ScriptEase: Generating Scripting Code for Computer Role-Playing Games

M. McNaughton, M. Cutumisu, D. Szafron, J. Schaeffer
Department of Computing Science, University of Alberta, Edmonton, AB Canada T6G 2E8
{mcnaught, meric, duane, jonathan}@cs.ualberta.ca

J. Redford
BioWare Corp.
200, 4445 Calgary Trail
Edmonton, AB
Canada T6G 5R7
jamesr@bioware.com

D. Parker
Electronic Arts (Canada) Inc.
4330 Sanderson Way
Burnaby, BC
Canada V5G 4X1
dparker@ea.com

Abstract

The state-of-the-art in game scripting is to manually script individual game objects that interact in the game. Thousands of non-player characters (NPCs) and props need to be scripted before they play a part in a game adventure. This situation introduces serious concerns about programming effort and reliability. We demonstrate ScriptEase [5], a tool to facilitate the game scripting process. It is based on generative design patterns for automatic code generation of scripts associated with game objects. ScriptEase is intended for a broad audience, from programmers to game designers and users without programming experience. Game designers can use commonly occurring patterns to generate scripting code without any programming knowledge. This demonstration illustrates the entire process of creating and scripting game props and NPCs.

1. Scripting in Computer Role-Playing Games

The computer game industry's commercial success is due to its ability to create captivating alternative worlds and to entertain game players by allowing them to meaningfully interact with these artificial environments. Continuing hardware advances are shifting the focus from form to content. Games are increasingly sold on the basis of their depth of story and engaging game play rather than their graphics performance.

A computer role-playing game (CRPG) contains an engine that dispatches game events to scripts. Each story is composed of individual modules that can contain thousands of areas, non-player characters (NPCs) and other game objects (props). Each object must be scripted to respond to these game events. Currently, programmers write scripts manually and this situation introduces serious concerns about programming effort, reliability, and testability. Moreover, the scripting process is difficult for those game designers that lack programming skills.

We present ScriptEase [5], a scripting tool that facilitates the scripting process for Neverwinter Nights (NWN), a multi-award winning (86 awards) computer role-playing game (CRPG) from BioWare Corp. NWN is a popular community-based game. In addition to BioWare designers, thousands of amateur designers also write and share their own game adventures on the Web. For example, there are 3357 adventure stories posted in a common repository [4] and the most popular adventure has been downloaded over 200,000 times.

We demonstrate how ScriptEase is used to create interactions between the player character (PC) and game objects by attaching scripts to them. We show the steps in scripting a module with ScriptEase, from creating a module using NWN's Aurora toolset, through using ScriptEase and finally, playing the module with NWN.

2. ScriptEase

The state-of-the-art in game scripting is to script individual game objects that interact in the game. Objects must be tracked using criteria that include their physical areas, their associated sub-plots, or their static/adaptive status. Tracking the objects in this manner is hard, but tracking the scripts is even more difficult since most scripts involve the interaction of several objects.

ScriptEase [5] is a scripting tool that enables game designers to script their game adventures easily and quickly. Generative design patterns (GDPs) [1] are used to automatically generate game scripts [3] and ScriptEase organizes the large number of generated scripts. ScriptEase supports a basic set of patterns that represent common interactions between the PC and props in the game, as well as more specialized patterns that occur frequently in the development of a particular story.

ScriptEase enables users to manage their scripts by creating folders to organize their patterns during game development. The pattern import/export mechanism available in ScriptEase is a high-level code reuse mechanism that stimulates the circulation of ideas among

Proceedings of the 19th International Conference on Automated Software Engineering (ASE'04)
1068-3062/04 $ 20.00 IEEE

designers. They can share patterns that improve the game experience and build incrementally on each other's work.

Testing a complex interactive system with thousands of scripts is challenging. Many common errors are difficult to detect without playing through all of the game scenarios and trying all of the different combinations of user choices. For example, scripts are often created using cut-and-paste techniques, and it is not uncommon for the programmer to cut-and-paste scripts without making all the changes needed for the new context. There are so many game objects and scripts that it has become standard practice to use object numbers or script numbers in the names of objects and scripts. An off-by-one error in a name often results in a script that is syntactically correct, but performs incorrectly, since it refers to the wrong object. ScriptEase eliminates these common errors by replacing explicit tags of props and NPCs by a simple set of dialogs that allow the designer to pick appropriate objects directly.

