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

November 7, 2005

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. The results relate to (i) TRAITS– a mechanism for composing data abstractions from fine-grained components, (ii) CLASSBOXES– a module system for controlling the scope of changes, (iii) DIAMOND– a new programming language intended to support software evolution, and (iv) composable tests – a rigorous approach to organize and structure the test cases for complex systems. Some of the most significant results include (i) both formal specification and analysis of TRAITS, and practical experience using traits to refactor a Smalltalk kernel, (ii) formal specification of CLASSBOXES as well as practical experience applying classboxes to various large case studies, (iii) the development of a new framework to support run-time behavioural reflection for DIAMOND, and (iv) the elaboration of a taxonomy of unit tests and the development of a corresponding framework and tool for composing tests from simpler tests.

Results

We present the results obtained during the period from 2004-10-01 to 2005-09-30 in the four areas covered by this project: TRAITS, CLASSBOXES, DIAMOND and Composable Tests.

Traits

TRAITS offer a means to construct classes in object-oriented programming languages from fine-grained components, called TRAITS, which bundle a reusable set of collaborating methods. TRAITS avoid fragility problems arising from other approaches, like multiple inheritance and mixins, by offering the composing class mechanisms to control the way in which TRAITS are composed.

The TRAITS model and its validation have been published in Nathanael Schärli's PhD thesis [Sch05] and in a forthcoming TOPLAS paper [DNS+05].

An important property of the TRAITS model is that TRAITS can be *flattened*, *i.e.*, any program written with TRAITS is strictly equivalent to another program written without TRAITS, but possibly introducing duplicated code. We have used the flattening property to investigate the integration of TRAITS into statically-typed languages like Java and C#, by extending a formal calculus called *Featherweight Java* with mechanisms to support TRAITS. Programs in the extended language can be flattened to the base language while respecting type constraints [NDS05, NDS06].

We have used TRAITS to refactor and bootstrap the new kernel of Squeak Smalltalk [Lie04]. This effort represents a serious and performance-critical application of TRAITS. This experiment has allowed us to express the new TRAITS-aware kernel in itself, this following the fundamental principle of a reflective language – using the features of the language to define the behavior of the language itself. This has not only been an interesting experience in respect to bootstrapping a language, it also helped us to assess TRAITS: refactoring the kernel with TRAITS not only helped us to eliminate duplicated code, but it also made the code more understandable and maintainable. Furthermore it facilitates experimentation with the language because the different aspects of the kernel are now available as TRAITS for recomposition.

We have also shown how TRAITS can be used to safely compose metaclass features, a well-known problem in reflective languages like Smalltalk [DSW05].

Over the last two years, TRAITS have been introduced into languages such as Perl 6, Squeak, Scala and Fortress. With the increasing number of languages incorporating TRAITS the question of refactoring existing libraries and applications with TRAITS gains importance. Up until now, the identification of TRAITS has been a complex, manual task. We have developed a semi-automatic approach based on Formal Concept Analysis (FCA) to identify TRAITS and restructure the inheritance hierarchy of a system [LDA05]. This work combines our expertise in the fields of language design and re-engineering [NDD05].

Classboxes

CLASSBOXES represent a module system allowing an application to be modified in a non-invasive way. Modifications are only visible within a restricted scope called a CLASSBOX. The definition of the CLASSBOX module system and its semantics are stable and mature [BDNW05]. A large case study based on the Swing Java library shows that CLASSBOXES can help developers to define more robust software extensions [BDN05b]. A formal framework was defined to compare and contrast features of CLASSBOXES with those of other module systems [BDN05a].

Various extensions of CLASSBOXES have also been investigated. By combining CLASSBOXES with TRAITS, it is possible to package cross-cutting changes [BD05b]. This work resulted in a collaboration with the Ecole des Mines de Nantes, France [MBCD05]. Dynamic CLASSBOXES allow software changes to be loaded and unloaded while the software is running [BD05a].

Diamond

DIAMOND is the working name for a new programming language that will provide mechanisms and features to support software evolution. DIAMOND will build on previous and ongoing work, including TRAITS, CLASSBOXES and PICCOLA.

