus, the interface will have to suit the needs of the developers of the game who can be considered as expert users, as well as those of the end-users who may not understand the details and subtilities of the game.

It will be possible to achieve a variety of tasks using the editor. The features that will be part of the editor are listed below by priority in a decreasing manner and will be implemented as time permits.

- Intuitive paint-style interface for creating maps.
- Edit unit statistics, abilities, and upgrades.
- Create scenarios including specific maps and units.
- Customize sound effects that are associated with specific events.

The game is only intended to entertain its users in a purely recreational context. Therefore, the success of the game can only be measured by the interest it generates and by its popularity.

The user interface of the editor will be designed in a way to encourage instinctive learning of the software. Therefore, a user should be able to learn and use the basic features of the system with minimal or no help. Detailed help files will be available for the advanced features of the editor in order to make sure that the users can work with the software to its full capacity.

Since the editor is an inherent part of the game, it will be installed as part of the game installation. During the installation process the users will be informed of its existence.

As the editor is built, some main scenarios will be scripted to make sure the editor does not regress. Obviously, there is no assurance that there will be enough time to implement everything within this short time frame. The tools will have to be organized such that the user does not feel that something is missing and giving the chance to add new tools in the future without completely redesigning the GUI.

2 Problem

Trying to balance flexibility and usability will pose a challenge when designing the map editor and the configuration menus for the game. The game will try to please a large audience. If the game is too simple then avid users will get bored quickly. On the other hand, if it is complicated to start with, then players will get discouraged quickly and will not investigate further the game. Since the game will contain some advanced features that will require some fine tuning, a clever design will be needed to incorporate them smoothly.

A solution that some software designers provide today is a dual interface, one for beginners and one for experts. These modes can be toggled easily from one to the other.

3 Analysis

Having the dual interface will permit beginners to have the suggested defaults to some of the more advanced options. This will give the user the choice of customizing his environment when he or she feels ready.

We were unable to find any editor for a Sidescrolling Shooter available to the public. Most game editors are made specifically for the developers that are building the game and never meant to be made public.

For a good example of the dual user interface, we can look at Mirabilis ICQ. The software gives two interfaces. The first one is sufficient enough to get the gist of the software, but when the user will feel comfortable with it and decide to take advantage of its full potential; a simple click on the advanced button changes the whole layout of the software.

4 Suggested Improvement

Our goal here is to make an editor that non-programmers can also easily use. While you have to be a programmer to do things like making a new AI, everything else can easily be added or modified from a very intuitive User Interface.

To do that without having to force non-programmers to learn too much technical details that they do not need to know, the interface's "dual mode", in its mode for beginners, will offer default values for non-intuitive settings. Furthermore, instead of forcing the user to enter numerical values, the UI will tend to use other graphical widgets that are more intuitive.

Also, unlike "in-house" editors, configuring the editor to make it use the data of the game must be easy and flexible, not with any kind of "hard-coded" assumptions about the computer that uses the editor (except the minimum requirements, of course). Thus, installation will be as simple as possible and can be done by the user himself.

The editor can be customized a little, for example choosing the mode of the "dual mode" editor, moving the different utility windows, and so on.

User Interface Requirements Portfolio

Space Shooter Map Editor

1 Introduction

This Space Shooter Map Editor[1] (SSME) Usability Requirements and Task Analysis Document (URTAD) document defines and describes the operations, interfaces, performance and quality assurance requirements of the SSME Software. The requirements described in this document are derived from the project proposal and inputs from the various stakeholders.

1.1 Definitions, Acronyms and Abbreviations

The following is a list of definitions, acronyms and abbreviations that will facilitate the understanding of the document.

1.1.1 Definitions

Actor	Actor are a object displayed on the screen (e.g. a ship or a bullet). Actor have type, initial energy level, weapon and state definitions. During the game, at run time, they will be assigned a dynamic position and amount of energy.
AI	Controls the behavior of all the actors in a formation.
Chapter	A playable section of the game. You go through a Chapter by flying your ship through the Chapter while avoiding being hit. If your ship explodes, you start again at the beginning of the current Chapter.
Episode	Collection of chapters. It is presented to the Gamer as one ”Level”: at the beginning the name of the level is presented; at the end the ship exits the screen to go to the next level. The transitions between the chapters are continuous to the eyes of the Gamer, even if some things change like the scrolling speed, the music, the background image, and so on.
Formation	Group of actors that can be controlled by a single AI. It is often seen in the game as a fleet of ships moving together or as a “boss” with multiple body parts.
Gamer	Someone who plays video games.
Game Engine	The game engine is the creature that sits inside a game and controls the objects within the game. It is usually closely linked with the AI and, of course, the map system.
Map	The map of a game is the combination of background images that last throughout a Chapter.
Media	Sounds, images, animations, and so on.
Scenario	Collection of maps, actors, AI and different media. Scenarios may contain several episodes.
White Dwarf	The tentative name of the game for which the game editor will be used with.

1.1.2 Acronyms

AI	Artificial Intelligence
FAQ	Frequently Asked Questions
SSME	Space Shooter Map Editor
GUI	Graphical User Interface
UI	User Interface

1.1.3 Abbreviations

N/A	Not Applicable
s/he	He/She

[1]The SSME is part of the *White Dwarf*, a multiplayer sidescrolling game.

2 Users, Stakeholders and Main Roles

The intended audience for the software is split between the gamers who will buy the game and wish to extend its functionalities using the editor, and the developers of the game that will use it to create the initial scenarios that are shipped with the latter.

As a matter of fact, this will enable the users to have a very powerfull editor, capable of customizing almost every aspect of the game. Moreover, the editor will be constructed in such way that even the non experienced users will be able to have fun creating their own additions to the game.

2.1 Main Users

Essentially the player is the user who will buy the game. S/he should not have any knowledge of the implementation details of software nor should s/he need to be experienced with other game editors.

ID	Name	Main Task Goals
U1	Player	◇ Avid players who get bored with the predefined scenarios will be encouraged to use the game map editor to create their own scenarios. A future web site will be setup so that users will be able to share their creations. The editor should not limit the player's creativity. ◇ Players wishing to customize existing scenarios to the game should be able to do so with ease. For example, changing the music of an episode or adding extra weapons should be effortless.

2.2 Other Users

The developer of the game will build an editor, first and foremost, to be able to create the main scenarios of the game itself. This will permit early feedback about the editor, because the developer will be using it frequently to build the game.

ID	Name	Main Task Goals
U2	Developer	◇ Create exciting game scenarios quickly. ◇ Take full advantage of the game's capacities and features. ◇ Have all the scenarios stored in an orderly fashion. ◇ Make sure that the scenarios are consistent and are usable with the game engine.

3 User Characteristics

Table 1: Player Characteristics

U1	Player	
Characteristics	Potential System Requirements	Ref.
Skill And Knowledge		
Training and experience in the processes and methods which the system supports.		
◇ *Little to no knowledge in programming.*	◇ The GUI must be presented with intuitive concepts for non-programmers.	3.1
◇ *Familiar with the White Dwarf game.*	◇ The UI of the SSME must be representative of White Dwarf.	3.2
Experience in:		
a) Using the current system *Will be the first time user.*	Use simple language and concepts.	3.3
b) Using other systems with similar main functions *Most gamers have seen a map editor before. The system should also take into account existing map editors so to not alienate the existing gaming community.*	Although the editor will be inspired by existing ones, it should provide an improved interface compared to existent map editor.	3.4
c) Using systems with the same interface style or operating system *The platform of the game will be Windows. The game should use toolbars, menus and buttons that can be seen in many Windows specific softwares.*	SSME should use the Microsoft Windows SDK, in order to to reuse the familiar menus and buttons.	3.5
Knowledge or training in:		
a) Tasks supported by the system main functions *Create a scenarios for White Dwarf game.*	The interface should reflect the organization of a scenario.	3.6
b) Using the systems main functions *Knowledge of the White Dwarf game required. Learning the SSME should be instinctive.*	The SSME should have an UI that can be learned without any formal training.	3.7

Continued on next page...

Characteristics	Potential System Requirements	Ref.
c) Using other systems with similar main functions *None.* **d) Using systems with the same interface style or operating system.** *Assuming familiarity with Windows.*		
Education/Qualification *None required.*		
Relevant input skills *Imagination and creativity.*	The UI interface should not restrict creativity.	3.8
Linguistic ability *Basic English.*	The SSME and its documentation should be in English.	3.9
Background knowledge/IT Knowledge *Basic computer knowledge.*		
PHYSICAL ATTRIBUTES		
Age Range *12 to 99 years old.*		
Typical Age *16 years old.*		
Gender *95% male, 5% female.*		
Physical attributes, limitations and disabilities *No limitation.*		
MENTAL ATTRIBUTES		
Intellectual abilities **a) Distinctive abilities** *Easily understands concepts represented visually.* **b) Specific mental disabilities** *None.*	System should provide an instinctive UI.	3.10

Continued on next page. . .

Characteristics	Potential System Requirements	Ref.
Motivations **a) Attitude to job and task** *Positive attitude.* **b) Attitude to the system** *Positive attitude.* **c) Attitude to information technology** *Not Relevant.* **d) Employees attitude to the employing organization** *Not Relevant.*	Target novice and normal users.	3.11
JOB CHARACTERISTICS		
Job function *Purely recreational.*		
Job history **a) How long employed** *Not Relevant.* **b) How long in current job** *Not Relevant.*		
Hours of work/operation **a) Hours of work** *Not Relevant.* **b) Hour using the system** *2 hours per day.*		
Job flexibility *Not Relevant.*		
Frequency of use *Used frequently, can become addictive.*		
Discretion to use *Can ignore system or abandon it for any reason.*		
Other relevant features *None.*		

Table 2: Developer Characteristics

U2	Developer	
Characteristics	Potential System Requirements	Ref.
SKILL AND KNOWLEDGE		
Training and experience in the processes and methods which the system supports.		
◇ *Familiar with the White Dwarf game internals.*	◇ The GUI must be representative of the features supported by the White Dwarf game.	3.12
◇ *Familiar with the SSME internals.*	◇ The UI of the SSME must allow the use of all the features internally supported by the SSME.	3.13
Experience in:		
a) Using the current system *Considerable experience, participated in the developement of the White Dwarf game and the SSME.*	The UI of the SSME should provide an easy access to advanced features.	3.14
b) Using other systems with similar main functions *Used other map editors available for comparison purpose.*		
c) Using systems with the same interface style or operating system *Familiar with other map editors available.*	Must provide an improved interface compared to existing map editors.	3.15
Knowledge or training in:		
a) Tasks supported by the system main functions *Create a scenarios for White Dwarf game.*	The interface should reflect the organization of a scenario.	3.16
b) Using the systems main functions *Knowledge of the White Dwarf game and SSME acquired during development.*		
c) Using other systems with similar main functions		

Continued on next page...

