FMSLEclipse
Software Requirements Specification

Jesse Englert
Advisor: Dr. Fisher
California Polytechnic State University

January 30, 2006

Contents

1 Introduction

1.1 Purpose

This document defines the requirements for the FMSLEclipse project.

1.2 Audience

This document is intended to be read by my senior project advisor and students doing further work on this project. The rest of this document describes the scope, functional requirements, and non functional requirements of FMSLEclipse.

1.3 Problem Statement

At Cal Poly University, most students taking an undergraduate software engineering course are required to learn and use Formal Modeling Specification Language (FMSL). They need an Integrated Development Environment (IDE) to ease the development process. The IDE should have functionality found in existing IDEs for modern languages such as Eclipse and Visual Studio.

2 Overall Description

2.1 Product Perspective

FMSLEclipse is an entirely new product. It will be written as a plugin to extend Eclipse. Eclipse is a development platform that provides a framework for different development tools. All development tools for Eclipse are written as plugins. The plugin will interface with Eclipse by using the framework API provided by Eclipse.

2.2 Product Features

FMSLEclipse will provide a full featured IDE for FMSL. This includes the ability to compile, a text editor, a project based structure with file navigation, and a hierarchical view of each FMSL file.

2.3 User Classes

The primary users of the system are software engineers who want to use FMSL to specify their software applications. The immediate users will be engineering students attending Cal Poly.

2.3.1 Student Users

Undergraduate students under the 2005-07 catalog will take the required software engineering courses during their junior or senior year. It is assumed they know how to program at this juncture and are familiar with at least one professional IDE.

2.4 Operating Environment

This section describes the environment that FMSLEclipse will operate in. Each requirement is designated by [OE-#] and is referred to in the rest of this document by that designation.

OE-1: FMSLEclipse shall run within Eclipse 3.1 .
Priority: High

OE-2: FMSLEclipse shall operate on any computer and operating system that Eclipse 3.1 does. For the scope of this project, FMSLEclipsewill only be tested on: an x86 based machine running Windows XP and a machine running OSX.
Priority: Low

OE-3: FMSLEclipse shall be Java 1.5 compatible. Any known portability issues between platforms will be avoided. Focusing on x86 Windows, Mac OSX, and SPARC Solaris.

2.5 Constraints

This section describes the design and implementation constraints of FMSLEclipse. Each requirement is designated by [CO-#] and is referred to in the rest of this document by that designation.

CO-1: FMSLEclipse shall use the existing FMSL compiler written in C and C++. Porting the FMSL compiler to Java could be a senior project in itself and is outside the scope of this project.
Priority: High

CO-2: FMSLEclipse shall be written in Java. Eclipse is written in Java and expects its plugins to be so also.
Priority: High

2.6 User Documentation

Any user documentation for FMSLEclipse will be found online in a section titled ”FMSLEclipse” under the Eclipse help system.

2.7 Assumptions and Dependencies

This section describes the assumptions and dependencies that affect the **monitor program**. Each requirement is designated by [AS-#] or [DE-#] and is referred to in the rest of this document by that designation.

DE-1: Dr. Fisher will provide the grammar for FMSL in a format that can be read by the parser generator we decide to use.
Priority: High

