WHITE MAGIC SOFTWARE

Hierarchy: Inheritance Diagrams for Java

Software Requirements Specification

CONTENTS

1 INTRODUCTION

This document describes the complete development of a software application, as developed from scratch, tracking technical details of various aspects (including documentation) to the minute.

1.1 PURPOSE

To automatically create a visually appealing Java class inheritance hierarchy.

1.2 INSIDE SCOPE

The following features are required for the application:

- Automatically derive class dependencies.
- Generate high-quality images.
- Command-line execution or interactive user interface.
- Remove or prune packages or classes from the diagram.
- Easy to add files to analyze (directories or Java archive files).

1.3 OUTSIDE SCOPE

The following features are not required for the application:

- Call-graph hierarchy
- UML-based diagrams
- Perform analysis of the hierarchy based on:
  - Java source code
  - IDE project files
  - Ant scripts
- 3D visualizations
- Custom representation of:
  - classes, abstract classes, interfaces, and enumerated type
  - lines and arrow heads

2 Specific Requirements

This section describes the architecture and global system attributes for the project.

2.1 Architecture

Components developed for this project should be independently reusable (when possible). The major components are listed in Figure 2.1.

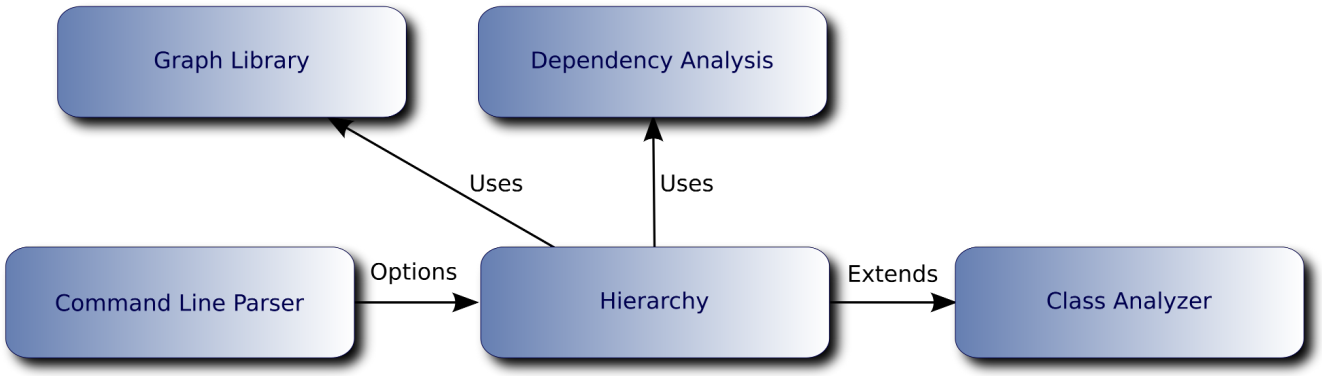

Figure 2.1: Architectural Components

2.1.1 Graph Library

The Graph Library produces an aesthetically pleasing class hierarchy.

2.1.2 Dependency Analysis

The Dependency Analysis library discovers the relationships between compiled Java classes.

2.1.3 Command Line Parser

The Command Line Parser provides Hierarchy with configuration parameters. These parameters control various aspects of the application (such as visual items, class locations, and so forth).

2.1.4 Class Analyzer

The Class Analyzer facilitates finding and analyzing class dependencies.

2.2 External Interface Requirements

This section describes the ways in which the software interacts with other systems.

2.2.1 User Interfaces

The program must be usable from the command line and through an interactive graphical user interface.

2.2.2 Hardware Interfaces

The program must be hardware-independent. For this reason the Java programming language was selected.

2.2.3 Software Interfaces

The program requires a Java Virtual Machine (JVM) to execute. Given the nature of the application, the JVM must be provided by the end user (typically a Java software developer).

2.3 Software Product Features

This section describes typical usage scenarios for the command line interface (CLI) and via graphical user interface (GUI).

2.3.1 Graphical User Interface

This is the primary way to use the program. After launching the application, the user:

1. Provides files to analyze, in any combination of:
   a) individual class files
   b) directories (recursively traversed by the application)
   c) Java archive files
2. Requests application to generate graph; the application then:
   a) performs hierarchical analysis to produce an intermediary graph format.
   b) reads default visualizations.
   c) uses the graph format to produce a visual graph.
3. Interactively makes changes to the graph.
4. Exports graph to a file.

2.3.2 Command Line Interface

Interactive usage of the application can be thought of as a visual display of functionality provided by various application components. To use the program from the command-line, the user:

1. Provides a list of classes, Java archives, and directories to analyze.
2. Provides a list of packages and classes to excluded from analysis.
3. Requests application to generate hierarchy.

2.4 Non-functional Requirements

This section describes application attributes that are independent of specific functionality.

2.4.1 Performance

The application must determine and graph the inheritance hierarchy from a set of over 2,000 classes in under 10 seconds. The class files will be located within Java archives and stored locally inside of deep directory hierarchies.

2.4.2 Maintainability

The software must include the following documents:

- Software Requirements Specifications
- Class Analyzer Technical Overview
- Hierarchy User Manual

2.4.3 Portability

The application must run on the following operating systems:

- Windows
- Linux (Unix variants)
- Mac OS X

2.4.4 Safety

The program has no safety requirements.

