Final Scientific Report SNF Project no. 200020-131827
"Synchronizing Models and Code"

February 1, 2013

1 Summary of results

This project explores various ways of synchronizing software source code with implicit application domain knowledge. The key results achieved in each of the tracks of this project are as follows:

1. **Dynamic meta-objects:** In this track we have explored the use of *dynamic meta-objects* to manipulate and adapt high-level representations of running software systems, thus providing an infrastructure for keeping applications in sync with their meta-level representation. We have developed Bifröst, an object-centric approach to reflection. Bifröst models meta-objects explicitly, exclusively targeting objects as the sole reflective unit of change. This model provides a unification of various approaches to reflection. We have validated Bifröst by demonstrating how it is well-suited to implement a number of key reflective applications in the domain of software development.

2. **First-class, active contexts:** Software updates must be installed dynamically to avoid costly maintenance downtimes. We have developed an approach to dynamic updates called *Theseus* that structures objects and classes from distinct versions into distinct first-class contexts. Contexts prevent inconsistent execution of code from mixed versions. The update of the software is incremental and first-class contexts control the progressive migration from one version of the software to the next. We have implemented prototypes of our approach for Smalltalk and Java.

3. **Linked, active source code:** Current software development tools fail to address synchronization of code shared between independent systems (medium-grained and medium-term as well as long-term adaptations). As software evolves, libraries, components and even fragments are frequently duplicated, adapted and specialized across software projects. We have developed a family of performant techniques for the detection of inter-system source code duplication and validated it on a large ecosystem consisting of all the projects in the Squeaksource code repository. We have established that over 14% of all methods in the repository are present in at least 2 different projects.

Results

We summarize the results obtained during the period from 2010-10-01 to 2012-12-31, and indicate here the new publications since filing the intermediate report.

Dynamic meta-objects

In this track we explored techniques to dynamically adapt running systems to changing requirements. To this end, we have developed *Bifröst*[1], a fine-grained approach to structural and behavioral reflection based on *explicit meta-objects*. Bifröst is based on the notion of *object-centric reflection*, a technique that enables any reflective change on a specific object. This simplification of the reflective targets provides a uniform and unified reflection mechanism with several advantages and benefits over the state of the art. The work

[1] `http://scg.unibe.ch/research/bifrost`

on Bifröst is described in detail in Ressia's PhD thesis [Res12]. We have explored the application of the Bifröst model to several challenging problems related to software development.

We have used Bifröst to model the program execution as explicit meta-events observable and interceptable by development tools. Chameleon is a tool built with Bifröst that provides an operational decomposition of the meta-level. Instrumentation is dedicated to generating meta-events, and is fully separated from analysis tools that selectively subscribe to these events by applying the observer pattern at the meta-level [Qua12]. In particular, this allows multiple instrumentation tasks to be performed in parallel, without interference.

Talents are dynamically composable units of reuse that can be applied to individual objects at runtime. Like *stateful traits*, talents can encapsulate sets of related methods and instance variables for reuse in new contexts. Unlike traits, talents can be applied at runtime, and can be applied to individual objects, not just to classes. This work was published in the journal Software: Practice and Experience [RGN+12].

Domain-specific profiling refers to the ability to adapt the activity of profiling an application to the needs of a particular application domain. Traditional profiling tools perform well on host language code itself but they often fail to provide meaningful results related to a domain abstractions. MetaSpy is an infrastructure that enables developers to quickly prototype new profilers for their domain-specific languages and models. Bifröst meta-objects are used for instrumenting the domain to feed the profilers with their required data. This work was published in the Journal of Object Technology (JOT) [RBNR12].

Object-centric debugging is an alternative approach to interacting with a running software system. By focusing on objects as the key abstraction, natural debugging operations can be defined to answer developer questions related to runtime behavior. The user does not have to leave the runtime environment to specify breakpoints or conditional breakpoints, the user can apply object-specific operations on objects *e.g.*, halt on next message, halt when a particular instance variable in a particular object is changed. Object-centric debugging offers more effective support for many typical developer tasks than a traditional stack-oriented debugger. This work was presented at ICSE 2012, International Conference of Software Engineering [RBN12], the top conference in this domain. The work on Object-Centric Debugging also received the Technology Innovation Award (1st prize) from the European Smalltalk User Group for 2012.

First-class, active contexts

Dynamic languages support dynamic code updates out of the box. Such code updates entail however the risk that threads execute code of mixed versions inconsistently. We have proposed an approach called *Theseus* that structures objects and classes from distinct versions into first-class contexts. Each thread runs in a specific context. Objects are migrated from one context to another automatically, based on previously defined bidirectional transformations, to avoid conflicts in the execution of inconsistent versions. The update is incremental and completes when all threads run in the latest context. Following on an initial prototype, we have implemented our approach in Java [Wer12] and Smalltalk [WLN12]. This work received the EAPLS best paper award at TOOLS 2012. We have submitted a journal article with a full quantitative evaluation of the overhead of our approach during the various phases of the update. The overhead is proportional to the number of objects shared across threads. We have shown that the approach is attractive for long-lived systems such as web servers.