In two position papers [ND04, NBD+05] we outline the various features that characterize DIAMOND, and that we feel are essential to support software evolution: (i) "always up" systems eliminating of the distinction between "compile-time" and "run-time", (ii) first-class namespaces to manage the scope of changes, (iii) pluggable type systems to enable reasoning about different kinds of static aspects of evolving systems, (iv) reflection on demand to enable dynamic evolution, and (v) executable documentation in the form of example objects.

A key concept in DIAMOND is that of *first-class namespaces*, a concept originally developed within PICCOLA [AN05]. The dynamic use of CLASSBOXES mentioned in the previous section goes in the direction of supporting dynamic evolution in a controlled and well-delimited fashion.

BYTESURGEON [DDT05] enables transformation of Smalltalk bytecodes. Transformations can be done at runtime, without stopping program execution and without having the program's textual code available. BYTESURGEON is now being used for a number of experiments, one of which is GEPPETTO, a Meta Object Protocol that provides fine-grained and unanticipated reflective facilities for Smalltalk. This will be the key to supporting reflection on demand in DIAMOND.

We have started some activity investigating various approaches to supporting "always up" systems, focussing on transparent object persistence. Work on pluggable types is still only at the conceptual level, and no concrete project has been started yet. Work on example objects is related to the next section.

Composable Tests

We claim that unit tests implicitly relate both to the unit under test and to other unit tests, and that making this relations explicit helps the developer to co-evolve the system together with its tests. We therefore manually and semi-automatically analysed a test suite consisting of more than 1000 unit tests and built a taxonomy of unit tests with respect to composability and traceability. It turned out that most unit tests of this large case study either directly focused on single methods — we call these *one-method commands* —

or were decomposable into such one-method commands [GLN05]. We believe this insight is the key to designing a test suite as a collection of composable tests.

We have investigated to what extent we can automatically detect the focused method under test in several Java case studies [Mar05].

We also built a prototype of an IDE for test-driven development in which we introduced a simple denotation mechanism to allow the developer to navigate between unit tests and its methods under test, and also to compose unit tests from each other [GND04]. While this IDE works well for creating new unit tests together with a new system, we also need support for understanding and refactoring legacy test suites. We therefore described how we can detect candidates of (re-)composable test cases by partially ordering and visualizing their sets of covered method signatures, and we presented techniques to refactor these unit tests accordingly. [GGN05]

Staff contributions

- Alexandre Bergel has completed his PhD on Classboxes and will defend his thesis in November 2005. He has published numerous conference and journal papers over the past year [BDNW05] [BDN05a] [BD05b] [BD05a] [MBCD05] including a paper at OOPSLA 2005 [BDN05b] (OOPSLA is one of the highest impact conferences in the domain, with an acceptance rate typically around 10%).

- Marcus Denker is in the second year of his PhD work, and making solid progress. He has developed BYTESURGEON [DDT05], tool for runtime transformation of Smalltalk bytecodes. BYTESURGEON is being used to develop GEPPETTO. Denker is participating actively in the development of DIAMOND [NBD+05] [ND04].

- Markus Gälli is in the final year of his PhD. He has developed and published a taxonomy of unit tests [GLN05] and demonstrated an approach based on this taxononomy for composing unit tests. [GGN05] [GND04].

- Adrian Lienhard is in his first year of the PhD. He has worked on bootstrapping Squeak Smalltalk with TRAITS [Lie04]. He has also been applying Formal Concept Analysis to automatically identify TRAITS in legacy source code, and a paper on this subject has been accepted for presentation at ASE 2005. Lienhard has started to investigate the subject of alias control for dynamically typed languages.

- Nathanael Schärli completed his PhD on TRAITS earlier this year [Sch05] and has left the University of Bern for a research position at Google Switzerland's Zurich research lab. Numerous publications have resulted from this work [DSW05] [NDS05] [SBD04] including a forthcoming journal paper [DNS+05] to appear in ACM TOPLAS.

Changes to the research plan

No major changes have occurred in the research plan.

Important events

- Alexandre Bergel presented papers on CLASSBOXES at Net.ObjectDays (NODE'05) [BD05b], and OOPSLA 2005 [BDN05b].

- Marcus Denker presented a paper on BYTESURGEON at ESUG 2005 (European Smalltalk Users' Group) [DDT05].