3 Functional Requirements

3.1 UI Overview

3.1.1 Workbench

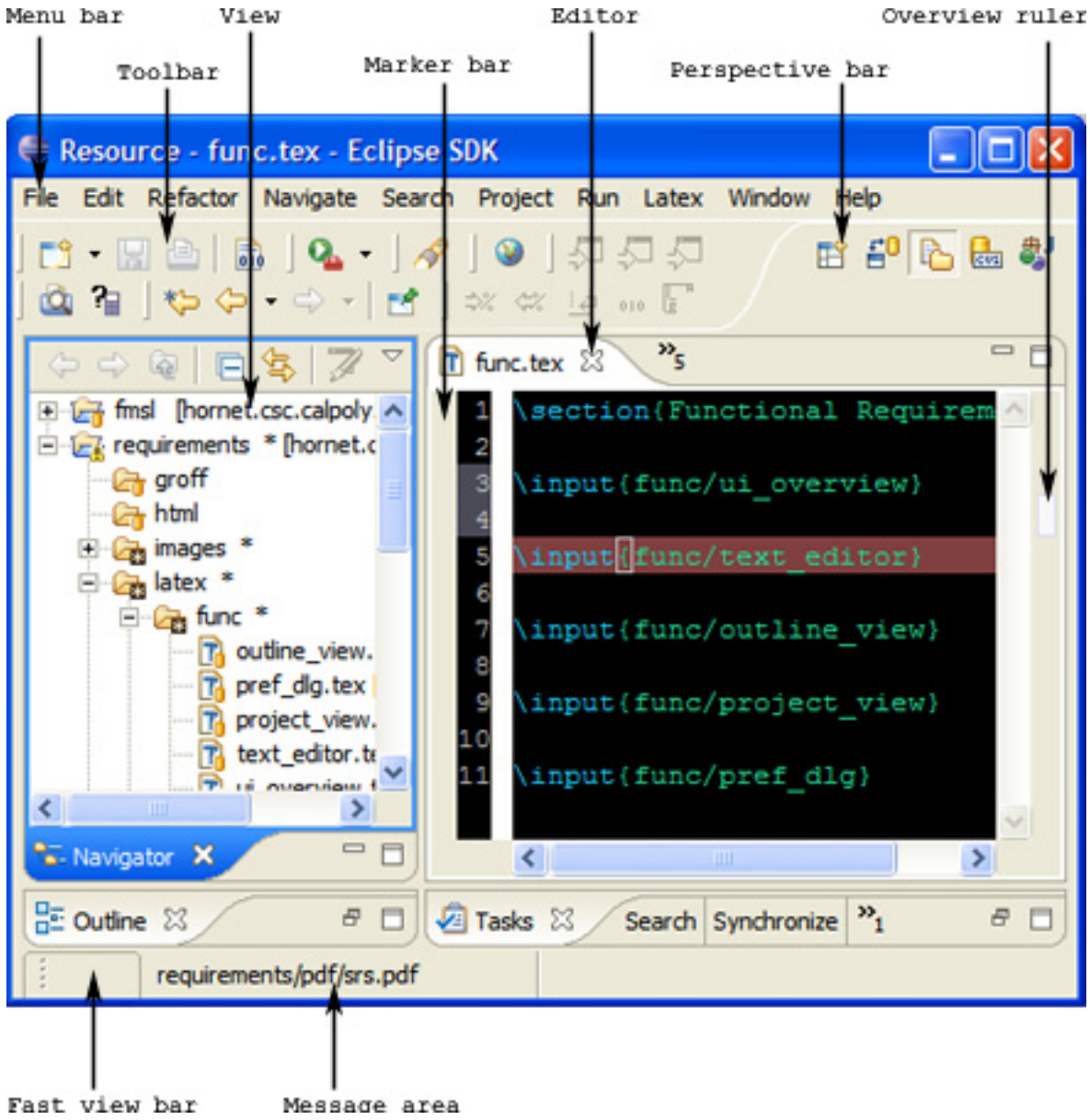

Figure 1: Workbench

The workbench (see figure 1 is the term used to refer to the main window presented to the user. It consists of a menu bar, toolbar, perspective bar, and a perspective. A perspective contains and controls the layout of multiple views and editors. An editor is used to edit a resource. A view is used to display information about the project or the resource being edited. Views may be detached from the workbench as seen in figure 2. The marker bar and overview ruler display annotations in the resource being edited. The fast view bar allows the user to place a view as

a shortcut in that bar and when selected it will display temporarily. The message area displays relevant messages.

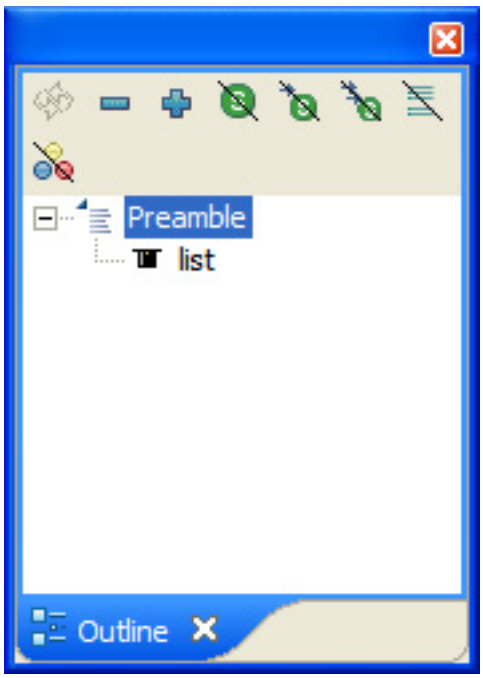

Figure 2: Detached View

3.2 Text Editor

The text editor in Eclipse allows the user to edit a source file. A specialized text editor for FMSL will provide features that help the user edit, navagate, and diagnose problems of a source file.

