Final Scientific Report
SNF Project no. 200020-105091/1
"A Unified Approach to Composition and Extensibility"

November 6, 2006

a) Summary of results

This project focuses on the design and implementation of programming language mechanisms and concepts to enable and control extensibility of complex software systems. Significant results have been achieved in the four areas covered by this project. (i) TRAITS offer a fine-grained mechanism for composing classes from reusable components. We have developed an approach to incorporate TRAITS into statically-typed languages like Java and C#. The TRAITS model has been extended to incorporate state, while retaining the key properties of the stateless TRAITS model. TRAITS have also been included in the current distribution of Squeak, an open-source Smalltalk system. (ii) CLASSBOXES offer a module system that confines the visibility of extensions to selected clients. We have prototyped an environment to support the development of CLASSBOXES from the programmer's perspective. (iii) DIAMOND concerns foundational work in the development of programming mechanisms to support software evolution. We have developed a framework to support high-level behavior reflection, a "back-in-time" debugger capability, and an approach to reason about aliasing in evolving systems. (iv) EG is a framework for composing unit tests in a rigorous way. We have elaborated the EG meta-model and developed an experimental environment to support the construction of tests and units under test according to this meta-model.

Results

We present the results obtained during the period from 2005-10-01 to 2006-09-30 in the four areas covered by this project: TRAITS, CLASSBOXES, DIAMOND and Composable Tests.

Traits

TRAITS are a mechanism for defining classes in an object-oriented language from fine-grained reusable components.

The work on TRAITS has considerably matured, by both research extensions and industrial acceptance. In the original TRAITS model, a TRAIT bundled a collaborating set of methods that can be used to compose classes while avoiding problems of fragility brought by multiple inheritance and mixin [DNS+06]. In this model, state can only be accessed via accessor methods that are required by the traits and are implemented in the classes. We have developed an extension to the model to also consider state in the definition of traits [BDNW06]. Stateful traits allow for the definition of variables, and unless explicitly stated otherwise by the designer, the visibility of the variables is restricted to the defining trait. In this way, we also solve the problem of name clashes.

Most implementations of traits have focused on dynamically-typed languages. That is why we focused on how to best introduce traits to statically-typed languages. We have conducted a project in partnership with Microsoft to explore and evaluate the problems residing in integrating TRAITS into CSharp 2.0 [Rei05]. We have designed an extension to the CSharp syntax and built a successful preprocessor to express the TRAITS into regular CSharp 2.0 code. We argued and showed that the flattening property of traits

should be used as a guiding principle for any attempt to add traits to statically-typed languages. We demonstrated how this principle applies to Featherweight-Trait Java, a conservative extension to Featherweight Java [NDS06].

We have led the work on integrating TRAITS into Squeak, and starting from version 3.9 TRAITS are part of the core of the language [DD06].

Classboxes

CLASSBOXES offer a module system that supports local rebinding by confining the visibility of extensions to the scope of a given CLASSBOX. In this way clients of existing classes are protected from potentially disruptive extensions that need not concern them.

The concept of scoped extensions extended beyond the metaphor used in the current code browsers. Therefore, we developed an advanced prototype browser for supporting CLASSBOXES [Hal05]. The browser is based on a generic meta driven browser [BDPW06].

We have performed an in-depth analysis on module diversity and we have built a formalism and a taxonomy through which different module systems can be compared [BDN05a]. Our goal was to provide for a common foundation for expressing and comparing different module systems. To validate our approach, we have expressed the module systems of several languages (*e.g.* Java packages, C# namespaces, CLASSBOXES) and summarized the results in a taxonomy.

We have developed Classbox/J, and have also performed an extended experiment on a large Java case study to show both the scalability of the approach and its applicability on statically typed languages [BDN05b]

The work on CLASSBOXES conducted by Dr. Alexandre Bergel during his PhD and part of this project received the prestigious Ernst Denert-Stiftung Prize for Software Engineering 2006[1].

Diamond

"DIAMOND" refers to our foundational work towards the design of a "programming language in the sky" to support dynamic software evolution. We have made considerable progress towards understanding the kinds of mechanisms such a language needs to support.