Characteristics	Potential System Requirements	Ref.
Required. **d) Using systems with the same interface style or operating system.** *Required.*		
Education/Qualification ◇ *Programming knowledge.* ◇ *Familiarity with Space Side Scroller.*		
Relevant input skills *Ability to work in a team.*		
Linguistic ability *Strong communication skills.*	◇ The SSME and its documentation should be in English. ◇ Strong oral and written English skills.	3.17 3.18
Backgroung knowledge/IT Knowledge *Software development environment and equipment.*		
Physical Attributes		
Age Range *20 to 40 years old.*		
Typical Age *25 years old.*		
Gender *99% male, 1% female.*		
Physical attributes, limitations and disabilities *No limitation.*		
Mental Attributes		
Intellectual abilities **a) Distinctive abilities** *Research perfection.* **b) Specific mental disabilities** *None.*	The UI should be aesthetically and functionnally perfect.	3.19

Continued on next page...

Characteristics	Potential System Requirements	Ref.
Motivations		
a) Attitude to job and task ◇ *Enthusiast about software development.* ◇ *Interested in computer games.*		
b) Attitude to the system *High degree of interest.*	Target experienced users.	3.20
c) Attitude to information technology *Cooperative.*		
d) Employees attitude to the employing organisation *Devoted.*		
JOB CHARACTERISTICS		
Job function ◇ *Design and implement a space shooter game.* ◇ *Design and implement a SSME.*		
Job history		
a) How long employed *From 1 to 10 years.*		
b) How long in current job *From 1 to 5 years.*		
Hours of work/operation		
a) Hours of work *From 8 to 12 hours a day.*		
b) Hour using the system *3 hours per day.*		
Job flexibility *Relativly flexible.*		
Frequency of use *Daily basis.*		

Continued on next page...

Characteristics	Potential System Requirements	Ref.
Discretion to use *Required.*		
Other relevant features *None.*		

4 Task List, Task Characteristics and Task Flow Diagrams

4.1 Task List

Even though the tasks are assigned to the Player User or the Developer User, one may perform the tasks of the other – they have simply been separated based on the likelyhood of one performing the given task.

Table 3: Player Task List

U1	Player		
Characteristics		Potential System Requirements	Ref.
T101	Build a Scenario Using the Wizard.	The Scenario Wizard should be displayed by default when the editor is open without any document. It should guide the user into creating many episodes using the Episode Wizard.	4.1
T102	Add an Episode using the Wizard	The Episode Wizard should guide the user step by step into creating an episode, adding chapters and changing their settings. It should also allow the user to add formations to the chapters and create them using the Formation Wizard if necessary as well as players.	4.2
T103	Add a Formation Using the Wizard.	The Formation Wizard should guide the user into creating a new formation. The user should be able to add actors to the formation, and to create them using the the Actor Wizard if necessary.	
T104	Add an Actor Using the Wizard	The Actor Wizard should guide the user into creating a new actor and should allow him to create new states to the actor.	4.3
T105	Import an Image.	Names associated to imported images should be unique	4.4
T106	Import a Sound.	Names associated to imported sounds should be unique	4.5
T107	Add a Player Formation Using the Wizard.	The Player Wizard should guide the user into creating new player formations allowing them to add actors and create them using the Actor Editor if necessary.	4.6

Table 4: Player Task List

U2	Developer		
Characteristics		Potential System Requirements	Ref.
T201	Create a New Scenario		
T202	View a Scenario		
T203	Add a New Episode	It should be possible to add any number of episodes to a scenario	4.7
T204	Remove an Episode		
T205	View an Episode		
T206	Change the Episode Order		

Continued on next page...

Characteristics		Potential System Requirements	Ref.
T207	Add a Chapter	It should be possible to add any number of chapters to an episode	4.8
T208	Remove a Chapter		
T209	View a Chapter		
T210	Change the Chapter's Order		
T211	Change a Chapter's Music	The music should be chosen from the list of imported sounds.	4.9
T212	Change a Chapter's Background	The background should be chosen from the list of imported images.	4.10
T213	Change a Chapter's Scrolling Speed	Scrolling speed should be between -100.0 and 100.0, 1.0 being the default value	4.11
T214	Add a Formation to a Chapter	It should be possible to add any number of formations to a chapter. Formation should be chosen from the list of formations.	4.12
T215	Remove a Formation from a Chapter		
T216	Move a Formation in a Chapter		
T217	Add a New Actor	It should be possible to add any number of actors to the scenario	4.13
T218	Remove an Actor		
T219	View an Actor		
T220	Change an Actor's name	The name of an actor should be unique	4.14
T221	Change an Actor's Default Image	The image should be from the list of imported images	4.15
T222	Change an Actor's Type		
T223	Change an Actor's Weapon	The weapon should be from the list of formations	4.16
T224	Change an Actor's Item	The item should be from the list of items	4.17
T225	Add a State to an Actor	It should be possible to add any number of states to an actor	4.18
T226	Remove a State to an Actor		
T227	View the State of an Actor		
T228	Change the Image of a State	The image should be from the list of imported images	4.19
T229	Change the Sound of a State	The sound should be from the list of imported sounds.	4.20
T230	Change the Name of a State	The name of the state should be unique	4.21
T231	Add a New Formation	It should be possible to add any number of formations	4.22
T232	Remove a Formation		
T233	View a Formation		
T234	Change the Type of a Formation		
T235	Change the Name of a Formation	The name of a formation should be unique	4.23
T236	Add an Actor to a Formation	It should be possible to add any number of actors to a formation. The actors should be from the list of actors.	4.24
T237	Remove an Actor from a Formation		
T238	Change the Role of an Actor in a Formation		

Continued on next page...

Characteristics		Potential System Requirements	Ref.
T239	Change the Item of a Formation	The item should be from the list of items	4.25
T240	Remove an Image		
T241	Change the Name of an Image	The name of the image should be unique	4.26
T242	Remove a Sound		
T243	Change the Name of a Sound	The name of the sound should be unique	4.27
T244	Add a New Player Formation	It should be possible to have any number of player formations in a scenario	4.28
T245	Remove a Player Formation		
T246	View a Player Formation		
T247	Change the Type of a Player		
T248	Add an Actor to a Player	It should be possible to add any nunber of actors to a player formation.	4.29
T249	Remove an Actor from a Player Formation		
T250	Change the Role of an Actor in a Player Formation		

4.2 Task Description and Task Flow Diagrams

Table 5: Build a Scenario Using the Wizard Task Description

T101	Build a Scenario Using the Wizard		
Characteristics		**Potential System Requirements**	**Ref.**
TASK GOAL			
Create a new scenario using the scenario wizard			
WHEN PERFORMED			
After the SSME launched			
TASK INPUTS OR DEPENDENCIES			
◇ *Name of the Scenario*		The name and saving location must be unique	4.30
◇ *Number of Episodes*			
◇ *Number of Chapters per Episode*			
◇ *Actors in the Scenario*			
◇ *Players in the Scenario*			
◇ *Formations in the Scenario*			
◇ *Position of the Formations*		Position in the chapters should be given with (x, y) coordinates, with $x > 0$ and $0 < y < 1.0$	4.31
◇ *Scrolling Speed of the Chapters*			
◇ *Music of the Chapters*		The music should be from the list of imported sounds	4.32

Continued on next page...

Characteristics	Potential System Requirements	Ref.
◇ *Background images of the Chapters*	The images should be from the list of imported images	4.33
Task Output		
A Scenario is created and saved.		
Task Variability		
The user may have different number of episode,chapters per episode, actors, formations and players. Most settings have default values and do not necessarily have to be modified.		
Task Frequency		
Frequent		
Task Duration		
About 10 to 30 minutes		
Task Constraints/Pacing		
None.		
Task Flexibility		
The wizard presents dialog panes to the user. He may at any time go to the previous pane.	The user should be able to go to the previous step at any time	4.34
Physical and Mental Demands		
This task is relatively easy since the user is given explanations for each step. The user needs to have an idea of what kind of scenario s/he wants before doing this task.	The task is intended for unexperiences users and should be intuitive.	4.35
Linked Task		
◇ *T102: Add an Episode using the Wizard* ◇ *T103: Add a Formation Using the Wizard* ◇ *T104: Add an Actor Using the Wizard* ◇ *T105: Import an Image* ◇ *T106: Import a Sound* ◇ *T107: Add a Player Formation Using the Wizard*		
Safety		
N/A		
Task Criticality		
Required		