Since a context encapsulates all objects of a given software version it can be seen as a special form of *ownership*. We have explored another approach that dynamically structures objects into an ownership tree and dynamically enforces encapsulation [WMN12]. We have shown that the effort required to adapt a popular web server to our approach is modest, and that our approach can detect encapsulation breaches that are sources of program errors in practice.

Linked, active source code

Hot clones is our approach to dealing with the reuse of source code across multiple evolving software systems by finding cloned code snippets across repositories and then showing their divergence to all ends of a clone [Sch12b, Sch12a]. When these clone ends belong to different projects, this eases collaboration between projects.

The techniques needed to find code clones at the scale of entire ecosystems must be highly efficient and we have proposed an approach that scales well and can detect multiple types of clones (*i.e.*, not only exact matches, but also clones that diverge from the original). The approach is based on "bad hashes": "good" hash functions normally map similar values to different hashes; in contrast we define "bad" hash functions that map similar values (*i.e.*, near clones) to the same hash. As a validation of our approach we ran an empirical study on the SqueakSource ecosystem in which we have shown that more than 14% of all methods are copied across projects [SLR12]. This is evidence for the need of mechanisms to support source code reuse at the method level and track the evolution of clones over time.

We have also carried out a brief survey to categorize the existing ways of displaying code clones to users [SS12].

Code clones often arise from the need to code minor variations of the same behavior for different application configurations. Mechanisms to extract these variation points can eliminate the need for clones in the first place. In *Seuss* we introduce a means to disentangling dependencies between source code entities with the help of dependency injection. This work was previously presented at TOOLS 2011 where it was short-listed for the best paper award. An extended journal paper has now been published [SLN12].

Agile software assessment

We have also initiated activities leading into the continuation project (SNSF project no. 200020_144126/1 — *Agile Software Assessment*[2]). This project aims to enable software developers to quickly and effectively analyze complex software systems with the help of highly customizable and adaptable tools for the construction, querying and manipulation of software models. We refer to this goal as *agile software assessment*. We have presented two invited papers that motivate the research activities we intend to carry out [Nie12, NL12].

One goal in the new project is expanding the "large-scale software analysis" beyond code clone detection to other challenges related to the evolution of entire software ecosystems. We have published a position paper detailing this approach in ERCIM News [LNS12]. We have further published several research contributions related to software ecosystems and architectural recovery [HDLL12, LLN12, LN12, RLR12].

Staff contributions

This project is supervised by Oscar Nierstrasz (Full Professor) and Mircea Lungu (Postdoc). Here we summarize the contributions of the project staff (PhD students) in the final year of this project.

- Jorge Ressia successfully defended his PhD thesis on *Object-Centric Reflection* in October 2012 [Res12]. He published two journal papers, a conference paper at ICSE, the most prestigious international conference in the domain of Software Engineering, and a further workshop paper. He co-supervised Andrea Quadri's MSc thesis on *Chameleon: Decoupling instrumentation from development tools with explicit meta-events*, a reflective approach to debugging and instrumentation based on Ressia's Bifröst framework [Qua12].

- Erwann Wernli has completed the third year of his PhD. He is working on *first-class, active contexts*. He has authored and co-authored several papers in the past year [WLN12, Wer12, WMN12]. Erwann Wernli is scheduled to defend his PhD thesis in August 2013. He is co-supervising the ongoing Bachelor thesis of Pascal Maerki.

- Niko Schwarz has completed the third year of his PhD. He co-authored one journal paper, three international conference papers, a book chapter and a white paper. He also spent four months on an internship at the Google Munich lab, where he was able to work with colleagues there on case studies related to his PhD research. He is expected to defend his PhD thesis at the end of 2013.

  He is co-supervising the Master's theses of Cedric Reichenbach, Simon Vogt and Nicole Haenni.

[2] `http://scg.unibe.ch/research/snf13`

- Andrea Caracciolo started his PhD in April 2012, and has started to work on the topic of *Architectural Monitoring* defined in the continuation SNSF project. He has been carrying out a study of "Software Architecture in the Wild" by interviewing software practitioners concerning the precise nature of architectural constraints to be found in the software systems they are responsible for developing and maintaining. We are planning to submit a paper on this topic in 2013 to a suitable international conference.