2.4.5 Security

The program has no security requirements. It should not be run as under an administrative account.

3 System Features

This section describes various features of the application.

3.1 Configuration

Aspects of the program that can be configured include:

- Diagram fonts and colours
- Arrow head size
- Canvas width and height
- Graph render quality

3.2 Class Inputs

The program must allow the user to indicate which classes are to be analyzed.

3.3 Path Inputs

The program must allow the user to explicitly set (and display) the search path for class files. This includes directories to traverse and Java archive files to scan.

3.4 Exclusion Inputs

The program must allow the user to exclude both packages and specific classes from being included in the graph. This should be provided to the user as a simplified regular expression pattern.

3.5 Outputs

This section describes the various outputs for the program.

3.5.1 Graph Image

The user must have the ability to save the image as a high-resolution PNG file. Both interactively and from the command line.

3.5.2 Class Path

The user must have the ability to display the class path used for finding classes. The class path can be written to standard output.

3.5.3 Font List

The user must have the ability to list all the fonts available to the program. These fonts can be written to standard output.

3.6 Graph Editing

The user must have the ability to edit the graph when running the application's graphical user interface.

- This feature does not need to be included in the first release.

4 SOFTWARE COMPONENTS

This project consists of a number of problems that have already been solved (such as reading class files, graphing hierarchies, and so forth). This project uses open source libraries to significantly reduce implementation time. However, it takes time to investigate the suitability of a library for any particular problem. This section lists the problem areas, available software components that help resolve the problem, and whether or not the library is suitable for the project.

Not all libraries were completely analysed. If a suitable library was encountered, no subsequent libraries were examined. As soon as a roadblock was encountered, the library was abandoned. Roadblocks included:

- No build script, or does not compile.
- Too many dependent libraries, or the binaries are too large.
- Insufficient documentation, or a poorly documented API.
- Too complex.

4.1 DEPENDENCY RELATIONSHIPS

This section provides a review of libraries that ascertain the relationship between Java classes. The rows for each column in Table 4.1 contain answers to the following questions:

- Interfaces – Get interfaces a class implements?
- Java – Read the latest Java bytecode format?
- JAR – Read Java archive files (including .ear, .war, .jar, and .zip)?
- Class Path – Discover dependent classes from the classpath?
- Size – How large is its binary as a Java archive (less dependent libraries)?

Library	Interfaces	Java	JAR	Class Path	Size (KB)	Suitable?
ASM	Y	Y			1912	N
BCEL	**Y**	**Y**	**Y**	**Y**	**556**	**Y**
Class Dependency Analyzer		Y	Y		4000	N
Dependency Finder		N			1017	N

Table 4.1: Java Class File Analysis Libraries

4.1.1 Dependency Finder 1.2.1-b1

Dependency Finder requires Apache's log4j and Jakarta ORO. It contains a number of applications, and does not successfully build. The last version was released in 2007, making support for Java 6 dubious.

4.1.2 Class Dependency Analyzer

Class Dependency Analyzer depends on a number of packages, including the Byte Code Engineering Library (BCEL) to display the class hierarchy in an application. No source code available.

4.1.3 ASM

Requires configuration of several libraries, including BCEL. Does not build without configuration.

4.1.4 BCEL

BCEL builds without any dependencies. The API is well documented. It took about 25 minutes to create a test harness that was able to determine its own superclass structure. BCEL is lightweight and fast.

4.2 Graph Libraries

The rows for each column in Table 4.2 contain answers to the following questions:

- Orthogonal – Allow nodes to be aligned horizontally and vertically?
- Custom – Allows for custom node representations?
- Toolkit – Swing, AWT, or SWT?

Library	Orthogonal	Custom	Toolkit	Size (KB)	Suitable?
prefuse	**Y**	**Y**	**Swing**	**909**	**Y**
JGraph			n/a		Maybe
JGraphT			Swing		Maybe
JUNG					**N**
JSDL					**N**
Grappa					**N**

Table 4.2: Graph Libraries

4.2.1 prefuse

The TreeView demonstration could be compiled and run in under 30 minutes. The library depends on the Lucene library (for text searching). It has a well documented API and several useful examples. No other graphing libraries were investigated.

4.3 Graphical User Interface Libraries

This section provides a review of various libraries that provide common graphical user interface features for desktop applications.

4.3.1 Application Framework

The only framework that would have been small or simple enough to justify inclusion was JSR 296, the Swing Application Framework. However, given the minimalist constraints of the project, it was deemed unnecessary at this point. It may be worth revisiting when it is included in Java 7. The application framework is available at:

https://appframework.dev.java.net

4.3.2 Docking

Popular docking frameworks include:

http://www.javadocking.com

http://mydoggy.sourceforge.net

4.3.3 Layout

Frameworks to ease application layout requirements include:

https://designgridlayout.dev.java.net

http://cookxml.yuanheng.org/cookswing

5 TOOLS

This section describes the software applications used for the project.

- Document writer: OpenOffice.org Writer
- Time tracker: ClockingIt
- Mock-ups: Pencil
- IDE: JDeveloper

6 Copyright

7 License

Apache License
Version 2.0, January 2004
http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to

Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

(a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

(b) You must cause any modified files to carry prominent notices stating that You changed the files; and

(c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

(d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational

purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.