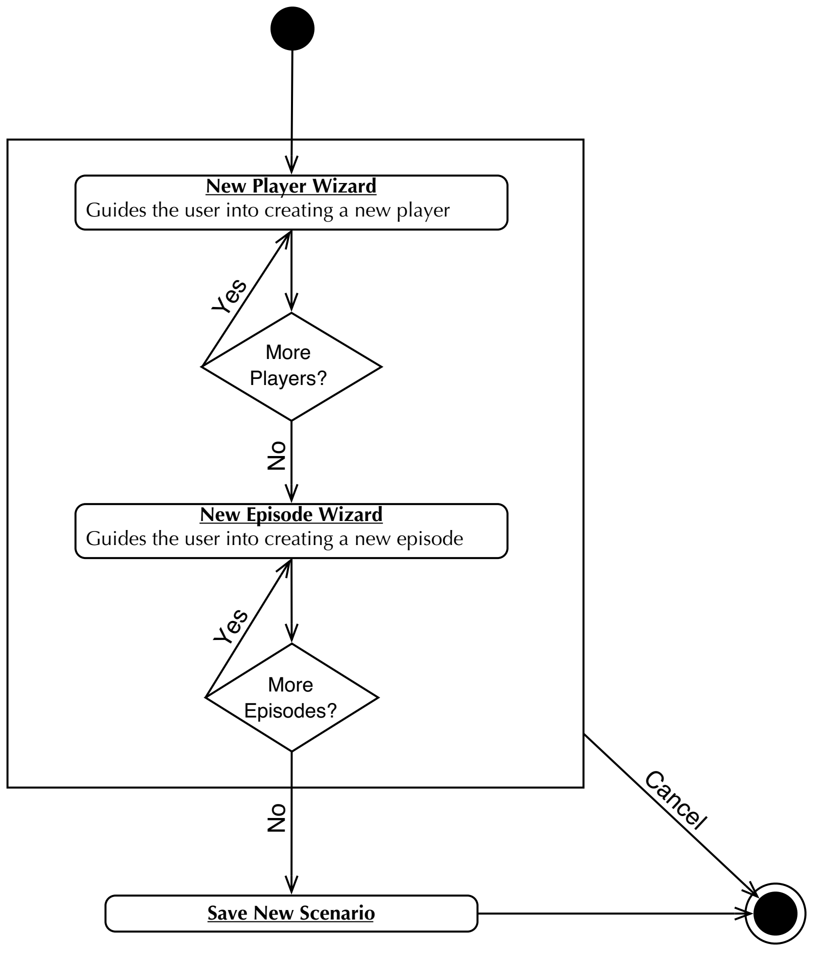


Figure 1: Task 101 State Diagram

Table 6: Add an Episode using the Wizard Task Description

T102	Add an Episode using the Wizard	
Characteristics	Potential System Requirements	Ref.
Task Goal		
Guide the user into creating and setting up a new episode.		
When Performed		
After a scenario has been open and the user wishes to create a new episode		
Task Inputs or Dependencies		
A scenario must be open. ◇ *The Number of Chapters for the Episode* ◇ *The Scroll Speed for each Chapter* ◇ *The Music for each Chapter* ◇ *The Background image for each chapter* ◇ *The Formations for the Episode* ◇ *The Position of the Formations in the Episode.*		
Task Output		
A new episode with the specified number of chapters is created with the values entered by the user.		
Task Variability		
The number of chapter and formation can vary. The user may not provide music or background for some chapters.		
Task Frequency		
Frequent		
Task Duration		
About 5 to 10 minutes		
Task Constraints/Pacing		
This task is either for beginers or user wishing to edit a level quickly. Thus, it should be accomplished quickly, as to not discourage beginers and make it useful even for advanced users.		
Task Flexibility		

Continued on next page...

Characteristics	Potential System Requirements	Ref.
Default values will be provided when possible. The user can choose to edit the episode and chapters after they have been created. At any time, the user can go back to the previous configuration step. The user may not use the name of an already existing episode.		
PHYSICAL AND MENTAL DEMANDS		
Since the user has to provide the setting for the entire level, this task requires some concentration. Also, if the user gets to a point where he does not understant what he is supposed to do and cancels, he will lose the entire level (even if he was at the last step). Thus a good description of what is expected from him is probably a good idea.	The task is intended for unexperiences users and should be intuitive.	4.36
LINKED TASK		
◇ *T103: Add a Formation Using the Wizard* ◇ *T104: Add an Actor Using the Wizard* ◇ *T105: Import an Image* ◇ *T106: Import a Sound*		
SAFETY		
N/A		
TASK CRITICALITY		
Required		

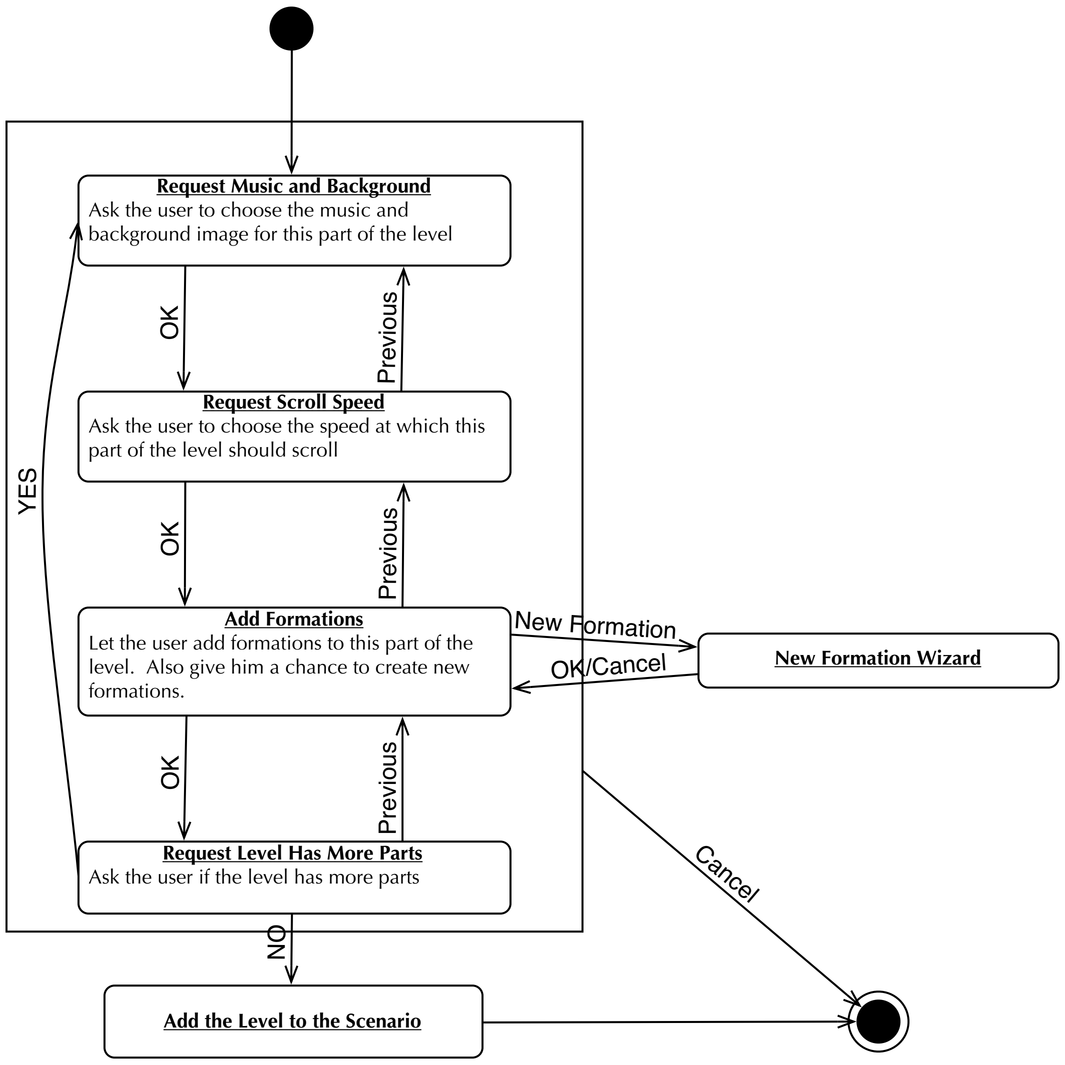

Request Music and Background
Ask the user to choose the music and background image for this part of the level
OK
Previous
Request Scroll Speed
Ask the user to choose the speed at which this part of the level should scroll
OK
Previous
Add Formations
Let the user add formations to this part of the level. Also give him a chance to create new formations.
New Formation
OK/Cancel
New Formation Wizard
OK
Previous
YES
Request Level Has More Parts
Ask the user if the level has more parts
NO
Cancel
Add the Level to the Scenario

Figure 2: Task 102 State Diagram

Table 7: Add a Formation Using the Wizard Task Description

T102	Add a Formation Using the Wizard		
Characteristics		Potential System Requirements	Ref.
TASK GOAL			
Guide the user into creating and setting up a new formation.			
WHEN PERFORMED			
After a scenario has been open and the user wishes to create a new formation for a chapter			
TASK INPUTS OR DEPENDENCIES			
A scenario must be open. ◇ *The Number of Actors in the Formation* ◇ *The Role of each Actor in the Fomation* ◇ *The Type of Formation* ◇ *The Name of the Formation*			
TASK OUTPUT			
A new formation with the specified number of actors is created with the values entered by the user.			
TASK VARIABILITY			
The number of actors can vary. The user may not provide values for all the fields.			
TASK FREQUENCY			
Frequent			
TASK DURATION			
About 3 to 5 minutes			
TASK CONSTRAINTS/PACING			
This task is either for beginers or user wishing to edit a level quickly. Thus, it should be accomplished quickly, as to not discourage beginers and make it useful even for advanced users.			
TASK FLEXIBILITY			
Default values will be provided when possible. The user can choose to edit the episode and chapters after they have been created. At any time, the user can go back to the previous configuration step. The user may not use the name of an already existing formation.			
PHYSICAL AND MENTAL DEMANDS			

Continued on next page...