Important events

- Oscar Nierstrasz was a keynote speaker at **IWRE 2012** (3rd India Workshop on Reverse Engineering), **ICPC 2012** (20th IEEE International Conference on Program Comprehension), **BENEVOL 2012** (11th BElgian-NEtherlands software eVOLution symposium),
- Oscar Nierstrasz was Tutorials Co-Chair for ICSE 2012 (34th International Conference on Software Engineeering),
- Oscar Nierstrasz was a program committee member of the international conferences and workshops: C5, GPCE 2012, ICPC 2012, ICSM 2011, ICSM 2012, OOPSLA 2011, Onward! 2011, SE 2012, SLE 2012 Doctoral Symposium, Smalltalk Directions, XM 2012, XP 2012.
- Mircea Lungu was a keynote speaker at **Smalltalks 2012** (Sixth International Conference on Smalltalk Technologies)
- Mircea Lungu was Program Co-Chair for the Tool Demo Track of CSMR 2012 (16th European Conference on Software Maintenance and Reengineering).
- Mircea Lungu was a program committee member of: ICSE 2012 (Posters and TD Track), WCRE 2012, ICSM 2012 (TD Track), CSMR 2012
- Mircea Lungu was a reviewer for: Journal of Software Maintenance and Evolution, Empirical Software Engineering, Journal of Systems and Software, Science of Computer Programming, Foundations and Trends in Human Computer Interaction, The Journal of Object Technology.
- Jorge Ressia was a program committee member of the international workshops: DYLA 2012, IWST 2012.
- Jorge Ressia was a coreviewer for the international conferences and workshops SE 2012, DSAL 2012, ICPC 2012, ICSM 2012, GPCE 2012, SLE 2012.
- Erwann Wernli was a coreviewer for the international conferences and workshops: SE 2011, Onward! 2011, OOPSLA 2011, ICMT 2011, ICSM 2011, XP 2012, ICSM 2012, ICPC 2012, GPCE 2012, CSMR 2012.
- Niko Schwarz was a program committee member of ECOOP AEC 2013 (Artifact Evaluation Committee).
- Niko Schwarz was a coreviewer for the international conferences and workshops: SE 2012, ICPC 2012, ICSM 2012, SLE 2012.

2 Research output

All reported publications are available electronically from the project's home page:

`http://scg.unibe.ch/research/snf10`

Published papers

[HDLL12] Lile Hattori, Marco D'Ambros, Michele Lanza, and Mircea Lungu. Answering software evolution questions: An empirical evaluation. *Information and Software Technology*, 2012. URL: `http://scg.unibe.ch/archive/papers/Hatt12a.pdf`, `doi:10.1016/j.infsof.2012.09.001`.

[LLN12] Mircea Lungu, Michele Lanza, and Oscar Nierstrasz. Evolutionary and collaborative software architecture recovery with Softwarenaut. *Science of Computer Programming (SCP)*, 2012. URL: `http://scg.unibe.ch/archive/papers/Lung12b.pdf`, `doi:10.1016/j.scico.2012.04.007`.

[LN12] Mircea Lungu and Oscar Nierstrasz. Recovering software architecture with Softwarenaut. *ERCIM News*, 88, January 2012. URL: `http://ercim-news.ercim.eu/en88/special/recovering-software-architecture-with-softwarenaut`.

[LNS12] Mircea Lungu, Oscar Nierstrasz, and Niko Schwarz. Big software data analysis. *ERCIM News*, 89, April 2012. URL: `http://ercim-news.ercim.eu/en89/special/big-software-data-analysis`.

[Nie12] Oscar Nierstrasz. Agile software assessment with Moose. *SIGSOFT Softw. Eng. Notes*, 37(3):1–5, May 2012. URL: `http://scg.unibe.ch/archive/papers/Nier12aASA.pdf`, `doi:10.1145/180921.2180925`.

[NL12] Oscar Nierstrasz and Mircea Lungu. Agile software assessment. In *Proceedings of International Conference on Program Comprehension (ICPC 2012)*, pages 3–10, 2012. URL: `http://scg.unibe.ch/archive/papers/Nier12bASA.pdf`, `doi:10.1109/ICPC.2012.6240507`.

[RBN12] Jorge Ressia, Alexandre Bergel, and Oscar Nierstrasz. Object-centric debugging. In *Proceeding of the 34rd international conference on Software engineering*, ICSE '12, 2012. URL: `http://scg.unibe.ch/archive/papers/Ress12a-ObjectCentricDebugging.pdf`, `doi:10.1109/ICSE.2012.6227167`.

[RBNR12] Jorge Ressia, Alexandre Bergel, Oscar Nierstrasz, and Lukas Renggli. Modeling domain-specific profilers. *Journal of Object Technology*, 11(1):1–21, April 2012. URL: `http://www.jot.fm/issues/issue_2012_04/article5.pdf`, `doi:10.5381/jot.2012.11.1.a5`.