We have continued the work on runtime bytecode transformation [DDT06]. The ByteSurgeon system has been used for a number of practical systems (*e.g.* test coverage analysis). The availability of ByteSurgeon improves the capabilities of Squeak for prototyping programming languages and tools [BD06]. Using ByteSurgeon we built a back-in-time debugger, a debugger that addresses the non-locality problem of runtime errors and their causes: the cause of a runtime error might be out of the scope of the current stack [HDD06]. The debugger not only stores the state for the current stack, but for the entire execution trace, and it allows us to navigate the entire execution history to look for the problem [Hof06].

Dynamic, unanticipated adaptation of running systems is of interest in a variety of situations, ranging from functional upgrades to on-the-fly debugging or monitoring of critical applications. Based on our work on Bytecode manipulation, we have implemented a framework (Geppetto) that allows these kinds of change by providing unanticipated partial behavioral reflection for Smalltalk [RDT06]. Geppetto combines the selectivity and efficiency of partial behavioral reflection with the dynamic nature of ByteSurgeon that does not require preparation of the code of any sort [Röt06].

One of the primary challenges in building and evolving object-oriented systems is reasoning about aliasing between objects. During the last ten years, much research has been carried out in the field of aliasing control (ownership types or confined types to name only two). However, almost all approaches rely on type annotations and static type systems and therefore (i) cannot be applied to dynamically typed languages and (ii) put burden on the developer because of complex annotation of source code. Recently we have started investigating into a generic, run-time based model of alias control [NDGL06].

[1] www.denert-stiftung.de

Composable Tests

In addition to expressing functional requirements that software units should fulfil, tests also provide concrete examples of how the code can be used. This idea is at the root of the PhD thesis of Markus Gälli [GÖ6]. A meta-model called EG is developed in which the examples are linked with the methods under test and they provide reusable pieces of code and of assumptions.

We have implemented the EG meta-model, and we have built a prototype IDE, called EGBROWSER [Wam06]. We have carried out an initial empirical study to evaluate the effectiveness of our example-driven model. The study provides some evidence that the meta-model helps to improve programmer productivity when developing in a test-driven fashion.

From a different perspective, we have also experimented with writing tests for legacy systems. The main problem with this is that legacy systems are typically purely understood and modularized, hence bringing the system into the wanted state is difficult. As a result, we have expressed tests as logic rules directly on top of collected traces [DGW06].

Staff contributions

- Marcus Denker is in the third year of the PhD. He extended the work on reflection embodied in BYTESURGEON [DDT06], he initiated a project to build a high level interface for reflection control [Röt06], and participated in building a back-in-time debugger [HDD06]. He was a key person responsible for the release of the Squeak 3.9, an open source Smalltalk system [DD06].
- Markus Gälli has completed his PhD on Composable Tests and will defend his thesis in November 2006.
- Adrian Kuhn is in his first year of the PhD. He explored the use of signal processing to analyze traces [KG06], and worked on a generic visualization to represent software systems [DGK06].
- Adrian Lienhard is in the 2nd year of the PhD. During the past year, he continued the work on capturing and modeling aliases as first class entities[NDGL06].

Changes to the research plan

No major changes have occurred in the research plan.

Important events

- Adrian Kuhn presented two papers at the International Conference on Software Maintenance (ICSM 2006) [KG06, DGK06].
- Marcus Denker presented papers at International ERCIM Workshop on Software Evolution [DD06], 14th International Smalltalk Conference [Röt06], and NET.ObjectDays 2006 [HDD06].
- Markus Gälli presented two papers at the 4th International Conference on Creating, Connecting and Collaborating with Computers (C5 2006) [GNS06, TG06].
- Alexandre Bergel has won the Ernst Denert-Stiftung Prize for Software Engineering 2006 for his PhD thesis on Classboxes.
- Oscar Nierstrasz presented a paper at the Workshop on Revival of Dynamic Languages – co-located with ECOOP 2006 (RDL 2006) [NDGL06].
- Oscar Nierstrasz was a keynote speaker at NODe 2006 (NET.ObjectDays 2006 – Erfurt, Germany, Sept. 18-21, 2006), where he gave the presentation "Taming Software Change", summarizing current and ongoing research funded by this SNF project and the successor project (SNF 200020-113342, *Analyzing, Capturing and Taming Software Change*).