Characteristics	Potential System Requirements	Ref.
Since the user has to provide the setting for the entire formation, this task requires some concentration. Also, if the user gets to a point where he does not understant what he is supposed to do and cancels, he will lose the entire formation (even if he was at the last step). Thus a good description of what is expected from him is probably a good idea.	The task is intended for unexperiences users and should be intuitive.	4.37
Linked Task		
◇ *T104: Add an Actor Using the Wizard*		
Safety		
N/A		
Task Criticality		
Required		

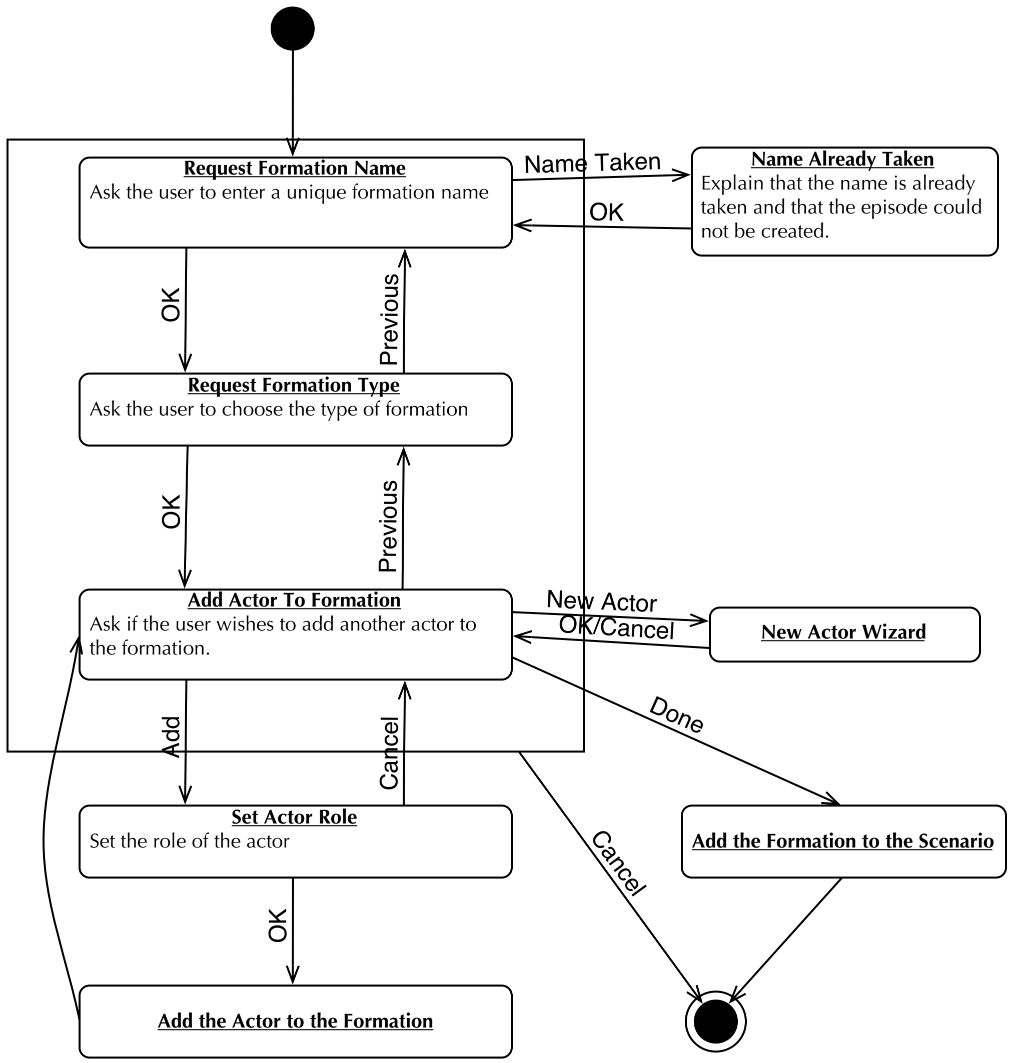

Request Formation Name
Ask the user to enter a unique formation name
Name Taken
OK
Name Already Taken
Explain that the name is already taken and that the episode could not be created.
OK
Previous
Request Formation Type
Ask the user to choose the type of formation
OK
Previous
Add Actor To Formation
Ask if the user wishes to add another actor to the formation.
New Actor
OK/Cancel
New Actor Wizard
Done
Add
Cancel
Cancel
Set Actor Role
Set the role of the actor
Add the Formation to the Scenario
OK
Add the Actor to the Formation

Figure 3: Task 103 State Diagram

Table 8: Add an Actor Using the Wizard Description

T104	Add an Actor Using the Wizard	
Characteristics	Potential System Requirements	Ref.
Task Goal		
Guide the user into creating and setting up a new actor.		
When Performed		
After a scenario has been open and the user wishes to create a new actor, or when the user is in the Build Formation Wizard task and needs to create a new actor.		
Task Inputs or Dependencies		
An scenario must be open.		
◇ *Name of the new actor*		
◇ *The type of the actor*	The user should be provided the types available	4.38
◇ *The Number of States of the Actor*	The basic types should have default vaules	4.39
◇ *The Image and the Sound for each State of the Actor*		
◇ *The Weapon of the Actor*		
◇ *The Item of the Actor*		
◇ *The amount of energy of the Actor*		
Task Output		
A new actor is created with the values entered by the user.		
Task Variability		
The number of states in the actor can vary.		
Task Frequency		
Frequent		
Task Duration		
About 2 to 5 minutes		
Task Constraints/Pacing		
This task is either for beginers or user wishing to edit an actor quickly. Thus, it should be accomplished quickly, as to not discourage beginers and make it useful even for advanced users.		
Task Flexibility		

Continued on next page...

Characteristics	Potential System Requirements	Ref.
Default values will be provided when possible. The user can choose to edit the actor after it has been created. At any time, the user can go back to the previous configuration step. The user may not use the name of an already existing actor.		
PHYSICAL AND MENTAL DEMANDS		
Since the user has to provide the setting for the entire actor, this task requires some concentration. Also, if the user gets to a point where he does not understant what he is supposed to do and cancels, he will lose the entire formation (even if he was at the last step). Thus a good description of what is expected from him is probably a good idea.	The task is intended for unexperiences users and should be intuitive.	4.40
LINKED TASK		
◇ *T105: Import an Image* ◇ *T106: Import a Sound*		
SAFETY		
N/A		
TASK CRITICALITY		
Recommended		

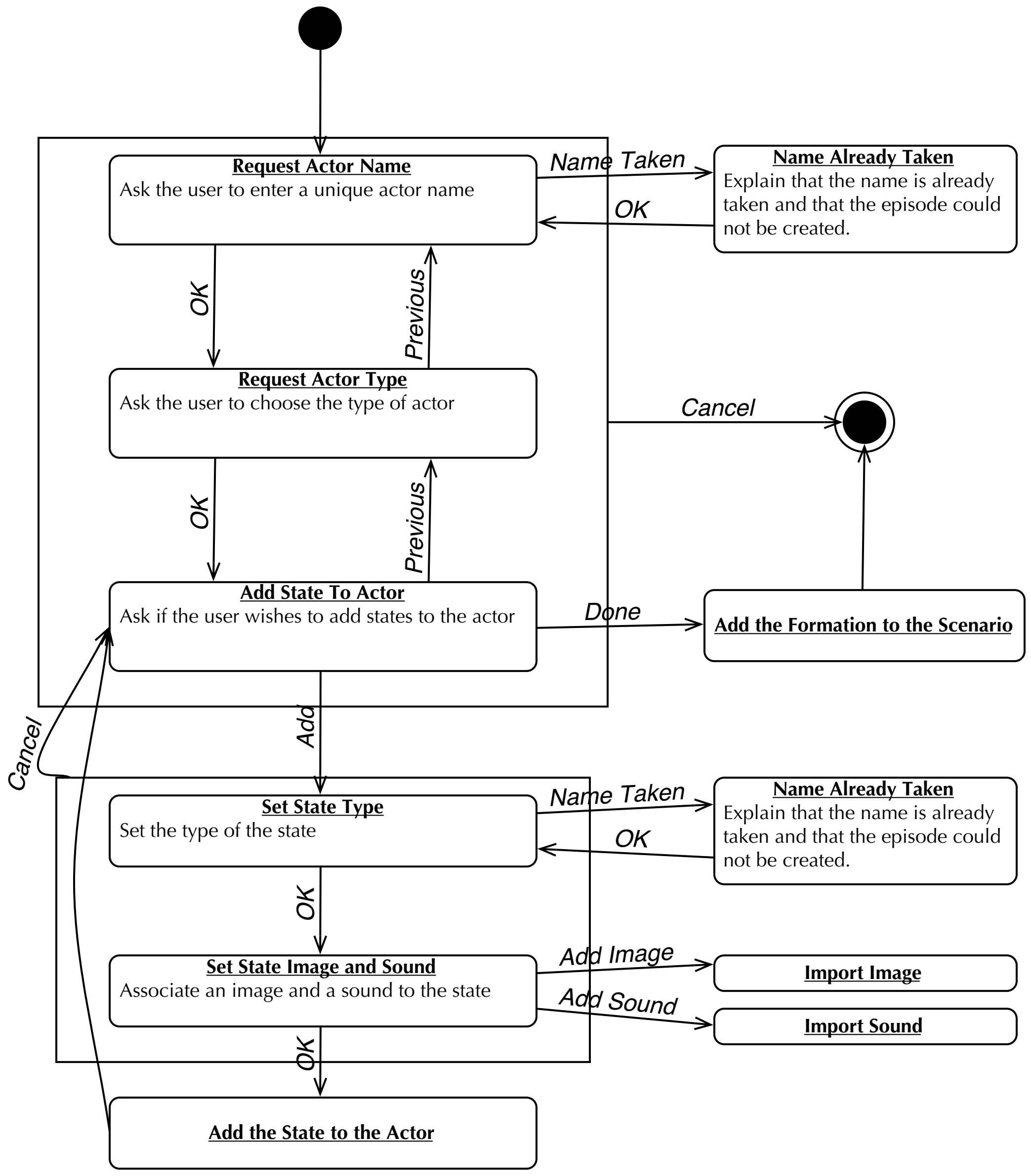

Request Actor Name
Ask the user to enter a unique actor name
Name Taken
Name Already Taken
Explain that the name is already taken and that the episode could not be created.
OK
OK
Previous
Request Actor Type
Ask the user to choose the type of actor
Cancel
OK
Previous
Add State To Actor
Ask if the user wishes to add states to the actor
Done
Add the Formation to the Scenario
Add
Cancel
Set State Type
Set the type of the state
Name Taken
Name Already Taken
Explain that the name is already taken and that the episode could not be created.
OK
OK
Set State Image and Sound
Associate an image and a sound to the state
Add Image
Import Image
Add Sound
Import Sound
OK
Add the State to the Actor

