RECAST: Evolution of Object-Oriented Applications

SNF 620-066077

Final Scientifique Report

September 2005 - August 2006

15 Octobre 2006

Abstract

The goal of the Recast project is to support the evolution of object-oriented applications by focusing on three main directions: reverse engineering and reengineering, versions analysis, and migration towards components. The results of the Recast project for the current period can be sorted into the following categories: software evolution analysis, system understanding, feature analysis and code analysis.

As we are hosted by the Software Composition Group of the University of Bern, we worked on topics related to the Recast project but also on topics related to the project "A Unified Approach to Composition and Extensibility"[1]. This report only lists the results obtained in the context of the Recast project over the last period *i.e.*, 2005-2006. For results of the previous periods please read the corresponding report. Scientific results not related to the Recast project are presented in the report of the project mentioned previously.

Note that some papers are not yet on print or presented and as such are not in the direct time span of this report, still we include them in this report under a specific category (See Section 2.7) since logically belong to the Recast research effort.

1 Recast Results

The results of the Recast project for the current period can be sorted into software evolution analysis, system understanding, feature analysis and code analysis.

1.1 Evolution Analysis

Understanding the evolution of an object-oriented system based on various versions of source code requires analyzing a vast amount of data. We worked on the analysis of such an evolution by designing a extensible and dedicated meta-model called Hismo. Hismo supports various analysis at various level of abstractions. It served as the basis for several applications [4][7]. We included a human dimension in the analysis of the evolution by taking into account the behavior of the developers [2][26]. We also analyzed how developers copy code [8].

[1](SNF Project No. 200020-105091/1, Oct. 2004 - Sept. 2006).

We also reported our experience as maintainers of a large open-source project so that the practical experience that we gain from the field could serve to stimulate research on software evolution [24].

1.2 System Understanding

We continue our research on the class understanding [25] but at the same time during this period we broadened our research on system wide analysis. We worked on two axes: using symbolic information and supporting package analysis.

To support the understand at the system level we developed a new visualization called Distribution Map which presents the distribution of properties over set of elements [33]. We also developed a new environment for building new visualization dynamically by scripting objects [36].

Symbolic information and semantic clustering. Understanding a software system by just analyzing the structure of the system reveals only half of the picture, since the structure tells us only how the code is working but not what the code is about. What the code is about can be found in the semantics of the source code: names of identifiers, comments etc. We analyzed how these terms are spread over the source artifacts using Latent Semantic Indexing, an information retrieval technique. We identified implementation topics by using the assumption that parts of the system that use similar terms are related [1][12][32].

Package Analysis. Understanding sets of classes, or packages, is an important activity in the development and reengineering of large object-oriented systems. Packages represent the coarse-grained structure of an application. They are the artefacts to deploy and structure software, and therefore more than a simple generalization of classes. The relationships between packages and their contained classes are key in the decomposition of an application and its (re)-modularisation. However, it is difficult to quickly grasp the structure of a package and to understand how a package interacts with the rest of the system. We tackle this problem using butterfly visualizations,*i.e.*,, dedicated radar charts built from simple package metrics based on a language-independent meta-model [5] [9]. We also proposed to take into account the context and the use interface of the packages [16] [19]. We experimented with the exploration of semantic clusters [30] and the identification of visual patterns to guide package exploration [14].

1.3 Features Analysis

Software developers are constantly required to modify and adapt features of an application in response to changing requirements. The problem is that just by reading the source code, it is difficult to determine how classes and methods contribute to the runtime behavior of features. We continued our work on feature analysis and strengthen it by proposing a consistent set of views. We proposed a deeper approach and set of views to understand the way classes and features interact with each others [31]. In particular we analyzed the evolution of features [11].

We explored the use of 3D visualization to support the management of large traces [3][27] and also how the system evolves [34]. We applied symbolic analysis to feature traces [28]. We also applied signal processing techniques to identify the similarity between traces [35].

Related but not totally linked to features, program traces as one source of precious information about the internal working of a system. We showed that we can use this information to support tests definitions. We represented trace information as logic facts and show how we could express tests using the run-time of the application [10].

1.4 Code Analysis

At the code analysis levels several paths have been followed: We used FCA to support the transformation of code to introduce traits [13], we presented some simplified implementation of object-oriented metrics [15]. We continued our work on code duplication analysis [6] taking into account the developers [8].

1.5 Others

We presented reengineering patterns to broader audiences [20] [22]. We presented our experience building the Moose environment and stressed the importance of meta-models in reverse and reengineering [21]. We participated to the definition of exchange formats in the context of code repository mining effort [29]. We participated in the definition of simple example for teaching refactorings [23].

2 Publications

2.1 Masters Theses

[1] Adrian Kuhn, Semantic Clustering: Making Use of Linguistic Information to Reveal Concepts in Source Code, Masters thesis, University of Bern, March 2006.

[2] Mauricio Seeberger, How Developers Drive Software Evolution, Masters thesis, University of Bern, Switzerland, January 2006.