Game designers create the game's story line, but often do not write scripts, since many are not programmers. Therefore, programmers implement the story line by writing scripts. The extra level of indirection in the process (the programmer) increases the chances of creating a product that does not match the designer's intentions. Such miscues are analogous to the ones that occur between customers/requirements analysts and designer/programmers during the development of more general software systems.

The use of patterns to generate scripts will have a large impact on the methodology of game development. This technique will provide game designers with the power to execute their vision without sacrificing valuable programming time writing detailed scripts. The saved time can be used by designers to produce more compelling stories and richer characters that create a more realistic game experience. Time saved by programmers can be used for pattern writing and pattern testing.

Code generated by ScriptEase is self-documenting. The patterns themselves are used as documentation and comments are generated in the code. Thus, testing and debugging become easier for the programmers (should they ever need to go down to this level of detail). Moreover, using patterns decreases programming errors and increases code generation productivity and reliability by enabling rapid prototyping and code reuse. We have evaluated the re-use of *encounter patterns* in the original BioWare produced NWN story. In the case study [2], we have identified the implicit patterns that exist in the game modules and used GDPs to generate the code for all scripts in the Neverwinter Nights CRPG that are attached to a particular representative prop object called a *placeable*.

The case study determined which of the CRPG patterns constructed before the case study could be re-used and which new patterns were necessary. It showed that there is significant pattern reuse in successive modules and consequently fewer new patterns are required as the pattern base grows. It also demonstrated the strengths of our approach on an industrial-grade computer role-playing game (NWN), not just in an artificially constructed test environment.

Our tool demonstration shows all the steps in scripting a module with ScriptEase (Figure 1). First, we use NWN's Aurora toolset to create the environment for a module that represents a particular game adventure. After all the props are set in various areas of the module, specific game objects are selected to be scripted. ScriptEase is used to script props that interact in the game to enact the designer's vision. The game designer selects patterns from the pattern library provided with ScriptEase. Then, after adapting the patterns, the game designer compiles the module and the scripting code is automatically generated. The last step is running the module in the NWN game.

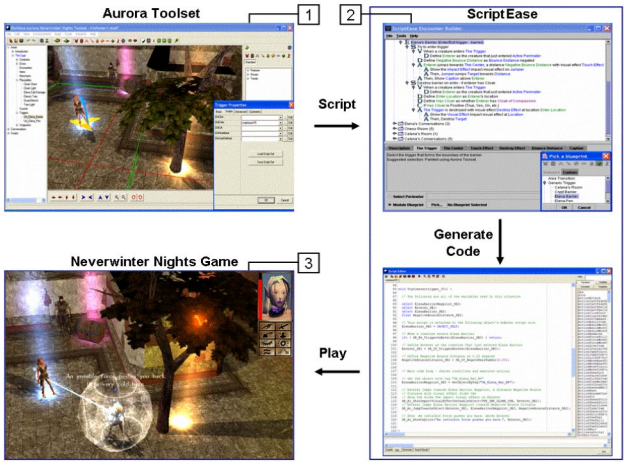

Figure 1. The ScriptEase development process (Aurora Toolset and Neverwinter Nights Game 2004 © Atari, Inc.).

References:

[1] S. MacDonald, D. Szafron, J. Schaeffer, J. Anvik, S. Bromling, K. Tan, "Generative Design Patterns", ASE, Edinburgh, UK, September 2002, pp 23-34.

[2] M. McNaughton, M. Cutumisu, D. Szafron, J. Schaeffer, J. Redford, D. Parker, "ScriptEase: Generative Design Patterns for Computer Role-Playing Games", to appear, ASE, Linz, Austria, September 2004.

[3] M. McNaughton, J. Redford, J. Schaeffer, D. Szafron, "Pattern-based AI Scripting using ScriptEase", AI 2003, Halifax, Canada, June 2003, pp 35-49.

[4] NWVault. http://nwvault.ign.com/Files/modules/modulesTop3.shtml.

[5] ScriptEase, www.cs.ualberta.ca/~script/scriptease.html.