3.2.1 Syntax Highlighting

Priority: High

FMSLEclipse shall highlight the FMSL syntax in a source file. The colors used shall be selectable by the user (see 3.5. The following syntax shall be highlighted:

- comments
- import/export declarations
- expression keywords
- objects, operations, modules
- values
- attribute names

3.2.2 Automatic Indentation

Priority: High

FMSLEclipse shall automatically indent FMSL code.

comments: indent to same level as following token.

import/export declarations: level 0

modules: level 0

objects, operations, values: level 1

attributes: level 2

multi line attribute: level 3

3.2.3 Problem Markers

Priority: High

FMSLEclipse shall indicate that a line of code is causing a problem by marking it with a problem marker (figure 3). A problem marker is an annotation to the text editor that is displayed as an icon next to the line of code causing the problem. FMSLEclipse shall support the following markers:

- Warning
- Error

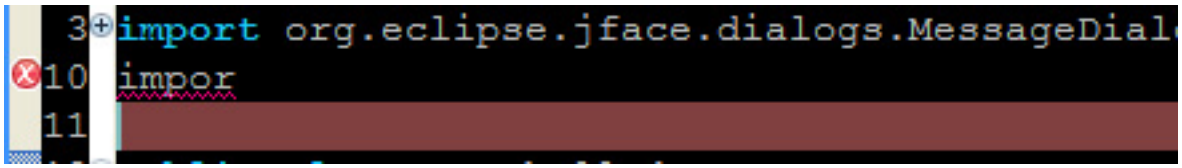

Figure 3: Problem Marker

3.2.4 Content Assist

Priority: Medium

FMSLEclipse shall provide content assist (figure 4) while editing a FMSL source file. Content assist will auto complete code as it is typed by providing a list of possibilities.

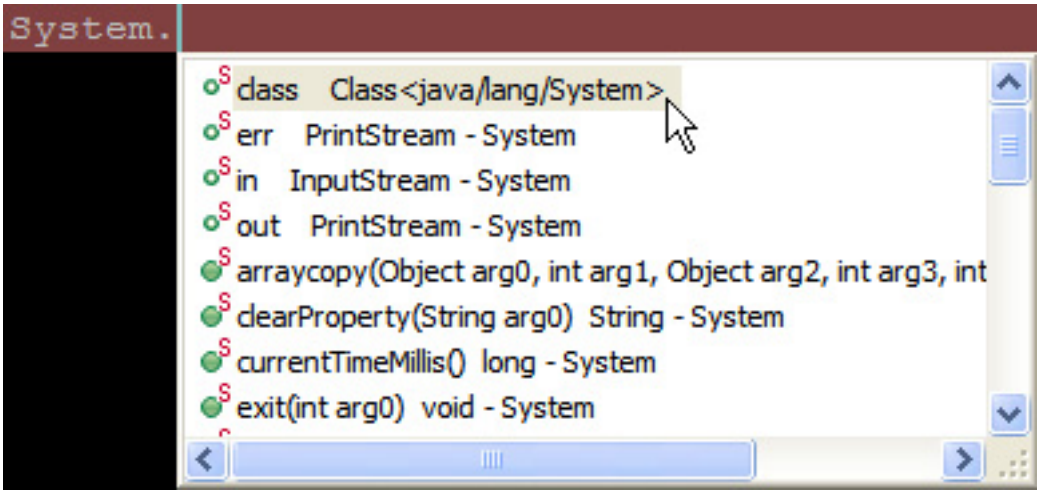

Figure 4: Content Assist

3.2.5 Code Folding

Priority: Low

FMSLEclipse shall allow the user to fold sections of code in order to improve readability. Folding code reduces a section of code so that only one line is viewable. The following FMSL sections are foldable:

- Module
- Object
- Operation

3.2.6 Line Level

Priority: Low

FMSLEclipse shall indicate that a section of code is causing a problem by underlining it with a colored wave like line. FMSLEclipse shall underline warnings in yellow and errors in red.