[RGN+12] Jorge Ressia, Tudor Gîrba, Oscar Nierstrasz, Fabrizio Perin, and Lukas Renggli. Talents: an environment for dynamically composing units of reuse. *Software: Practice and Experience*, 2012. URL: `http://scg.unibe.ch/archive/papers/Ress12eTalentsSPE.pdf`, `doi:10.1002/spe.2160`.

[RLR12] Romain Robbes, Mircea Lungu, and David Roethlisberger. How do developers react to API deprecation? The case of a Smalltalk ecosystem. In *Proceedings of the 20th International Symposium on the Foundations of Software Engineering (FSE'12)*, pages 56:1 – 56:11, 2012. URL: `http://scg.unibe.ch/archive/papers/Rob12aAPIDeprecations.pdf`, `doi:10.1145/2393596.2393662`.

[RPR12] Jorge Ressia, Fabrizio Perin, and Lukas Renggli. Suicide objects. In *Proceedings of the 6th Workshop on Dynamic Languages and Applications*, DYLA '12, pages 1:1–1:2, New York, NY, USA, 2012. ACM. URL: http://scg.unibe.ch/archive/papers/Ress12cSuicideObjects.pdf, doi:10.1145/2307196.2307197.

[Sch12a] Niko Schwarz. Hot clones: A shotgun marriage of Search-Driven development and clone management. In *2012 16th European Conference on Software Maintenance and Reengineering*, volume 0, pages 513–515, Los Alamitos, CA, USA, March 2012. IEEE. URL: http://scg.unibe.ch/archive/papers/Schw12d-clones-shotgun-marriage.pdf, doi:10.1109/CSMR.2012.67.

[Sch12b] Niko Schwarz. Hot clones: Combining search-driven development, clone management, and code provenance. In *2012 34th International Conference on Software Engineering (ICSE)*, pages 1628–1629. IEEE, June 2012. URL: http://scg.unibe.ch/archive/papers/Schw12c-clones-provenance.pdf, doi:10.1109/ICSE.2012.6227221.

[SLN12] Niko Schwarz, Mircea Lungu, and Oscar Nierstrasz. Seuss: Decoupling responsibilities from static methods for fine-grained configurability. *Journal of Object Technology*, 11(1), 2012. URL: http://www.jot.fm/issues/issue_2012_04/article3.pdf, doi:10.5381/jot.2012.11.1.a3.

[SLR12] Niko Schwarz, Mircea Lungu, and Romain Robbes. On how often code is cloned across repositories. In *Proceedings of the 2012 International Conference on Software Engineering*, ICSE 2012, pages 1289–1292, Piscataway, NJ, USA, 2012. IEEE Press. URL: http://scg.unibe.ch/archive/papers/Schw12a-scalable-clones.pdf.

[SS12] Sandro Schulze and Niko Schwarz. How to make the hidden visible — code clone presentation revisited. In Rainer Koschke, Ira D. Baxter, Michael Conradt, and James R. Cordy, editors, *Software Clone Management Towards Industrial Application (Dagstuhl Seminar 12071)*, volume 2, pages 35–38. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, June 2012. URL: http://scg.unibe.ch/archive/papers/Schu12a-clone-representation.pdf.

[Wer12] Erwann Wernli. Theseus: Whole updates of Java server applications. In *Proceedings of HotSWUp 2012 (Fourth Workshop on Hot Topics in Software Upgrades)*, pages 41–45, June 2012. URL: http://scg.unibe.ch/archive/papers/Wern12b.pdf, doi:10.1109/HotSWUp.2012.6226616.

[WLN12] Erwann Wernli, Mircea Lungu, and Oscar Nierstrasz. Incremental dynamic updates with first-class contexts. In *Objects, Components, Models and Patterns, Proceedings of TOOLS Europe 2012*, pages 304–319, 2012. URL: http://scg.unibe.ch/archive/papers/Wern12a.pdf, doi:10.1007/978-3-642-30561-0_21.

[WMN12] Erwann Wernli, Pascal Maerki, and Oscar Nierstrasz. Ownership, filters and crossing handlers: flexible ownership in dynamic languages. In *Proceedings of the 8th symposium on Dynamic languages*, DLS '12, pages 83–94, New York, NY, USA, 2012. ACM. URL: http://scg.unibe.ch/archive/papers/Wern12c.pdf, doi:10.1145/2384577.2384589.

Theses

[Qua12] Andrea Quadri. Chameleon: Decoupling instrumentation from development tools with explicit meta-events. Master's thesis, University of Bern, January 2012. URL: http://scg.unibe.ch/archive/masters/Quad12a.pdf.

[Res12] Jorge Ressia. *Object-Centric Reflection*. Phd thesis, University of Bern, October 2012. URL: http://scg.unibe.ch/archive/phd/ressia-phd.pdf.