Figure 4: Task 104 State Diagram

5 Technical Characteristics and Constraints

Table 9: Technical Characteristics and Contrains

Characteristics	Potential System Requirements	Ref.
HARDWARE		
The computer selected must be able to run Windows 2000		
The computer must be equipped with a video card having a minimum of 32MB of RAM and supporting a minimum screen resolution of 800 by 600 pixels.		
The mouse and keyboard are used to navigate, select options and provide input information to the program.		
SOFTWARE		
The application has to run on Windows 2000 or any subsequent version.		
The application requires the OpenGL libraries.		
Space Shooter Map Editor software.	Software should be simple to operate and efficient in allowing users to design scenarios quickly.	5.1
REFERENCE MATERIALS		
The SSME help menu is intended to provide basic help.	The SSME should include a help menu.	5.2
The SSME will be accompanied by a user manual that is intended to provide complete help.	◇ Documentation should be clear enough in order to locate a topic easily and rapidly. Therefore, it must contain a table of contents, a list of figures, a list of tables and an index. ◇ Documentation should be available in LaTeX in order to be generated in many formats.	5.3 5.4

6 Physical Environment

Table 10: Physical Environment

Characteristics	Potential System Requirements	Ref.
Atmospheric Conditions		
Not Relevant.		
Auditory Environment		
Office or home sound environment.	The system should provide a mean to control the level of noise it generates.	6.1
Thermal Environment		
Not Relevant.		
Environment Instability		
Not Relevant.		
User Posture		
The user will be sitting while using the SSME.		
Space And Furniture		
Not Relevant.		
Location		
Office and home.		
Health And Safety Hazard		
Not Relevant.		
Protective Clothing And Equipment		
Not Relevant.		

7 Organizational Environment

Table 11: Organizational Environment

Characteristics	Potential System Requirements	Ref.
GROUP WORKING		
User is normally alone, sometimes with a partner.	Allow two users to view the SSME comfortably.	7.1
ASSISTACE REQUIRED OR AVAILABLE		
Possibly available over the Internet from FAQs and News groups.	◇ The SSME has to favor instinctive learning.	7.2
	◇ The SSME user manual has to be complete and unambiguous.	7.3
INTERRUPTION		
Can be caused by a power interruption, a computer crash or a fatal error that occurs within the application.	◇ The application should be able to save the current work within 10 secs.	7.4
	◇ The application should create backups intermittently.	7.5
MANAGEMENT STRUCTURE		
Not Relevant.		
COMMUNICATION STRUCTURE		
The different scenarios created will be distributed over the web.	◇ The physical structure of the scenerio must be compatible with on-line distribution.	7.6
	◇ The size of the scenario must be resonably small to permit reasonable download time.	7.7
IT POLICY		
Not Relevant.		
ORGANIZATIONAL AIMS		
Permits the creation of scenarios for White Dwarf.	The scenarios structure should be well documented to permit the game developers to use it as a base to develop the White Dwarf game.	7.8
INDUSTRIAL RELATIONS		
Not Relevant.		
PERFORMANCE MONITORING		
Not Relevant.		
PERFORMANCE FEEDBACK		
Not Relevant.		
PACING		

Continued on next page...

Table 11: Organizational Environment

Characteristics	Potential System Requirements	Ref.
Not Relevant.		
Safety And Security		
Some scenarios created with the SSME can be copyrighted and non-freely distributed.		
Privacy		
Not Relevant.		

8 Usability Factors Requirements

Priority	Description
1	Must have
2	Should have
3	Nice to have
N/A	Not Applicable

Table 12: Usability Factors Requirements

Usability Goal	User Requirement with Respect to Goal	Priority
Effectiveness		
Quality or quantity of task completion.	The system should help the users to create quality scenarios in a more efficient manner than with the manual alternative.	1
Efficiency		
Time to perform task, time compared with an expert.	◇ The system should provide a more efficient way of creating and editing scenarios than if it was done completly manually.	1
	◇ A normal user should become as efficient as an expert user within a period of one week.	2
Satisfaction		
Perceived satisfaction or enjoyment in using the system.	The satisfaction will be accomplished if the SSME is widely used within the gaming community. Popularity will be a direct measure of the degree of satisfaction.	1
Learnability		
Ability to use the system help or manuals to perform the task.	◇ The software will contain a help menu which will provide the first source of help. It will be useful to solve simple problems and answer general questions that the user might have.	1
	◇ The software documetation should be available in many formats to promote its consultation in many environments.	2
Intuitiveness		

Continued on next page...

Table 12: Usability Factors Requirements

Usability Goal	User Requirement with Respect to Goal	Priority
Ability to perform the tasks with limited introduction.	◇ Using the system main functions should be very intuitive. Anyone familiar with side scroller games should be able to use the basic functionalities of the SSME by interacting with it less than 30 minutes. ◇ Using the system advanced functions should be quite intuitive. Anyone familiar with map editors should be able to use those functionalities without requiring external help or training.	1 2
Helpful/Supportiveness		
Ability to overcome problems that arise.	Problems should be easily overcome by using the different help resources distributed with the software. The help menu, user manual and examples provided should be enough to solve most common problems.	1
Contrability		
Perceived feeling of being in control/tracking performance etc.	The system should fullfil the users needs. The users should have the feeling that the system is an useful tool that increases their performances as opposed to be a bottleneck to their inspiration and creativity.	2
Avoiding Excessive Mental Load		
Perceived mental effort, or physical indicators.	The only mental effort required to create scenerios using the SSME application should be the creativity and imagination that the user has to use to create interesting and challenging scenarios.	1
Avoiding Excessive Physical Load		
Heart rate, respiratory measurement.	Not Relevant.	N/A
Safety		
To be able to operate the system safely.	Not Relevant.	N/A

Conceptual Design and High Fidelity Prototype

Space Shooter Map Editor

1 Analysis of Previous Paper Prototypes

The various paper prototypes described in details the contents of the window, but by doing so we avoided several of the most important questions about our User Interface: How the different windows interact with each other? Should we use a "single-screen" system? A window with many frames, similar to HTML frames? A Multiple-Document Interface?

When you change one of the various game conceptual objects (Actor, Formation, Chapter), what happens to the other windows? Are they "updated", and if so, when? Since the data must follow some rules, for example some names must be unique in a scenario, when do we validate the data?

Also, there were little to no emphasis on the design of the various "Wizards" of the application. Should the Wizards be the primary focus of our user interface design?

As you can see, the previous paper prototypes put forth several distinct ideas, which had a similar result than a "brainstorming session", but the various ideas lacked of global, uniform design decisions, which will be described in detail in Section 2.

2 Chosen User Interface Design

The Space Shooter Map Editor for White Dwarf game is what we call a "dual-interface" application. It contains two separate yet equivalent set of User Interface elements, one for each user archetype. The "gamers" will mostly use a User Interface consisted only of "Wizards", while the game developers and more advanced users will use the Expert interface.

2.1 The "Wizard" Interface

The "Wizard" interface of the Space Shooter Map Editor is simply a series of dialog screens that takes the user through the process of creating a new scenario. At each step of the process, the user only needs to enter the requested information in the various User Interface elements before going to the next step.

Note that the "Wizard" interface is kept for the last development iterations, since the application is also made for SOEN 390 and testing of the core components of the application has a higher priority. Also, the Expert interface cannot be "faked" in any way and can only work if the entire application is functional, thus it is much more representative of the underlying work that had to be done for the development of this application. As a downside, it is much more difficult to assess to quality of the User Interface with the Expert interface, as it is much less "task-oriented" than the "Wizards".

Also, the "Wizard" has a pretty limited learning experience for the developers compared to the Expert interface, as it is much less complex and sophisticated to produce.

2.2 The Expert Interface

2.2.1 Multiple Document Interface

We chose to use the Multiple Document Interface (MDI) for the Expert interface of our application. This is because our application makes use of several independant windows that can interact with each other while still having some "global" palette windows that contain some shortcut buttons and the scenario tree.

An alternative would have been to use Palette Windows, but this interface concept is inexistent in Microsoft Windows, since windows are not logically grouped by application through Microsoft Windows's User Interface[2]. As a result, we properly use Microsoft's "one window / one application" concept to group all the windows and Palettes within another window, thus avoiding the problem altogether.

We avoided the use of framed windows, which is common in HTML-based application, since we had the problem that some simple dialog windows would be visually too large, and because MDI allows the user to quickly interact with several windows at the same time.

2.2.2 Screen Updates

Because the user knows the contents of the scenario only through the User Interface of our editor, it is extremely important that:

- The UI must be *very* clear about what is the value/contents of the data (the UI is not ambiguous).
- What is displayed on screen is effectively what will be stored on disk (What You See Is What You Get).
- Several UI elements refering to the same data effective display the same data *all the time*.

As a result, the UI elements must always work with the actual data, and never use any copy it has to make for the OS to display it on screen. Failure to do so could make the application's data seem to be, or efficitively be, corrupted.

While this may seem easy to do, this implies a rather complex database architecture that the UI must closely "listen" to. Also, we need to remain very consistent in the way the UI interacts with the database. This seems to be something "granted" for WYSIWYG applications, but when the data structure is complex, as it is the case for us, it isn't, and *must* be an explicit requirements for the developers.

2.2.3 Modal and Modeless Dialogs

One of the goals of the Expert interface is to not force, as much as possible, any kind of pre-determined task flow. As a result, most dialogs do not require to be Modal: they are simply Modeless, within the same "MDI space" as any other window of our application.

Some dialogs have to be Modal, since the application cannot continue to execute until the user makes its decision. This is the case of some "Add" buttons and the "Open Scenario" dialog. Otherwise, the dialogs are kept Modeless as much as possible.

As a result, the Modeless dialogs rarely require an "Apply" button[3]. Changes are immediatly made whenever possible. Obviously, this means that any action that can destroy data should be confirmed with a Modal Alert dialog box.