3.2.7 Quick Fix

Priority: Low

FMSLEclipse shall provide quick fixes for each problem makker generated (see 3.2.3). A quick fix (figure 5) is a set of resolutions for the problem that the user may select from to automatically solve the problem.

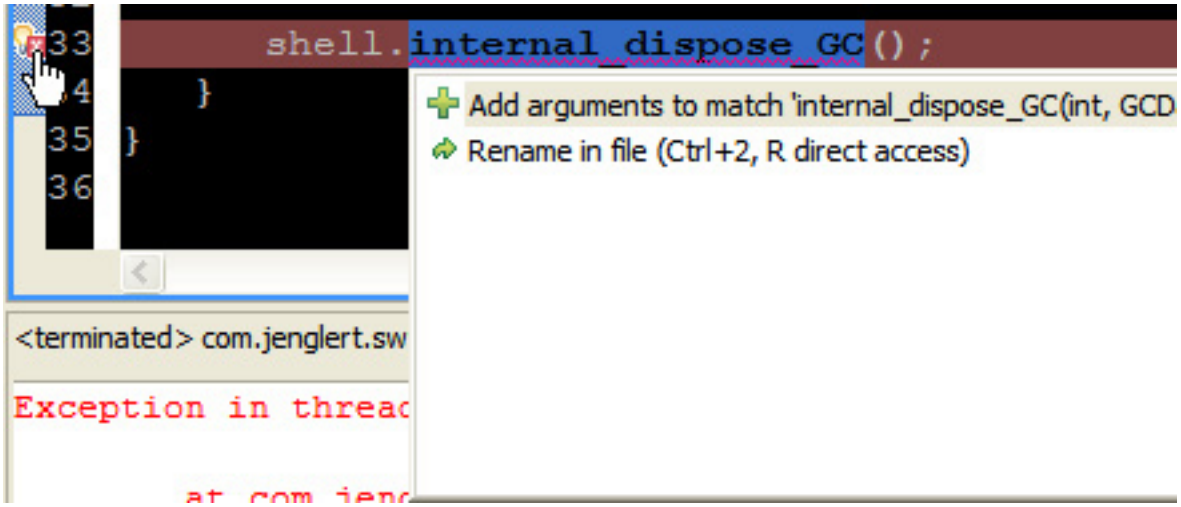

Figure 5: Quick Fix

3.2.8 Text Hover

Priority: Low

FMSLEclipse shall display text hovers (figure 6) for the FMSL syntax. A tool tip like text box is displayed when the user hovers the mouse over a FMSL keyword in the editor. There should be a user preference to turn this feature on and off.

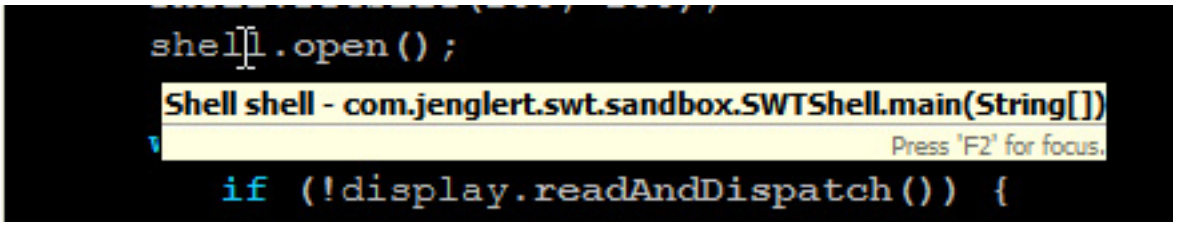

Figure 6: Text Hover

3.3 Outline View

The outline view (figure 7) in Eclipse displays a hierarchical structure of the current source file being edited. A specialized outline view for FMSL will display the structure of the current FMSL source file being edited.