- Markus Gälli presented papers at ESUG 2005 (European Smalltalk Users' Group) [GLN05] and at SPLiT 2006 (2nd International Workshop on Software Product Line Testing) [GGN05].

- Nathanael Schärli presented a paper on TRAITS at OOPSLA 2004 [SBD04].

- Oscar Nierstrasz presented a position paper at the OOPSLA 2004 Workshop on Revival of Dynamic Languages [ND04] and an invited keynote paper at Software Composition 2005 [NBD+05].
- Oscar Nierstrasz was an invited speaker at
  - UML 2004 (Oct. 11-15, 2004 – Lisbon, Portugal)
  - ESEC/FSE'05 (European Software Engineering Conference – Lisbon, Portugal, Sept. 5-9, 2005)
  - GPCE'05 (Generative Programming and Component Engineering – Tallinn, Estonia, Sep 29 - Oct 1, 2005)
- Oscar Nierstrasz was PC Member of
  - SC 2005 (Software Composition – Edinburgh, Scotland, April 9, 2005; co-located with ETAPS 2005)
  - IDM 2005 (Ingénierie Dirigée par les Modèles – Paris, France, Jun 30 - Jul. 1, 2005)
  - the Euromicro CBSE Track (31st Euromicro – Porto, Portugal, Aug. 30 - Sept. 3, 2005)
  - IWPSE '05 (International Workshop on Principles of Software Evolution – Lisbon, Portugal, Sept. 5-6, 2005)
- Stéphane Ducasse was PC Member of
  - ESUG 2005 (European Smalltalk User Group Conference – Brussels, Belgium, August, 2005)
  - LMO 2005 (Languages et Modèles à Objets – Berne, Switzerland, March 2005)
  - UML 2005 (8th International Conference on the Unified Modeling Language – Jamaica, October 2005)
  - JFDLPA (Journée Francophone sur le Développement de Logiciels Par Aspects, September 2005)
  - ECOOP 2005 (European Conference on Object-Oriented Programming – Glasgow, July 2005)
  - ICSM 2005 (International Conference on Software Maintenance – Bupadest, September 2005).
  - SDL 2005 (International Symposium on Dynamic Languages – San Diego, USA, October 2005).
  - WCRE 2005 (Working Conference on Reverse Engineering – Pittsburg, USA, November 2005).
  - NetObjectDays 2005 (Erfurt, Germany, September 2005)
- Alexandre Bergel was PC member of
  - SC 2006 (Software Composition – Vienna, Austria, 2006)
  - IWSAC 2005 (International Workshop on Software Aspects of Context – Santorini, Greece, Jul. 14, 2005)
  - the Euromicro CBSE Track (31st Euromicro – Porto, Portugal, Aug. 30 - Sept. 3, 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/oobib.cgi?snf05

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/oobib.cgi?recast05

Published papers

[AN05] Franz Achermann and Oscar Nierstrasz. A calculus for reasoning about software components. *Theoretical Computer Science*, 331(2-3):367–396, 2005.

[BD05a] Alexandre Bergel and Stéphane Ducasse. Scoped and dynamic aspects with Classboxes. *RSTI – L'Objet (programmation par aspects)*, 11(3):53–68, 2005.

[BD05b] Alexandre Bergel and Stéphane Ducasse. Supporting unanticipated changes with Traits and Classboxes. In *Proceedings of Net.ObjectDays (NODE'05)*, pages 61–75, Erfurt, Germany, September 2005.

[BDNW05] Alexandre Bergel, Stéphane Ducasse, Oscar Nierstrasz, and Roel Wuyts. Classboxes: Controlling visibility of class extensions. *Computer Languages, Systems and Structures*, 31(3-4):107–126, May 2005.

[DDT05] Marcus Denker, Stéphane Ducasse, and Éric Tanter. Runtime bytecode transformation for Smalltalk. In *Proceedings of ESUG Research Track 2005*, pages 75–98, August 2005.

[DLR04] Stéphane Ducasse, Adrian Lienhard, and Lukas Renggli. Seaside — a multiple control flow web application framework. In *Proceedings of ESUG Research Track 2004*, pages 231–257, September 2004.