This decision was made because Modal dialogs, while clear, are both intrusive and, for advanced users, are counter-poductive.

Validation is made whenever possible during user input. Validation that is too complex (either for the computer or counter-productive for the user) is kept for when the file will be saved, at which time the exact source of the problem and a possible solution is clearly offered to the user.

[2]The logical grouping of the windows by application is effectively used in Mac OS; the Palette Windows hide themselves when you switch to another application. Both MDI and Palette Windows concepts are not supported in XWindows, which severly limits complex application development in Linux.

[3]And obviously, they never need an "OK" and "Cancel" button, implying that changes are often immediate and must be canceled by the "Undo" menu item.

3 Analysis of Chosen Design

3.1 Initial Screen

The user interface has been designed so that any task is achieved in a minimal number of steps. The MapEditor can be opened in two different ways:

3.1.1 Opening the MapEditor from a Scenario File

Assumed User Intention:	Scenario should be displayed
Next Anticipated Activity:	• Browse the scenario • Edit and change settings of episode, chapters, formations, actors, players, images or sound • Add an episode, chapter, formation, actor, player using the wizard • Add an episode, chapter, formation, actor, player, image, sound using the standard interface • Close the scenario
Approach Chosen:	Scenario is loaded and displayed with the advanced interface.

The scenario can be browsed using the tree browser. Items in the tree browser can be edited by double-clicking or right-clicking. Items can be added using the Scenario menu and the scenario can be close from the File menu.

3.1.2 Opening the MapEditor from the Executable

Assumed User Intention:	Create a New Scenario
Next Anticipated Activity:	• Use the wizard to create a new scenario • Use the advanced interface to create a new scenario • Open an existing scenario
Approach Chosen:	A dialog is displayed giving the user the choice between creating a new scenario with the Scenario wizard, creating an empty scenario or opening an existing scenario. In case the user doesn't want to do any of this (though we don't really know why), the dialog can simply be closed.

3.2 Wizard Interface

Although a Scenario can be edited with random access, most beginners have no idea of what is required in a scenario or what the structure is. The Wizard provides a step by step approach to creating scenarios and entities within a scenario. It should be designed as if someone was behind the user, telling him what he should be doing next in order to accomplish a task smoothly. Once the user has used the wizard a few times, he can choose to bypass the sequential approach and use the expert interface with random access.

The wizard has two main purposes:

- Teach the user how to create a scenario
- Allow a user to quickly create a valid scenario

3.2.1 Scenario Wizard

The scenario wizard is accessible from the file menu (New Scenario) and is also shown by default with the MapEditor is launched from the executable file (as opposed to from a scenario file). The Scenario Wizard is divided in four steps:

1. Setting the name of the scenario
2. Adding episodes
3. Adding players
4. Confirming the settings

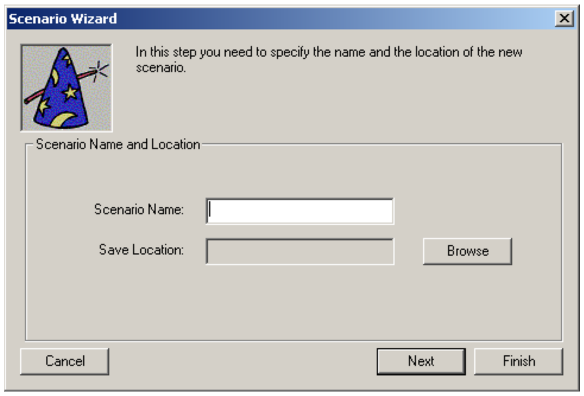

In this step you need to specify the name and the location of the new scenario.

Figure 5: Scenario Wizard: screen 1

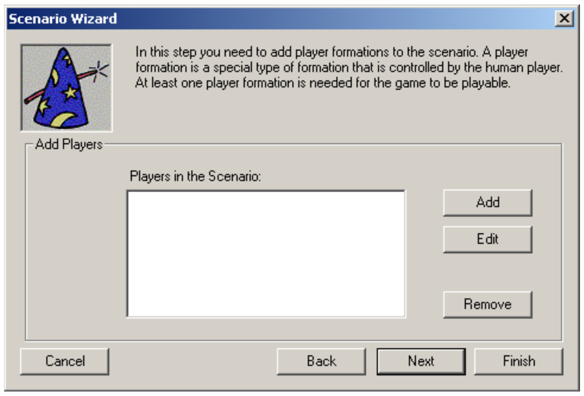

In this step you need to add player formations to the scenario. A player formation is a special type of formation that is controlled by the human player. At least one player formation is needed for the game to be playable.

Figure 6: Scenario Wizard: screen 2

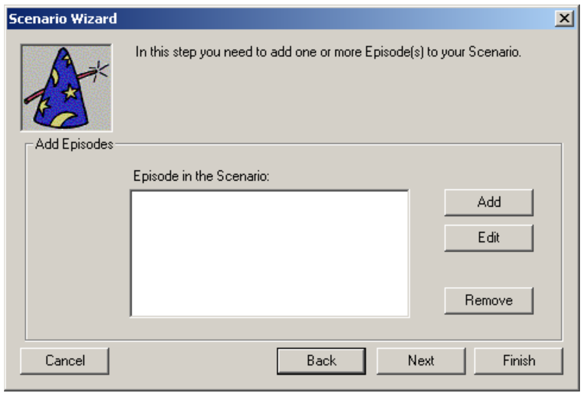


Figure 7: Scenario Wizard: screen 3

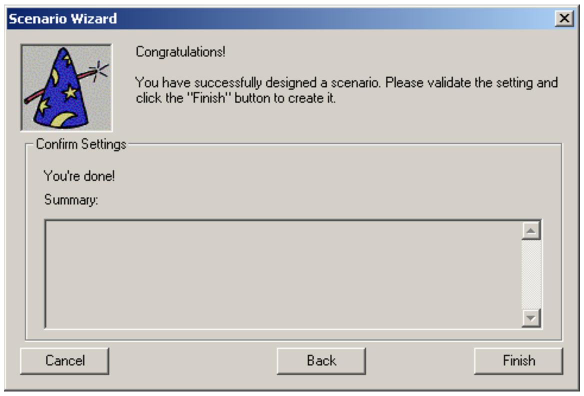

Scenario Wizard
Congratulations!
You have successfully designed a scenario. Please validate the setting and click the "Finish" button to create it.
Confirm Settings
You're done!
Summary:
Cancel
Back
Finish

Figure 8: Scenario Wizard: screen 4

Assumed User Intention:	Be guided into creating and setting a new scenario and scenario entities.
Next Anticipated Activity:	• Save the Scenario • Browse the Scenario • Edit and change settings of episode, chapters, formations, actors, players, images or sound • Add an episode, chapter, formation, actor, player using the wizard • Add an episode, chapter, formation, actor, player, image, sound using the standard interface • Close the scenario
Approach Chosen:	Since this task involves many steps and the user expects them to be sequential, we choose to make the wizard modal with four options: Cancel, Back, Next and Finish. **Cancel** closes the wizard without creating the new scenario. **Back** brings the user to the previous step. **Next** brings the user to the next step. **Finish** closes the wizard and creates the scenario. If the user had not entered values for all the fields or did not complete all the steps, default values are used.

3.2.2 Episode Wizard

The Episode wizard is accessible from the Scenario menu (Episode Wizard) and is also used by the Scenario Wizard. The Player Wizard is divided in four steps:

1. Setting the name of the Episode
2. Setting Episode settings
3. Adding Formations to the player formation
4. Confirming the settings

Assumed User Intention:	Be guided into creating and setting a new episode.
Next Anticipated Activity:	• Save the Scenario • Browse the Scenario • Edit and change settings of episode, chapters, formations, actors, players, images or sound • Add an episode, chapter, formation, actor, player using the wizard • Add an episode, chapter, formation, actor, player, image, sound using the standard interface • Close the scenario • Continue with the Scenario Wizard
Approach Chosen:	Since this task involves many steps and the user expects them to be sequential, we choose to make the wizard modal with four options: Cancel, Back, Next and Finish. **Cancel** closes the wizard without creating the new episode. **Back** brings the user to the previous step. **Next** brings the user to the next step. **Finish** closes the wizard and creates the episode. If the user had not entered values for all the fields or did not complete all the steps, default values are used.

3.2.3 Formation Wizard

The Formation wizard is accessible from the Scenario menu (Formation Wizard) and is also used by the Episode Wizard. The Formation Wizard is divided in four steps:

1. Setting the name of the formation
2. Setting formation settings
3. Adding actors to the formation
4. Confirming the settings

Assumed User Intention:	Be guided into creating and setting a new formation.
Next Anticipated Activity:	• Save the Scenario • Browse the Scenario • Edit and change settings of episode, chapters, formations, actors, players, images or sound • Add an episode, chapter, formation, actor, player using the wizard • Add an episode, chapter, formation, actor, player, image, sound using the standard interface • Close the scenario • Continue with the Episode Wizard
Approach Chosen:	Since this task involves many steps and the user expects them to be sequential, we choose to make the wizard modal with four options: Cancel, Back, Next and Finish. **Cancel** closes the wizard without creating the new formation. **Back** brings the user to the previous step. **Next** brings the user to the next step. **Finish** closes the wizard and creates the formation. If the user had not entered values for all the fields or did not complete all the steps, default values are used.

3.2.4 Actor Wizard

The Actor wizard is accessible from the Scenario menu (Actor Wizard) and is also used by the Formation Wizard. The Actor Wizard is divided in four steps:

1. Setting the name of the Actor
2. Setting Actor settings
3. Adding states to the Actor
4. Confirming the settings

Assumed User Intention:	Be guided into creating and setting a new Actor.
Next Anticipated Activity:	• Save the Scenario • Browse the Scenario • Edit and change settings of episode, chapters, formations, actors, players, images or sound • Add an episode, chapter, formation, actor, player using the wizard • Add an episode, chapter, formation, actor, player, image, sound using the standard interface • Close the scenario • Continue with the Formation Wizard
Approach Chosen:	Since this task involves many steps and the user expects them to be sequential, we choose to make the wizard modal with four options: Cancel, Back, Next and Finish. **Cancel** closes the wizard without creating the new Actor. **Back** brings the user to the previous step. **Next** brings the user to the next step. **Finish** closes the wizard and creates the Actor. If the user had not entered values for all the fields or did not complete all the steps, default values are used.