3.3.1 Structure

Priority: High

FMSLEclipse shall display the structure of the FMSL source file being edited in a hierarchical manner. The following hierarchy will be used:

- Module

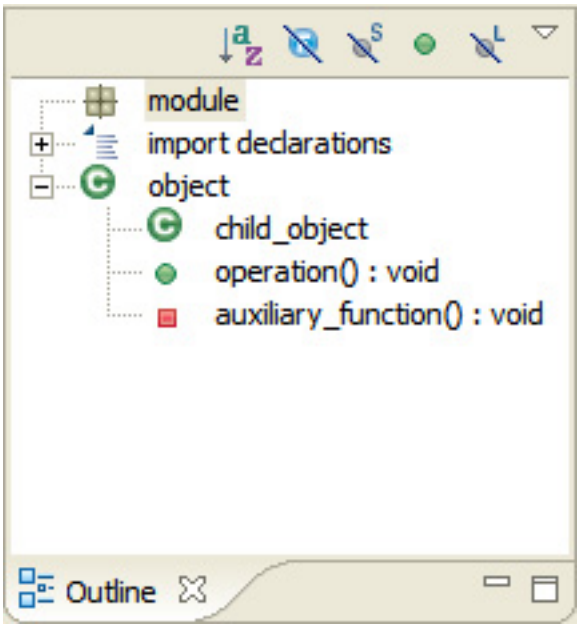

Figure 7: Outline View

- Imports
- Exports
- Object
  - Child Object
- Operation
- Value
- Auxiliary Function

3.3.2 Linking

Priority: Medium

FMSLEclipse shall link elements in the outline view to the source file. When the user selects an element in the outline view, the corresponding code in the source file will be highlighted.

3.4 Project View

The navigator view (figure 8) in Eclipse displays a hierarchical structure of the workspace file system. A specialized project view for FMSL will be built on top of the navigator view to provide more information about FMSL projects and files.

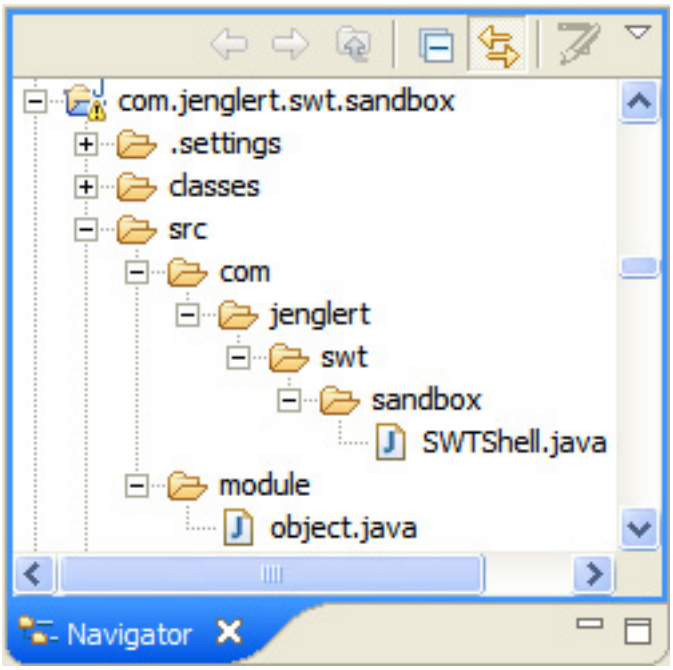

Figure 8: Navigator View

3.4.1 File Hierarchy

Priority: Medium

FMSLEclipse shall display the structure of a FMSL source file when the file is expanded (figure 9). It will display the same information as the outline view (see section 3.3) and use the hierarchy specified in section 3.3.1.

3.5 Preferences Dialog

The preferences dialog in Eclipse contains the preferences for each plugin. Plugins are responsible for contributing preferences to this dialog. FMSLEclipse will contribute preferences to the dialog that will be contained under the FMSLEclipse category.

3.5.1 Editor Preferences

Priority: High

The editor preference page is located under the FMSLEclipse section in the preferences dialog. The syntax highlighting preference page (figure 10) is located under the editor preferences. This page will allow the user to specify the color of different FMSL syntax.