[DSW05] Stéphane Ducasse, Nathanael Schärli, and Roel Wuyts. Uniform and safe metaclass composition. *Journal of Computer Languages, Systems and Structures*, 31(3-4):143–164, May 2005.

[GGN05] Markus Gälli, Orla Greevy, and Oscar Nierstrasz. Composing Unit Tests. In *Proceedings of SPLiT 2006 (2nd International Workshop on Software Product Line Testing)*, September 2005.

[GLN05] Markus Gälli, Michele Lanza, and Oscar Nierstrasz. Towards a Taxonomy of SUnit Tests. In *Proceedings of ESUG 2005 (13th International Smalltalk Conference)*, September 2005.

[GND04] Markus Gälli, Oscar Nierstrasz, and Stéphane Ducasse. One-method commands: Linking methods and their tests, October 2004. OOPSLA Workshop on Revival of Dynamic Languages.

[MBCD05] Florian Minjat, Alexandre Bergel, Pierre Cointe, and Stéphane Ducasse. Mise en symbiose des traits et des classboxes : Application à l'expression des collaborations. In *Proceedings of LMO 2005*, volume 11, pages 33–46, Bern, Switzerland, 2005.

[NBD+05] Oscar Nierstrasz, Alexandre Bergel, Marcus Denker, Stéphane Ducasse, Markus Gälli, and Roel Wuyts. On the revival of dynamic languages. In Thomas Gschwind and Uwe Aßmann, editors, *Proceedings of Software Composition 2005*, volume 3628, pages 1–13. LNCS 3628, 2005. Invited paper.

[ND04] Oscar Nierstrasz and Marcus Denker. Supporting software change in the programming language, October 2004. OOPSLA Workshop on Revival of Dynamic Languages.

[NDS05] Oscar Nierstrasz, Stéphane Ducasse, and Nathanael Schärli. Flattening Traits. Technical Report IAM-05-005, Institut für Informatik, Universität Bern, Switzerland, April 2005.

[SBD04] Nathanael Schärli, Andrew P. Black, and Stéphane Ducasse. Object-oriented encapsulation for dynamically typed languages. In *Proceedings OOPSLA '04 (International Conference on Object-Oriented Programming Systems, Languages and Applications)*, pages 130–149, October 2004.

Theses and Student projects

[Ber05] Alexandre Bergel. *Classboxes — Controlling Visibility of Class Extensions*. PhD thesis, University of Berne, November 2005.

[Lie04] Adrian Lienhard. Bootstrapping Traits. Master's thesis, University of Bern, November 2004.

[Mar05] Philippe Marschall. Detecting the methods under test in Java. Informatikprojekt, University of Bern, April 2005.

[Rot04] David Rothlisberger. The smallbb forum system. Informatikprojekt, University of Bern, October 2004.

[Sch05] Nathanael Schärli. *Traits — Composing Classes from Behavioral Building Blocks*. PhD thesis, University of Berne, February 2005.

Selected RECAST publications

[LDA05] Adrian Lienhard, Stéphane Ducasse, and Gabriela Arévalo. Identifying traits with formal concept analysis. In *Proceedings of ASE '05 (20th Conference on Automated Software Engineering)*. IEEE Computer Society Press, November 2005. To appear.

[NDD05] Oscar Nierstrasz, Stéphane Ducasse, and Serge Demeyer. Object-oriented reengineering patterns — an overview. In Michael Lowry Robert Glück, editor, *Proceedings of GPCE 2005*, pages 1–9. LNCS 3676, 2005. Invited paper.

c) Publications in press

Publications to appear

[BDN05a] Alexandre Bergel, Stéphane Ducasse, and Oscar Nierstrasz. Analyzing module diversity. *Journal of Universal Computer Science*, 2005. To appear.

[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, San Diego, CA, USA, 2005. ACM Press.

[DNS$^+$05] Stéphane Ducasse, Oscar Nierstrasz, Nathanael Schärli, Roel Wuyts, and Andrew Black. Traits: A mechanism for fine-grained reuse. *Transactions on Programming Languages and Systems*, 2005. To appear.

[NDS06] Oscar Nierstrasz, Stéphane Ducasse, and Nathanael Schärli. Flattening Traits. *Journal of Object Technology*, 5(3), May 2006. To appear.