3.2.5 Player Wizard

The Player wizard is accessible from the Scenario menu (Player Wizard) and is also used by the Scenario Wizard. The Player Wizard is divided in four steps:

1. Setting the name of the player
2. Setting player settings
3. Adding actors to the player formation
4. Confirming the settings

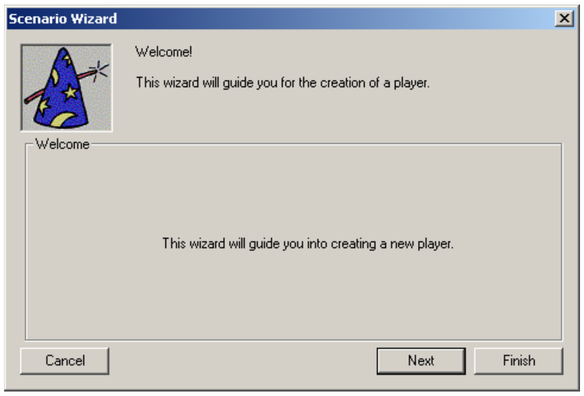
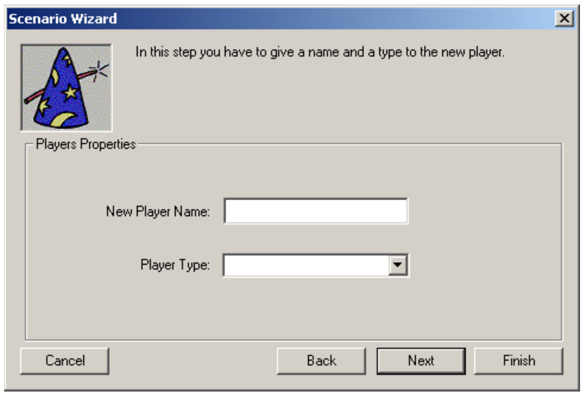
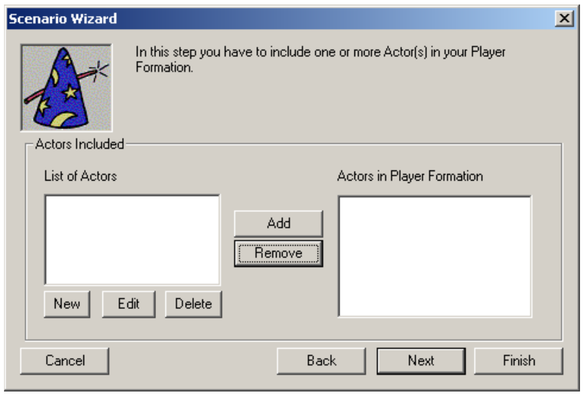

Scenario Wizard
Welcome!
This wizard will guide you for the creation of a player.
Welcome
This wizard will guide you into creating a new player.
Cancel
Next
Finish

Figure 9: Player Wizard: screen 1



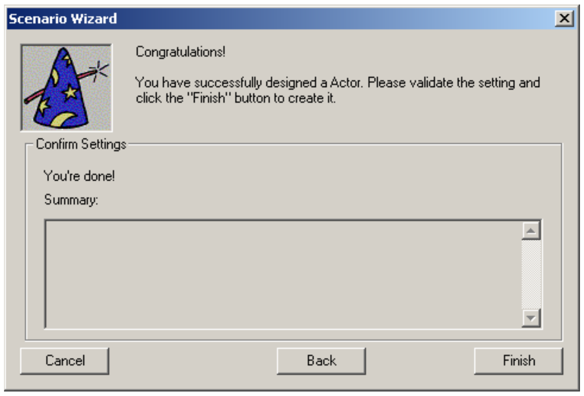
In this step you have to give a name and a type to the new player.

Figure 10: Player Wizard: screen 2



In this step you have to include one or more Actor(s) in your Player Formation.

Figure 11: Player Wizard: screen 3



Congratulations!

You have successfully designed a Actor. Please validate the setting and click the "Finish" button to create it.

You're done!

Figure 12: Player Wizard: screen 4

Assumed User Intention:	Be guided into creating and setting a new player.
Next Anticipated Activity:	• Save the Scenario • Browse the Scenario • Edit and change settings of episode, chapters, formations, actors, players, images or sound • Add an episode, chapter, formation, actor, player using the wizard • Add an episode, chapter, formation, actor, player, image, sound using the standard interface • Close the scenario • Continue with the Scenario Wizard
Approach Chosen:	Since this task involves many steps and the user expects them to be sequential, we choose to make the wizard modal with four options: Cancel, Back, Next and Finish. **Cancel** closes the wizard without creating the new Player. **Back** brings the user to the previous step. **Next** brings the user to the next step. **Finish** closes the wizard and creates the Player. If the user had not entered values for all the fields or did not complete all the steps, default values are used.

3.3 Expert Interface

The expert interface provides random access into the Scenario. Since we wanted the user to have a feeling that he is free to edit whatever he wants to, we decided that no dialog would be modal unless really needed. Thus, as soon as the user makes a change, it is updated and everything on the screen is updated to reflect the change.

3.3.1 Tree Browse View

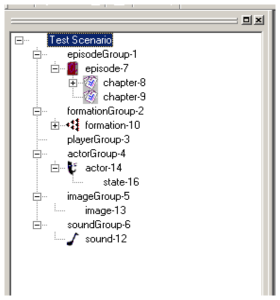


Figure 13: Tree Browse View

Assumed User Intention:	Navigate Through the Scenario.
Next Anticipated Activity:	• Edit and change settings of episode, chapters, formations, actors, players, images or sound • Add an episode, chapter, formation, actor, player using the wizard • Add an episode, chapter, formation, actor, player, image, sound using the standard interface
Approach Chosen:	The data in a scenario is organized as a three (a scenario contains episode, formations, actors, players, images and sounds. An episode contains chapters, etc...). Thus it was decided that the best way to access entities within the Scenario was to display the content as a tree. Also, since the user is allowed to edit any entity at any time, the tree becomes the main navigation tool and it was decided that it would always be on the screen. Additionally, since the Tree Browser would always be visible and provides access to almost everything in the scenario, we decided to add contextual menus when the user right-clicks on an item. For example, right-clicking on the Actor Group will display “New Actor” and “New Actor From Wizard”.

3.3.2 Episode Form

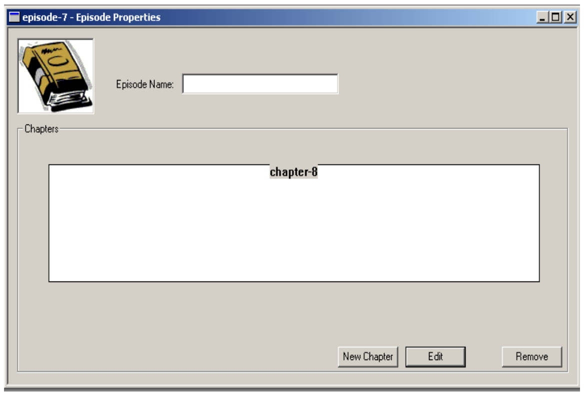


Figure 14: Episode Form

Assumed User Intention:	Modify the settings of an Episode.
Next Anticipated Activity:	• Save the Scenario • Browse the Scenario • Edit and change settings of episode, chapters, formations, actors, players, images or sound • Add an episode, chapter, formation, actor, player using the wizard • Add an episode, chapter, formation, actor, player, image, sound using the standard interface • Close the scenario
Approach Chosen:	The Episode contains Chapters so we decided that it should be possible to access the chapters from the episode form. A preview of the Chapters is thus displayed in the Episode, and by double-clicking or right-clicking, the user can open the associated Chapter. This form is not modal, and the user can close it or switch to another form at any time. Changes to the form are effective immediately.

3.3.3 Chapter Form

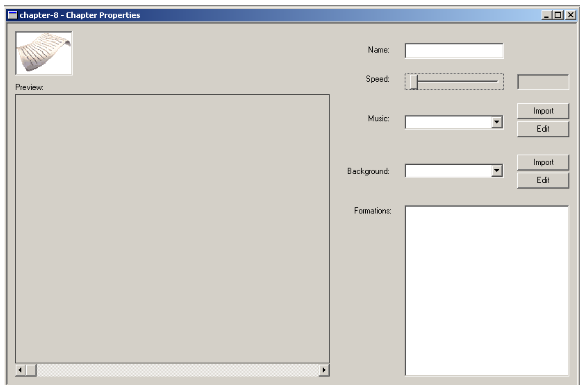


Figure 15: Chapter Form

Assumed User Intention:	Modify the settings of a Chapter.
Next Anticipated Activity:	• Save the Scenario • Browse the Scenario • Edit and change settings of episode, chapters, formations, actors, players, images or sound • Add an episode, chapter, formation, actor, player using the wizard • Add an episode, chapter, formation, actor, player, image, sound using the standard interface • Close the scenario
Approach Chosen:	The Chapter contains references to formations, to a background image and to a sound. Since the image and the sound have to be imported first (i.e. they must be in the Sound/Image group), we allow the user to import the sound or the image from the Chapter form. Formations are added through drag and drop. We chose to use a object approach, where any Formation displayed can be dragged and dropped onto the scenario (thus it can be selected from the Tree Browser). However, since we thought that dragging from the Tree Browser would not be intuitive, we added a list of formation on the Chapter dialog from which Formations can also be dragged and dropped. Also, once the formations have been placed on the Chapter, they can be moved to other positions or removed using the delete key or by right-clicking. Since this form is closely linked to formations, we also allow the user to edit a formation by double-clicking or right-clicking the formation. This form is not modal, and the user can close it or switch to another form at any time. Changes to the form are effective immediately.