- Oscar Nierstrasz was Program Chair of MoDELS 2006 (9th International Conference on Model Driven Engineering Languages and Systems – Genoa, Italy, Oct 1-6, 2006).
- Oscar Nierstrasz was PC Member of:
  - EVOL 2006 (International ERCIM Workshop on Software Evolution – Lilles, France, April 6-7, 2006)
  - DLS05 (Dynamic Languages Symposium at OOPSLA 2005 – San Diego, Oct 18, 2005)
  - MoDELS / UML 2005 (ACM/IEEE 8th International Conference on Model Driven Engineering Languages and Systems – Jamaica, Oct 2-7, 2005)
- Stéphane Ducasse was PC Member of:
  - IDM 2006 (Journées sur l'Inginérie Dirigée par les Modeèles – Lille, France, 27-28 June, 2006)
  - CSMR 2006 (10th European Conference on Software Maintenance and Reengineering – Bari, Italy, March 22-24, 2006)
  - ICPC 2006 (4th IEEE International Conference on Program Comprehension – Athens, Greece, June 14-16, 2006)
  - LMO 2006 (Conférence sur les Languages et Modèles à Objets – Nîmes, France, March 22-24, 2006)
  - MoDELS 2006 (9th International Conference on Model Driven Engineering Languages and Systems – Genoa, Italy, Oct 1-6, 2006)
  - Organizer of International Workshop on Visualizing Software for Understanding and Analysis 2005 (Vissoft)
  - DLS05 (Dynamic Languages Symposium at OOPSLA 2005 – San Diego, Oct 18, 2005)
  - International Conference on Software Maintenance (ICSM 2005)

b) Publications

Published papers are annexed to this report. They are all available electronically as PDF files at the following url:

www.iam.unibe.ch/∼scg/cgi-bin/scgbib.cgi?snf06

Please note that theses and student projects are *not* included with this report, but are nevertheless available electronically from the above URL.

Papers published in the context of the RECAST project are also not included with this report. They have been previously submitted with the intermediate report for RECAST. Electronic versions are available at:

www.iam.unibe.ch/∼scg/cgi-bin/scgbib.cgi?recast06

Published papers

[BD06] Alexandre Bergel and Marcus Denker. Prototyping languages, related constructs and tools with Squeak. In *Proceedings of the Workshop on Revival of Dynamic Languages (co-located with ECOOP'06)*, July 2006.

[BDN05a] Alexandre Bergel, Stéphane Ducasse, and Oscar Nierstrasz. Analyzing module diversity. *Journal of Universal Computer Science*, 11(10):1613–1644, November 2005.

[BDN05b] Alexandre Bergel, Stéphane Ducasse, and Oscar Nierstrasz. Classbox/J: Controlling the scope of change in Java. In *Proceedings of Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA'05)*, pages 177–189, New York, NY, USA, 2005. ACM Press.

[DDT06] Marcus Denker, Stéphane Ducasse, and Éric Tanter. Runtime bytecode transformation for Smalltalk. *Journal of Computer Languages, Systems and Structures*, 32(2-3):125–139, July 2006.

[DNS+06] Stéphane Ducasse, Oscar Nierstrasz, Nathanael Schärli, Roel Wuyts, and Andrew Black. Traits: A mechanism for fine-grained reuse. *ACM Transactions on Programming Languages and Systems*, 28(2):331–388, March 2006.

[GNS06] Markus Gaelli, Oscar Nierstrasz, and Serge Stinckwich. Idioms for composing games with EToys. In *Proceedings of C5 2006 (The Fourth International Conference on Creating, Connecting and Collaborating through Computing)*, pages 222–321, January 2006.

[HDD06] Christoph Hofer, Marcus Denker, and Stéphane Ducasse. Design and implementation of a backward-in-time debugger. In *Proceedings of NODE'06*, volume P-88 of *Lecture Notes in Informatics*, pages 17–32. Gesellschaft für Informatik (GI), September 2006.