[3] Christoph Wysseier, Interactive 3-D Visualization of Feature-Traces, Masters thesis, University of Berne, Switzerland, November 2005.

2.2 Ph.D. Theses

[4] Tudor Gîrba, Modeling History to Understand Software Evolution, Ph.D. thesis, University of Berne, Berne, November 2005.

[5] María Laura Ponisio, Exploiting Client Usage to Manage Program Modularity, Ph.D. thesis, University of Berne, Berne, June 2006.

2.3 Publications

Journal Papers

[6] Stéphane Ducasse, Oscar Nierstrasz and Matthias Rieger, On the Effectiveness of Clone Detection by String Matching, International Journal on Software Maintenance and Evolution: Research and Practice, vol. 18, no. 1, January 2006, pp. 37-58.

[7] Tudor Gîrba and Stéphane Ducasse, Modeling History to Analyze Software Evolution, International Journal on Software Maintenance: Research and Practice (JSME), vol. 18, 2006, pp. 207-236.

Conference Papers

[**8**] Mihai Balint, Tudor Gîrba and Radu Marinescu, How Developers Copy, Proceedings of International Conference on Program Comprehension (ICPC 2006), 2006, pp. 5665.

[**9**] Stéphane Ducasse, Michele Lanza and Laura Ponisio, Butterflies: A Visual Approach to Characterize Packages, Proceedings of the 11th IEEE International Software Metrics Symposium (METRICS'05), IEEE Computer Society, 2005, pp. 70-77.

[**10**] Stéphane Ducasse, Tudor Gîrba and Roel Wuyts, Object-Oriented Legacy System Trace-based Logic Testing, Proceedings 10th European Conference on Software Maintenance and Reengineering (CSMR 2006), IEEE Computer Society Press, 2006, pp. 35-44.

[**11**] Orla Greevy, Stéphane Ducasse and Tudor Gîrba, Analyzing Feature Traces to Incorporate the Semantics of Change in Software Evolution Analysis, Proceedings of ICSM 2005 (21th International Conference on Software Maintenance), IEEE Computer Society Press, September 2005, pp. 347356.

[**12**] Adrian Kuhn, Stéphane Ducasse and Tudor Gîrba, Enriching Reverse Engineering with Semantic Clustering, Proceedings of Working Conference on Reverse Engineering (WCRE 2005), IEEE Computer Society Press, Los Alamitos CA, November 2005, pp. 113122.

[**13**] Adrian Lienhard, Stéphane Ducasse and Gabriela Arévalo, Identifying Traits with Formal Concept Analysis, Proceedings of ASE '05 (20th Conference on Automated Software Engineering), IEEE Computer Society Press, November 2005, pp. 66-75.

[**14**] Mircea Lungu, Michele Lanza and Tudor Gîrba, Package Patterns for Visual Architecture Recovery, Proceedings 10th European Conference on Software Maintenance and Reengineering (CSMR 2006), IEEE Computer Society Press, Los Alamitos CA, 2006, pp. 183192.

[**15**] Cristina Marinescu, Radu Marinescu and Tudor Gîrba, Towards a Simplified Implementation of Object-Oriented Design Metrics, METRICS 2005, 2005, pp. 110-119.

[**16**] Laura Ponisio and Oscar Nierstrasz, Using Context Information to Re-architecture a System, Proceedings of the 3rd Software Measurement European Forum 2006 (SMEF'06), 2006, pp. 91-103.

Tool Demo

[**17**] Mauricio Seeberger, Adrian Kuhn, Tudor Gîrba and Stéphane Ducasse, Chronia: Visualizing How Developers Change Software Systems, Proceedings 10th European Conference on Software Maintenance and Reengineering (CSMR 2006), March 2006, pp. 345346, Tool demo.

[**18**] Stéphane Ducasse, Tudor Gîrba and Oscar Nierstrasz, Moose: an Agile Reengineering Environment, Proceedings of ESEC/FSE 2005, September 2005, pp. 99-102, Tool demo.

Technical Reports

[19] Laura Ponisio and Oscar Nierstrasz, Using Contextual Information to Assess Package Cohesion, Technical Report, no. IAM-06-002, University of Berne, Institute of Applied Mathematics and Computer Sciences, 2006, Technical Report. abstract PDF

Invited Papers

[20] Oscar Nierstrasz, Stéphane Ducasse and Serge Demeyer, Object-oriented Reengineering Patterns an Overview, Proceedings of Generative Programming and Component Engineering (GPCE 2005), Michael Lowry Robert Glck (Ed.), LNCS 3676, 2005, pp. 1-9, Invited paper.

[21] Oscar Nierstrasz, Stéphane Ducasse and Tudor Gîrba, The Story of Moose: an Agile Reengineering Environment, Proceedings of the European Software Engineering Conference (ESEC/FSE 2005), ACM Press, New York NY, 2005, pp. 1-10, Invited paper.

