Project Proposal
SE690/696
Anoop Mathew

ZOOM - Automatic Code Generation: UML State Charts to Executable Java Code

Meta-model & software architecture design

Purpose

Design of software that is of high quality, easily extensible and reusable are key requirements of any software project. Dynamic code generation has been touted as a promising emerging technique for achieving these software development goals.

The overall goal of this project is to provide software architecture and tools to translate UML state charts and state implementations into executable Java code. As part of a team working on this project, my individual responsibilities will include development of a meta-model which describes UML state charts and design of a software architecture to make generated Java code executable.

Objectives

The overall objectives of this project are as follows:

1. Test UML state chart specification syntax.
2. Develop a meta-model to describe UML state chart specifications.
3. Develop a translation engine to translate UML state chart specifications into executable Java code.
4. Develop a software architecture with built in support for persistence, generation of signals and events, transition and state priorities, threading and concurrency.
5. Provide a framework to allow application developers to specify the actions semantics (business-related logic) associated with the UML state charts.

My responsibilities on this project will involve items 1, 2, 4 specified above.

Rationale

There are various dynamic code generation tools/methodologies available today. Some of these include Executable UML, ROOM etc. This project aims to incorporate the benefits of these solutions while also addressing the drawbacks of these various solutions. The solution developed will aim to be generic enough to allow generation of executable code in other languages as well.

Design

In the first phase of this project, I will be investigating other software development tools/methodologies that support automatic code generation using UML state charts. Specifically, I will be investigating Executable UML (xUML) to understand the benefits/drawbacks of this methodology. This will enable the project team and I solidify the requirements and provide some context for this project. As part of this endeavor, I

have already researched Executable UML in great detail and given a detailed presentation on the topic to the SE690 class.

The second phase of this project that I will be working on is testing the UML state chart specification syntax for completeness and ease of use. This phase of the project is currently ongoing. This task entails understanding the UML state chart specification 1.4 provided by the OMG and ensuring that the syntax that has been designed is sufficient to cover all aspects of dynamic state chart description. The test process involves writing sample state chart descriptions using the supplied syntax, running the test files through a parser (which has been provided) and analyzing the output.

In the third phase of this project, I will be involved in developing a meta-model that completely and succinctly describes UML state charts. This meta-model will be used as input to the translation engine.

In the fourth phase of the project, I will be developing an underlying architecture for the generated java code. This architectural framework will be the glue that holds together the generated java code and makes it executable. The architecture will have built in support for persistence, generation of signals and events, transition and state priorities, threading and concurrency.

Work Plan

- Phase 1: Analysis
  - Milestone 0: Project Web page (Spring 03)
  - Milestone 1: Develop project proposal (Spring 03)
  - Milestone 2: Technology and tools review (Spring 03)
  - Milestone 3: Test UML state chart description syntax (Spring 03)
  - Milestone 4: Initial Presentation (Spring 03)
- Phase 2: Design & Development
  - Milestone 0: Develop meta-model (Spring 03/Summer03)
  - Milestone 1: Develop architectural framework (Summer 03)
  - Milestone 2: Unit tests (Spring 03/Summer03)
- Phase 3: Integration
  - Milestone 0: Unit tests (Fall 03)
  - Milestone 1: Intermediate Presentation (Fall 03)
  - Milestone 2: Integration Testing (Fall 03)
  - Milestone 3: Final Demo/Presentation (Fall 03)
- Completion: Fall 03