[KG06] Adrian Kuhn and Orla Greevy. Summarizing traces as signals in time. In *Proceedings IEEE Workshop on Program Comprehension through Dynamic Analysis (PCODA 2006)*, pages 01–06, Los Alamitos CA, October 2006. IEEE Computer Society Press.

[NDGL06] Oscar Nierstrasz, Marcus Denker, Tudor Gîrba, and Adrian Lienhard. Analyzing, capturing and taming software change. In *Proceedings of the Workshop on Revival of Dynamic Languages (co-located with ECOOP'06)*, July 2006.

[NDS06] Oscar Nierstrasz, Stéphane Ducasse, and Nathanael Schärli. Flattening Traits. *Journal of Object Technology*, 5(4):129–148, May 2006.

[TG06] Florian Thalmann and Markus Gaelli. Jam Tomorrow: Collaborative music generation in Croquet using OpenAL. In *Proceedings of C5 2006 (The Fourth International Conference on Creating, Connecting and Collaborating through Computing)*, pages 73–78, January 2006.

[TGDB06] Éric Tanter, Kris Gybels, Marcus Denker, and Alexandre Bergel. Context-aware aspects. In *Proceedings of the 5th International Symposium on Software Composition (SC 2006)*, LNCS 4089, pages 227–242, Vienna, Austria, March 2006.

Theses and Student projects

[GÖ6] Markus Gälli. *Modeling Examples to Test and Understand Software*. PhD thesis, University of Berne, November 2006.

[Hal05] Niklaus Haldimann. A sophisticated programming environment to cope with scoped changes. Informatikprojekt, University of Bern, December 2005.

[Hof06] Christoph Hofer. Implementing a backward-in-time debugger. Master's thesis, University of Bern, September 2006.

[Rei05] Stefan Reichhart. A prototype of Traits for C#. Informatikprojekt, University of Bern, 2005.

[Ren06] Lukas Renggli. Magritte – meta-described web application development. Master's thesis, University of Bern, June 2006.

[Röt06] David Röthlisberger. Geppetto: Enhancing Smalltalk's reflective capabilities with unanticipated reflection. Master's thesis, University of Bern, January 2006.

[Wam06] Rafael Wampfler. Eg – a meta model and editor for unit tests. Master's thesis, University of Bern, November 2006.

Selected RECAST publications

[DD06] Marcus Denker and Stéphane Ducasse. Software evolution from the field: an experience report from the Squeak maintainers. In *ERCIM workshop on Software Evolution*, 2006.

[DGK06] Stéphane Ducasse, Tudor Gîrba, and Adrian Kuhn. Distribution map. In *Proceedings International Conference on Software Maintainance (ICSM 2006)*, pages 203–212, Los Alamitos CA, 2006. IEEE Computer Society.

[DGW06] Stéphane Ducasse, Tudor Gîrba, and Roel Wuyts. Object-oriented legacy system trace-based logic testing. In *Proceedings 10th European Conference on Software Maintenance and Reengineering (CSMR 2006)*, pages 35–44. IEEE Computer Society Press, 2006.

[KG06] Adrian Kuhn and Orla Greevy. Exploiting the analogy between traces and signal processing. In *Proceedings IEEE International Confernce on Software Maintainance (ICSM 2006)*, Los Alamitos CA, September 2006. IEEE Computer Society Press.

c) Publications in press

Publications to appear

[BDNW06] Alexandre Bergel, Stéphane Ducasse, Oscar Nierstrasz, and Roel Wuyts. Stateful traits. In *Proceedings of the International Smalltalk Conference*, LNCS. Springer, 2006. To appear.

[BDPW06] Alexandre Bergel, Stéphane Ducasse, Colin Putney, and Roel Wuyts. Meta-driven browsers. In *Proceedings of the International Smalltalk Conference (ISC 2006)*, LNCS. Springer, 2006. To appear.

[RDT06] David Röthlisberger, Marcus Denker, and Éric Tanter. Unanticipated partial behavioral reflection. In *Proceedings of ISC 2006 (International Smalltalk Conference), LNCS*, 2006. To appear.