3.5.2 Builder Preferences

Priority: High

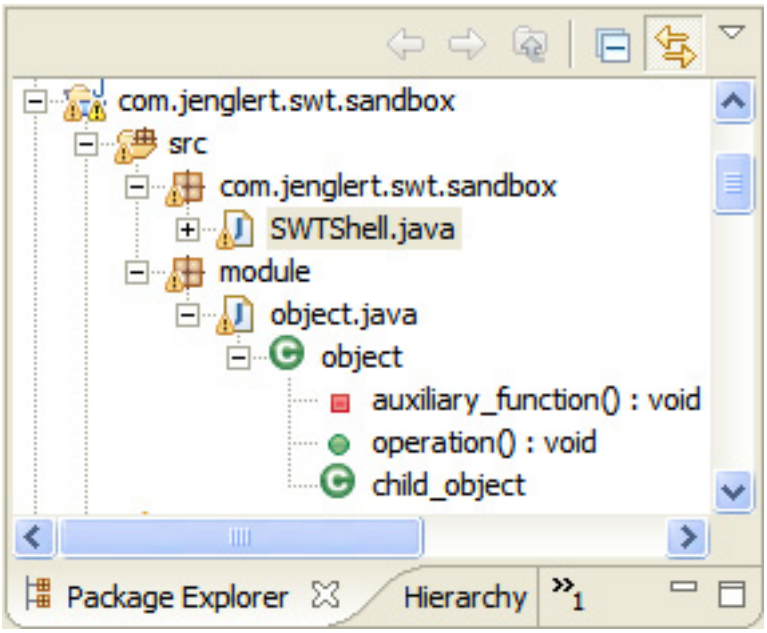

Figure 9: File Structure

The builder preference page is located under the FMSLEclipse section in the preferences dialog. This page will allow the user the specify a path to the external FMSL compiler. The page will look similar to figure 11.

3.6 Wizards

3.6.1 New Project Wizard

Priority: High

FMSLEclipse shall use a wizard (see figure 12 to guide the user in creating a new FMSL project. The wizard with prompt the user for the following information:

