

Exult: The Open Age of Ultima

Contributed by nitin

Old adventures never end--they're just replayed. In the case of Ultima VII, open source gives the classic RPG added replay mileage with an engine overhaul. Exult replaces Ultima VII's engine so that the game can be played on other operating systems and platforms. It also expands the graphics capabilities of the game and adds features not present under the original engine. (A legal copy of Ultima VII is required in order to use Exult.)

Lord British (aka Richard Garriott) himself, the creator of the entire Ultima franchise, has given the Exult team his unofficial blessing. In fact, Exult may be the only way to extend the game play of Ultima VII, officially or otherwise. Supposedly, the source code for the Ultima VII engine has been lost over the years as Origin Systems, the game's developer, shifted its focus to Ultima Online.

"Ultima was a really great saga and I can't remember how many hours I've spent playing it. Participating in this project was a way for me to thank the original developers," says Aurelien Marchand. The 24-year-old currently resides in Toronto, Canada, where he works in a brokerage company. He ported Exult to the Sharp Zaurus.

The Ultima-te Engine Replacement

Exult development began in the summer of 1998. In its earliest form, the program only displayed the Ultima VII landscape and let you browse the game environment. Since then, the engine has evolved into a viable replacement for the Ultima VII engine, with notable additions:

- * It runs on Linux, Windows, Mac, and other platforms such as the aforementioned Zaurus port. The official Ultima VII engine only runs on MS-DOS.
- * Exult supports programmable key mappings.
- * Graphics scalers let you run the game at double the normal graphic resolution and improved image quality.
- * An enhanced "Save Game" screen allows unlimited saves and shows a thumbnail mini-screenshot for each saved game.
- * You can optionally display your party's stats on the screen.
- * ExultStudio, an editor in development, will be a complete development environment for creating mods or entirely new games.

Ryan Nunn, a 22-year-old C++ programmer from Adelaide, Australia, joined the Exult project to improve the engine's MIDI music support and player movement/controls. He elaborates on these additions: "The scalers make a big improvement to the display. The new savegame interface supports a virtually infinite number of savegames, and shows various statistics about them. This is a big improvement from Ultima VII's savegame interface that only had support for 12 games without the features that I described. We also included new options menus that allow a greater control over how the game plays and looks. There are even simple things such as allowing the games to run in higher resolutions."

The vast majority of Exult's programming is handled by its developers. They use C++ as the main development language. Exult also incorporates libraries and small amounts of code from other projects, such as:

- * bison/flex for the usecode compiler.
- * GTK+, Glade, Freetype2, and Libpng for the ExultStudio GUI.
- * The FM synthesizer from the ScummVM project.
- * The SDL library for graphics, event-handling, and audio.

The biggest components not written by the Exult team are its graphics scalers. These were developed by Derek Liauw Kie Fa, based on the work he did for the Snes9x Super Nintendo Entertainment System emulator.

Translating Ultima VII's Engine Language

Jeff Freedman is a software engineer in Portland, Oregon. He wrangles with Exult's graphics coding and usecode (the

language upon which Ultima VII's game logic is built). According to Jeff, the toughest technical challenge to making Exult was, unsurprisingly, reverse-engineering of the Ultima VII engine. Since the engine is heavily based heavily on scripts, most of the game actions are controlled by functions in the usecode file.

"Usecode is bytecode for a virtual machine, very much like Java, except that its format isn't documented anywhere," says Freedman. "What made Exult possible was that a couple of fellows in Europe managed to figure out a sizable portion of usecode's format, and wrote a disassembler for it. However, we've still had to guess at many different functions and features, then re-evaluate our decisions when pieces of the game don't work properly."

While the usecode instruction format was already decoded by others, the Exult team had to figure out most of its intrinsic functions that are called by the interpreter. These functions manipulate things in the game's setting. Working out the parameters for these functions--figuring out what exactly they were meant to do--was a puzzle. "The real trick in the end is just knowing what the original game was attempting to do with the code," says Nunn. "By knowing this, you can make reasonable guesses about what an intrinsic actually does. In the end, you can only get it correct when something doesn't work as expected."

The Exult team then had to interpret game stats that they didn't fully understand, usually through a hit-or-miss technique. "If I were writing my own game, I'd have no trouble storing weapon stats in a file and then reading them back. With Ultima VII, we have those stats but have to guess what the numbers mean. And it's not always obvious right away when we get something wrong," says Freedman.

A problem with this guesswork might not turn up until weeks later when somebody playing the game would point out to the Exult developers that a particular monster was either too easy or too hard to beat. It would then be a chore to trace the root of the problem back to the Exult team's misinterpretation of the monster's weapon.

Understanding Ultima's usecode is advisable for anyone wanting to work with Exult code on their own. Even though there are a few problems with it--the usecode compiler isn't fully functional yet, and it's rather poorly documented. Nunn says that "we have developed a small example modification that can be used to understand usecode syntax and keywords. Mostly though, if you know C or C++, you shouldn't have much of a problem writing usecode."

Expanding Ultima's World for the Future

Features planned for future versions of Exult include improving the combat game play; that is, making it function as well as when the game is played under the Ultima VII engine, perhaps with a turn-based system. Exult's 1.1x branch already has support for digital music.

The Exult team also wants to make its engine more user-friendly to languages other than English. "Exult is all hard-coded in English. There are plans to move any hard-coded phrases into XML-like files," says Nunn. "This will allow Exult to be easily used by people who don't read English, by using translated XML files."

However, most future work by the Exult developers will focus on improving and completing ExultStudio. Once it is fully working, players will be able to tweak the inner-workings of games or create their own Ultima-style RPGs from scratch.

So better make that also: "Old adventures never end. They're just remade."