

Software Visualization

Stephan Diehl

Software Visualization

Visualizing the Structure, Behaviour,
and Evolution of Software

With 124 Figures, including 75 in Colour, and 5 Tables

Author

Stephan Diehl
Universität Trier
Fachbereich Informatik
54286 Trier, Germany
diehl@uni-trier.de

Library of Congress Control Number: 2007923067

ACM Computing Classification (1998): D.2, I.3.8, J.6

ISBN 978-3-540-46504-1 Springer Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilm or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable for prosecution under the German Copyright Law.

Springer is a part of Springer Science+Business Media

springer.com

© Springer-Verlag Berlin Heidelberg 2007

The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

Typeset by the author

Production: LE-TeX Jelonek, Schmidt & Vöckler GbR, Leipzig

Cover design: KünkelLopka Werbeagentur, Heidelberg

Printed on acid-free paper 45/3100/YL - 5 4 3 2 1 0

To Christine, Luca, and Jean-Luc

Preface

Software systems are designed, implemented, tested, debugged, analyzed, and maintained by many changing developers. All these tasks can be facilitated by visualization.

In this book we give an overview of the various areas of software visualization, the art and science of generating visual representations of various aspects of software and its development process.

In contrast to visual programming and diagramming for software design, software visualization is not so much concerned with the construction, but with the analysis of programs and their development process.

So far, there exist only anthologies and proceedings about software visualization. This book is the first textbook on software visualization. Although written mostly for graduate students, the book is also a valuable resource for researchers as it provides a broad and systematic overview of the area with many pointers to literature and systems for further study.

As the field of software visualization is growing fast, the book is not meant to be comprehensive, but we have attempted to select seminal work as well as promising new approaches to illustrate some emerging principles in the field. Each chapter is followed by a list of exercises including both pen&paper exercises, as well as programming tasks.

This book is aimed at graduate students and researchers who are new to the field of software visualization. The book is meant to be read from end to end, though some readers may want to skip some of the more formal sections. Ideally, after reading the book, the reader will be able to

- identify recurring concepts in various areas of software visualization;
- understand the purpose of various visualization techniques;
- appreciate the use of visualization in software engineering.

VIII Preface

We assume that the reader will have some programming experience, preferably in Java, and some basic knowledge of software engineering terminology. No prior knowledge of software visualization is required.

Additional material related to this book, including examples and program code for exercises, can be found at the following Web address:

<http://www.eposoft.org/svbook>

March 2007

Stephan Diehl

Contents

1	Introduction	1
1.1	What Is Software Visualization?	3
1.2	Organization of This Book	4
1.3	Software Visualization and Visual Programming	6
1.4	Examples of Software Visualization Tools	7
1.4.1	StackAnalyzer: Static Program Visualization	7
1.4.2	X-Tango: Algorithm Animation	8
1.4.3	SeeSoft: Software Evolution	9
1.5	Taxonomies and Surveys	9
1.6	The Visualization Pipeline	12
	Exercises	13
2	Visualization Basics	15
2.1	Perception and Cognition	15
2.1.1	Visual Memory	16
2.1.2	The Human Eye	16
2.1.3	Light, Color, and Color Perception	17
2.1.4	Pattern Perception	17
2.1.5	Preattentive Perception	18
2.1.6	Motion Perception	19
2.1.7	Implications for the Design of Visualizations	20
2.2	Graphical Representation	21
2.2.1	Graphical Primitives and Properties	21
2.2.2	Text	22
2.2.3	Diagrams	22
2.2.4	3D Graphics and Rendering	22
2.3	General Information Visualization Techniques	25
2.3.1	Visualization of Textual Data	25
2.3.2	Graph Drawing	26

2.3.3	Visualization of Hierarchies	29
2.4	Visual Metaphors	31
2.5	Summary	32
	Exercises	32
3	Static Program Visualization	35
3.1	Textual Representations	35
3.1.1	Pretty Printing	35
3.1.2	Program as Publication	36
3.2	Diagrammatic Representations	38
3.2.1	Jackson Diagrams	38
3.2.2	Control-Flow Graphs	40
3.2.3	Nassi-Shneiderman Diagrams	45
3.2.4	Control-Structure Diagrams	47
3.3	Visualizing the Results of Program Analyses	48
3.3.1	Static Analysis	48
3.3.2	Control-Flow Analysis	49
3.3.3	Data-Flow Analysis	50
3.3.4	Examples of Visualization of Analysis Results	53
3.4	Visualizing Software Architectures	56
3.4.1	Some Familiar Architectures	57
3.4.2	The Unified Modeling Language (UML)	58
3.4.3	Software Metrics	60
3.4.4	Software Visualization and Reverse Engineering	63
3.4.5	3D and Software Architecture	71
3.5	Summary	74
	Exercises	74
4	Dynamic Program Visualization	79
4.1	Dynamic Data Acquisition	79
4.1.1	How Is Runtime Data Collected?	80
4.1.2	What Runtime Data Is Collected?	80
4.1.3	Dynamic Data Acquisition in Java	81
4.2	Visualizing Dynamics	82
4.2.1	Fundamental Techniques	82
4.2.2	A First Example	83
4.3	Dynamic Architecture Visualization	85
4.3.1	Augmenting Static Diagrams	85
4.3.2	Generating Behavior Diagrams	86
4.4	Algorithm Animation	87
4.4.1	What Is It About?	87
4.4.2	Why Do People Animate Algorithms?	88
4.4.3	A Short History of Algorithm Animation	89

4.4.4	Some Animations Produced by X-Tango	90
4.4.5	3D for Algorithm Animation	95
4.4.6	Architectures of Algorithm Animation Tools	97
4.4.7	Abstract Algorithm Animation	99
4.4.8	Learning Scenarios	102
4.4.9	A Brief Introduction to SAMBA	105
4.5	Visual Debugging – Inspecting the Program State	108
4.5.1	Interactive Visual Unfolding	109
4.5.2	Traversal-Based Visualization	110
4.5.3	Memory Graphs and Memory Slices	111
4.5.4	Reference Patterns	114
4.6	Visual Testing – Detecting Possibly Buggy Program Code	115
4.6.1	Dynamic Program Slices	115
4.6.2	Visualizing Test Case Results	117
4.6.3	Web Service Flow Patterns	122
4.7	Summary	124
	Exercises	125
5	Visualizing the Evolution of Software Systems	129
5.1	Visualizing Changes in Software Metrics	130
5.1.1	SeeSoft	131
5.1.2	Revision Towers	135
5.1.3	The Evolution Matrix	135
5.2	Visualizing Software Archives	136
5.3	Visualizing Structural Change	138
5.4	Visualizing Evolutionary Coupling	140
5.5	Visual Data Mining	144
5.6	Summary	146
	Exercises	147
6	Evaluation	149
6.1	Claims About Visualization Techniques	149
6.2	Quantitative Evaluation	149
6.3	Qualitative Evaluation	150
6.3.1	Evaluation Based on Gestalt Theory	151
6.3.2	Task-Oriented Evaluation	152
6.3.3	The Cognitive-Dimensions Framework	152
6.4	Educational Evaluation	154
6.5	Some Interesting Empirical Results	157
6.6	Summary	159
	Exercises	160

7 Conclusions 161

 7.1 The Visualization Pipeline – Revisited 161

 7.2 Further Reading and Resources 163

 7.3 The Future of Software Visualization..... 165

References 169

Index 185