- Project Name
- Project Location
- Output Directory
- Source Directory

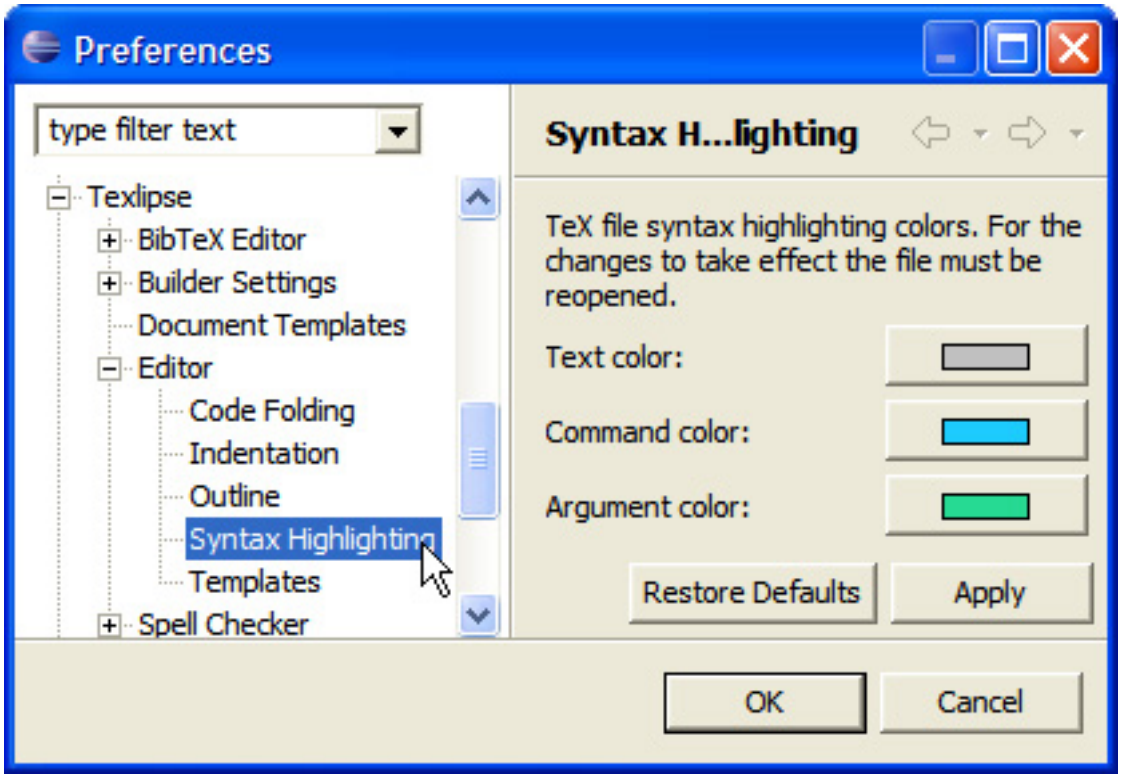

Figure 10: Syntax Highlighting In Editor Preferences

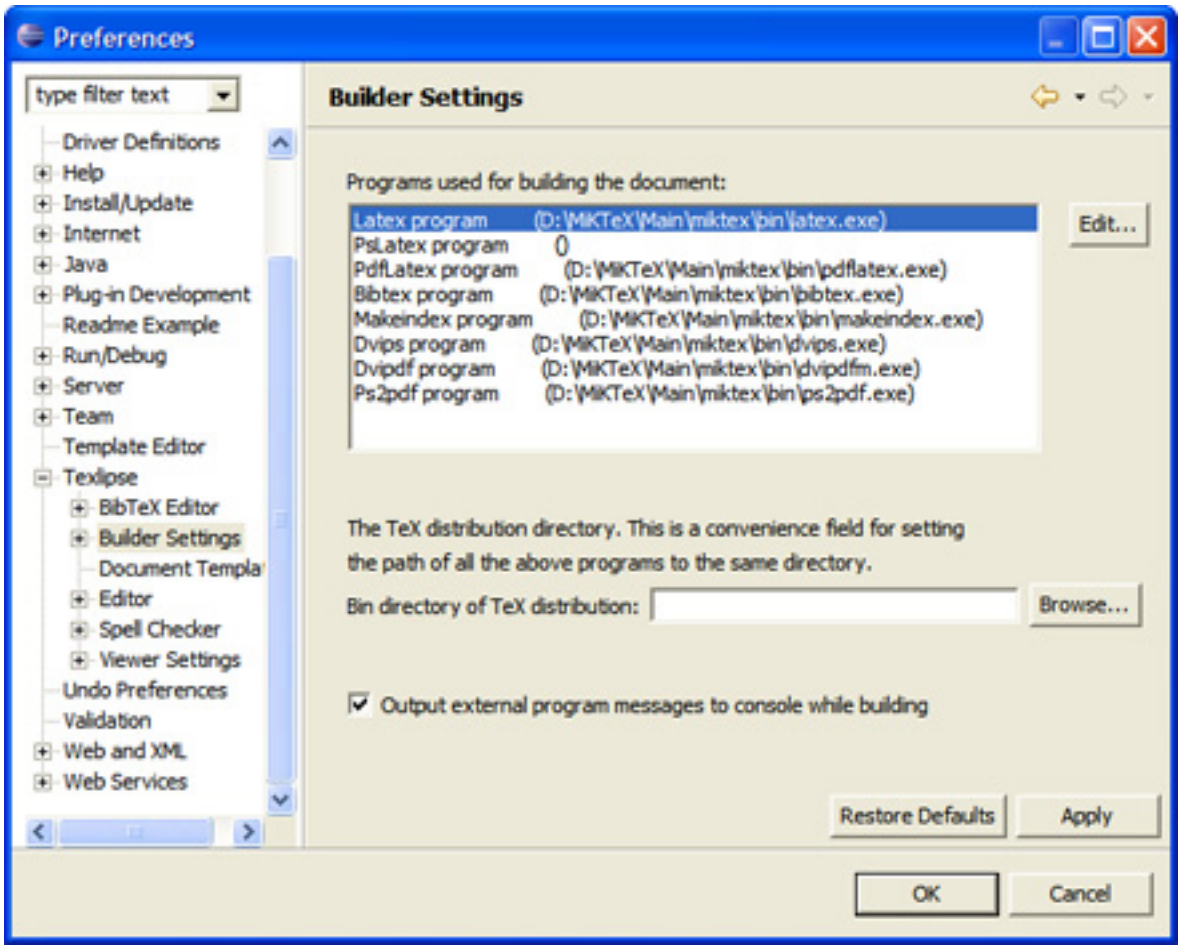

Figure 11: Builder Preferences

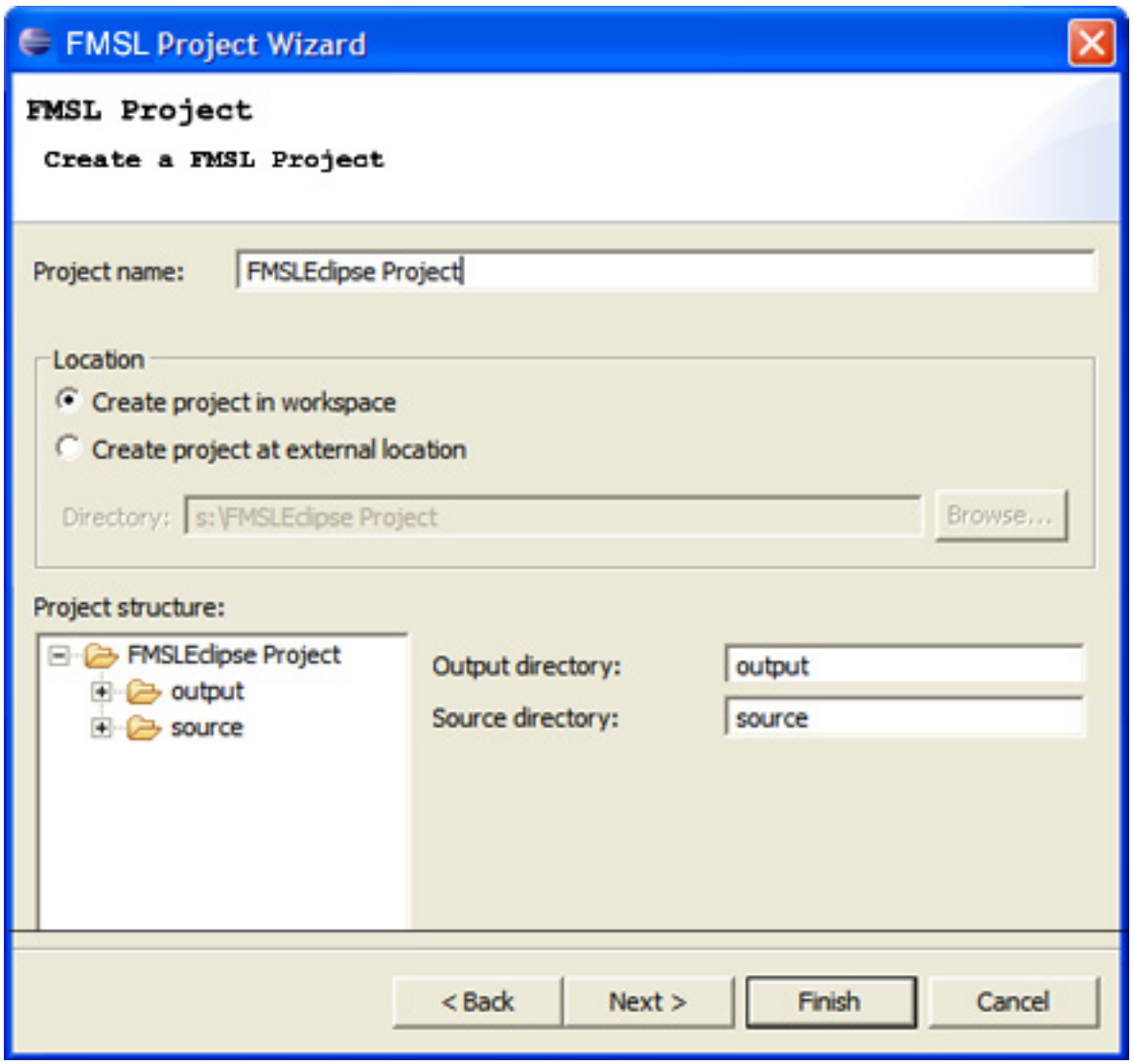

Figure 12: New Project Wizard

4 Non Functional Requirements

4.1 User Interface

FMSLEclipse shall adhear to the Eclipse User Interface Guidelines Version 2.1 found at:

http://www.eclipse.org/articles/Article-UI-Guidelines/Contents.html.

4.2 Software Interface

FMSLEclipse shall interface to the existing FMSL compiler by using a compiler executable.

4.3 Delivery

FMSLEclipse shall be delivered as one or more plugins managed by a feature. A feature specifies the prerequisites and organization of plugins. Prerequisites are feature and plugin dependencies. Each plugin may be packaged as a jar or zip file.

4.4 Installation

FMSLEclipse shall be installed through the Eclipse update manager. The update manager allows an Eclipse user to specify a URL to download features from.

A Acronyms

FMSL Formal Modeling Specification Language[1]

IDE Integrated Development Environment

RSL Requirements Specification Language

UML Unified Modeling Language

[1] FMSL used to be called Requirements Specification Language (RSL)

References