Magazine

[22] Oscar Nierstrasz, Stéphane Ducasse and Serge Demeyer, Objektorientierte Re-Engineering-Muster: ein berblick, ObjektSpektrum, vol. 2005, no. 6, 2005, pp. 46-51, German translation of Object-oriented Reengineering Patterns an Overview (GPCE 05).

Workshop papers

[23] Serge Demeyer, Filip Van Rysselberghe, Tudor Gîrba, Jacek Ratzinger, Radu Marinescu, Tom Mens, Bart Du Bois, Dirk Janssens, Stéphane Ducasse, Michele Lanza, Matthias Rieger, Harald Gall, Michel Wermelinger and Mohammad El-Ramly, The LAN-simulation: A Research and Teaching Example for Refactoring, Proceedings of IWPSE 2005 (8th International Workshop on Principles of Software Evolution), IEEE Computer Society Press, Los Alamitos CA, 2005, pp. 123-131.

[24] Marcus Denker and Stéphane Ducasse, Software Evolution from the Field: an Experience Report from the Squeak Maintainers, ERCIM workshop on Software Evolution, 2006.

[25] Stéphane Ducasse, Michele Lanza and Romain Robbes, Multi-level Method Understanding Using Microprints, Proceedings of VISSOFT 2005 (3th IEEE International Workshop on Visualizing Software for Understanding), September 2005.

[26] Tudor Gîrba, Adrian Kuhn, Mauricio Seeberger and Stéphane Ducasse, How Developers Drive Software Evolution, Proceedings of International Workshop on Principles of Software Evolution (IWPSE 2005), IEEE Computer Society Press, 2005, pp. 113122.

[27] Orla Greevy, Michele Lanza and Christoph Wysseier, Visualizing Feature Interaction in 3-D, Proceedings of VISSOFT 2005 (3th IEEE International Workshop on Visualizing Software for Understanding), September 2005, pp. 114-119.

[28] Adrian Kuhn, Orla Greevy and Tudor Gîrba, Applying Semantic Analysis to Feature Execution Traces, Proceedings of Workshop on Program Comprehension through Dynamic Analysis (PCODA 2005), November 2005, pp. 4853.

[29] Sunghun Kim, Thomas Zimmermann, Miryung Kim, Ahmed Hassan, Audris Mockus, Tudor Gîrba, Martin Pinzger, James Whitehead and Andreas Zeller, TA-RE: An Exchange Language for Mining Software Repositories, Proceedings Workshop on Mining Software Repositories (MSR 2006), 2006, pp. 22-25.

[30] Mircea Lungu, Adrian Kuhn, Tudor Gîrba and Michele Lanza, Interactive Exploration of Semantic Clusters, 3rd International Workshop on Visualizing Software for Understanding and Analysis (VISSOFT 2005), 2005, pp. 95100.

2.4 Contributions of Collaborators

Mr. Gîrba finished his PhD and integrated deeply his history model into the Moose reengineering environment. He significantly rewrote large parts of the environment and a new release of the Moose environment is included in the ESUG'2006 DVD. In addition Mr. Gîrba developed Mondrian a new visualization environment that is used by the Moose environment. Mr. Gîrba participates in the NOREX project, which is concerned with building a distributed environment for reengineering.

Mrs Ponisio finishes her PhD.

2.5 Network and Project Participation

We are currently participating in a new ERCIM working group on Software Evolution. S. Ducasse got a position as full professor at the University of Annecy and got a ANR (Agence National de la Recherche, the new French research agency) a research project related to the current one. The project is named Cook: Rearchitecturing Object-Oriented Applications.

2.6 Organized Events

- Organization of one workshop at the European Conference on Object-Oriented Programming ECOOP'2006 entitled Object - Oriented Reengineering.
- Organization of the Annual European Smalltalk User Group Conference (100 participants) at Prague.

2.7 Recast Pending Publications

Journals

[31] Orla Greevy, Stéphane Ducasse and Tudor Gîrba, Analyzing Software Evolution through Feature Views, International Journal on Software Maintenance and Evolution: Research and Practice, 2006, To appear.

[32] Adrian Kuhn, Stéphane Ducasse and Tudor Gîrba, Semantic Clustering: Identifying Topics in Source Code, Information and Software Technology, 2006, To appear.

International Conferences

[33] Stéphane Ducasse, Tudor Gîrba and Adrian Kuhn, Distribution Map, Proceedings International Conference on Software Maintainance (ICSM 2006), 2006, To appear.

[**34**] Orla Greevy, Michele Lanza and Christoph Wysseier, Visualizing live Software Systems in 3D, Proceedings of SoftVis 2006 (ACM Symposium on Software Visualization), September 2006, to appear.

[**35**] Adrian Kuhn and Orla Greevy, Exploiting the Analogy Between Traces and Signal Processing, Proceedings International Confernce on Software Maintainance (ICSM 2006), September 2006, To appear.

[**36**] Michael Meyer, Tudor Gîrba and Mircea Lungu, Mondrian: An Agile Visualization Framework, ACM Symposium on Software Visualization (SoftVis 2006), 2006, To appear.