3.3.4 Formation Form

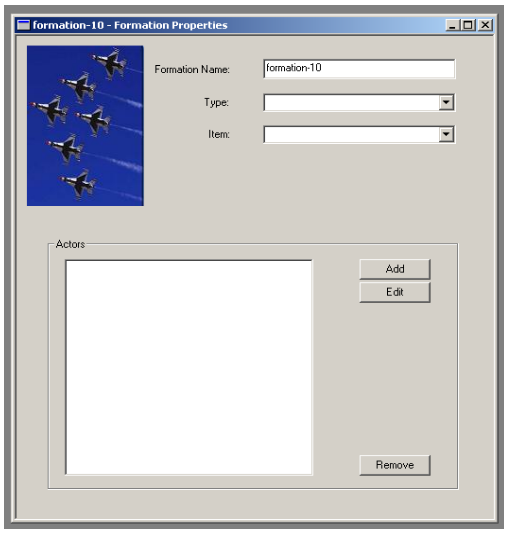
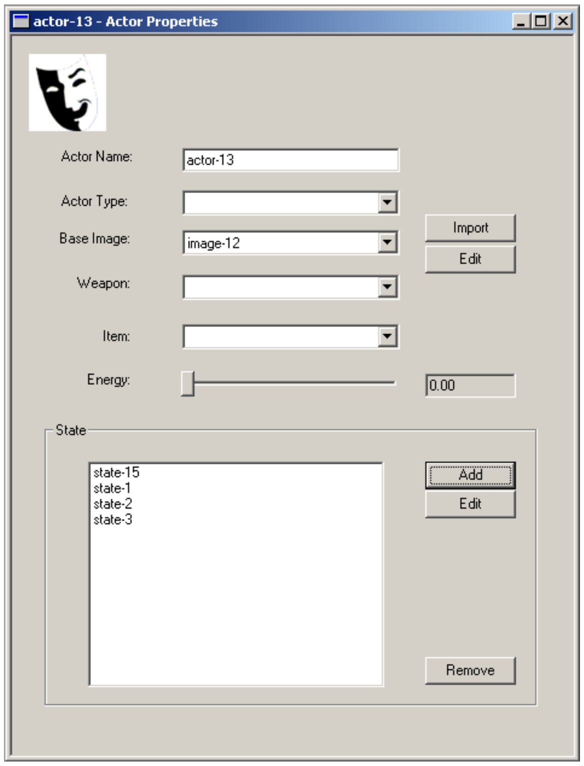


Figure 16: Formation Form

Assumed User Intention:	Modify the settings of a Formation.
Next Anticipated Activity:	• Save the Scenario • Browse the Scenario • Edit and change settings of episode, chapters, formations, actors, players, images or sound • Add an episode, chapter, formation, actor, player using the wizard • Add an episode, chapter, formation, actor, player, image, sound using the standard interface • Close the scenario
Approach Chosen:	The Formation form holds references to actors and various settings about the formation. The main issue was to allow the user to add actors to the formation and we decided to support to approaches: • Dragging an actor from the Tree Browser • Clicking on “Add Actor” prompts the user to chose an actor from a modal dialog containing a list of actor. Since this form is closely related to the Actor form, we allow the user to edit an actor by double-clicking or right-clicking the actor in the actor list. This form is not modal, and the user can close it or switch to another form at any time. Changes to the form are effective immediately.

3.3.5 Actor Form


actor-13 - Actor Properties

Actor Name: actor-13

Actor Type:

Base Image: image-12

Weapon:

Item:

Energy: 0.00

State

state-15
state-1
state-2
state-3

Figure 17: Actor Form

Assumed User Intention:	Modify the settings of an actor.
Next Anticipated Activity:	• Save the Scenario • Browse the Scenario • Edit and change settings of episode, chapters, formations, actors, players, images or sound • Add an episode, chapter, formation, actor, player using the wizard • Add an episode, chapter, formation, actor, player, image, sound using the standard interface • Close the scenario • Add, edit or remove a state to the actor
Approach Chosen:	The Actor form holds various settings about the actor, mainly the definition of its states. States can be created , edited or deleted. The actor has a base image which references an image from the image group. Since it is likely that the user will want to use an image not yet imported, we allow him to import an image directly from the actor form. This form is not modal, and the user can close it or switch to another form at any time. Changes to the form are effective immediately.

3.3.6 State Form

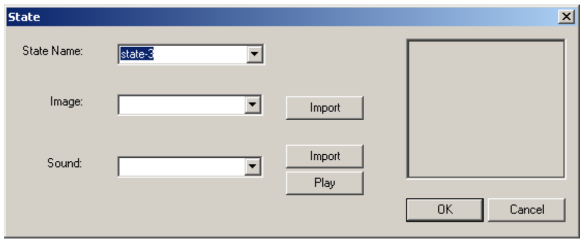


Figure 18: State Form

Assumed User Intention:	Modify the state of an actor.
Next Anticipated Activity:	• Save the Scenario • Browse the Scenario • Edit and change settings of episode, chapters, formations, actors, players, images or sound • Add an episode, chapter, formation, actor, player using the wizard • Add an episode, chapter, formation, actor, player, image, sound using the standard interface • Close the scenario
Approach Chosen:	Since a state can be owned by only one actor and since it does not depend on anything else, we thought that it would be unlikely for a user to quit editing a state to edit something else before he is finished with the state. We thus decided to make this form modal. The settings of a state require an image and a sound, so we allow the user to import them directly from the form if they are not in already in the image/sound group. Since this form is modal, changes are affective once the user clicks on "OK" and the changes to the state are discarded if he clicks on "Cancel"

3.3.7 Player Form

Assumed User Intention:	Modify the settings of a player.
Next Anticipated Activity:	• Save the Scenario • Browse the Scenario • Edit and change settings of episode, chapters, formations, actors, players, images or sound • Add an episode, chapter, formation, actor, player using the wizard • Add an episode, chapter, formation, actor, player, image, sound using the standard interface • Close the scenario
Approach Chosen:	A Player is similar to a Formation with less settings. It contains reference to actors, and we decided to be consistent with the Formation Form and use the same approach to let the user add actors to the formation. This form is not modal, and the user can close it or switch to another form at any time. Changes to the form are effective immediately.

3.3.8 Image Form

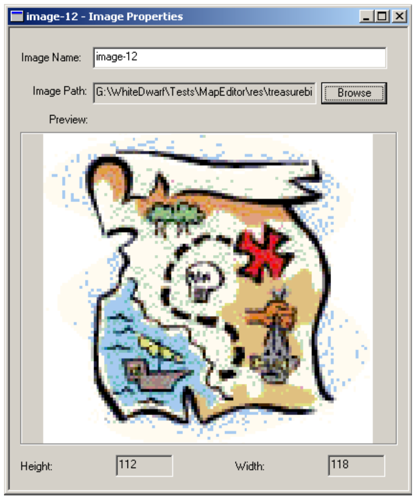


Image Name: image-12
Image Path: G:\WhiteDwarf\Tests\MapEditor\res\treasurebi
Height: 112
Width: 118

Figure 19: Image Form

Assumed User Intention:	View an image or change its name or location.
Next Anticipated Activity:	• Save the Scenario • Browse the Scenario • Edit and change settings of episode, chapters, formations, actors, players, images or sound • Add an episode, chapter, formation, actor, player using the wizard • Add an episode, chapter, formation, actor, player, image, sound using the standard interface • Close the scenario
Approach Chosen:	This form is straight-forward. We display the image in the form and allow the user to change the name of the image. Since the image might not fit in the form, we decided to scale it down (the use of Scroll Bars was rejected since the user probably wants to have a rough idea of what the image is), but to avoid confusion, we decided that it would be important to display the actual dimensions of the image so that the user be aware that it has been scaled to fit the window.

3.3.9 Sound Form

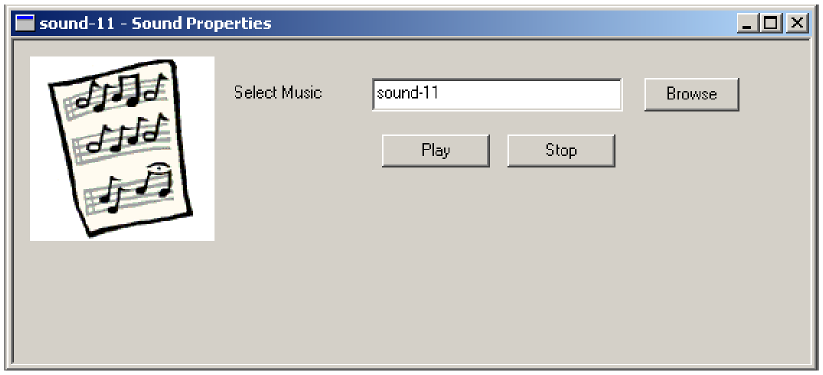


Figure 20: Sound Form

Assumed User Intention:	Hear a sound or change its name or location.
Next Anticipated Activity:	• Save the Scenario • Browse the Scenario • Edit and change settings of episode, chapters, formations, actors, players, images or sound • Add an episode, chapter, formation, actor, player using the wizard • Add an episode, chapter, formation, actor, player, image, sound using the standard interface • Close the scenario
Approach Chosen:	This form is straight-forward. We allow the user to play a sound and allow the user to change the name of the sound. Since the user doesn't need to a stop button when the sound is no playing, we decided to change the label of the "play" button into a "stop" button once it is pressed. Since the sound might be long, we also display the length of the audio clip and the current position as it plays. This form is not modal, and the user can close it or switch to another form at any time. Changes to the form are effective immediately.

3.3.10 Status Bar

Assumed User Intention:	None, provides feedback only.
Next Anticipated Activity:	• N/A
Approach Chosen:	Since we use a non-modal approach for most of the forms, while the user enters a value, it might be temporarily invalid (for example, as the user renames an entity, at one point it might have the same name as another entity). We do not was to display a modal error window when this happens, since the user might not be finished and might already be aware that the data is invalid. We thus provide information in the status bar to inform the user and give him a hint without preventing him from working.