b
UNIVERSITÄT
BERN
Philosophisch-naturwissenschaftliche Fakultät
Institut für Informatik
Software Engineering Group (SEG)

29 Dec., 2025

Review of "Smalltalk type inference improvement supporting development tooling and code analysis"

Dear Chair of the Ph.D. Committee,

I am pleased to submit my review of the dissertation entitled "Smalltalk type inference improvement supporting development tooling and code analysis", submitted by Mr. Jan Blizničenko in partial fulfillment of the requirements for the Ph.D. degree at the Faculty of Information Technology, Czech Technical University in Prague.

General Assessment

Mr. Blizničenko's dissertation builds on an extensive body of prior work on type inference techniques for dynamic programming languages, with a particular focus on Smalltalk—a highly dynamic and live programming environment for which a wide range of inference approaches has been proposed over several decades. The central contribution of Mr. Blizničenko's work lies in a novel architectural approach that integrates multiple heterogeneous type inference techniques into a configurable data pipeline. This pipeline allows different trade-offs between speed, accuracy, and completeness, thereby tailoring the inferred type information to the needs of specific developer tasks such as real-time UML visualization, static analysis, or deeper reverse-engineering activities.

Overall, the dissertation presents a mature and well-executed piece of research that combines solid conceptual design with careful empirical evaluation and practical tool support.

Up-to-dateness of the Dissertation

The dissertation demonstrates a high degree of awareness of the state of the art. It surveys a wide spectrum of type inference techniques developed for dynamic languages, extending well beyond Smalltalk itself. The survey encompasses classical static analyses, heuristic approaches (including name-based heuristics), as well as run-time techniques based on dynamic observation and test execution.

To the best of my knowledge, the dissertation adequately covers the most recent and relevant research results in the area of type inference for dynamic languages. In particular, the review of techniques specifically developed for Smalltalk is exceptionally thorough and appears both comprehensive and well balanced.

Formal Structure and Organization

The dissertation follows a classical and appropriate structure. It opens with a clear and well-written introduction that succinctly summarizes the goals, motivations, and key contributions of the work. This is followed by a detailed background and state-of-the-art chapter that not only situates the research within existing literature, but also clearly motivates the research questions and objectives pursued by the candidate.

Subsequent chapters describe the methodology and individual contributions in a logical and progressive manner, culminating in a concluding chapter that offers a thoughtful assessment of the achieved results. The overall organization supports readability and reflects a solid understanding of how to structure a substantial research contribution.

Prof. Dr. em. O. Nierstrasz
oscar@nierstrasz.org
https://www.oscar.nierstrasz.org
University of Bern
Neubrückstrasse 10
CH-3012 Bern, Switzerland
Tel. +41 78 628.0817

Completion of the Dissertation Objectives

The dissertation explicitly states three primary objectives:

1. To realize a *unifying framework* for integrating heterogeneous sources of type information in a configurable way.
2. To build an *empirical name-to-type pipeline and dataset* for Pharo Smalltalk based on lightweight heuristics.
3. To *demonstrate* the proposed framework through IDE tooling.

All three objectives have been fully realized and convincingly demonstrated.

The main contribution, the *TypeInfoTools* framework, provides a unifying domain model for representing and combining type information originating from diverse inference sources. The framework supports both *chaining* (where the first acceptable result is used) and *fusion* (where results from multiple sources are merged), enabling different trade-offs depending on requirements such as responsiveness or precision. The framework is fully implemented and available within the Pharo Smalltalk ecosystem.

Importantly, this contribution should not be viewed merely as a software artifact. Its true value lies in the conceptual design of a general framework for integrating heterogeneous typing sources. This design is, in principle, transferable to other dynamic languages beyond Smalltalk.

The second objective generalizes prior work on name-based typing heuristics by extending it to support application-specific naming conventions. This contribution is also realized as a concrete software artifact and empirically evaluated across multiple software systems, yielding convincing results.

Finally, several prototype demonstrators illustrate how the framework can be used to build IDE tools that exploit the inferred type information. These demonstrators effectively ground the conceptual contributions in realistic development scenarios.

Assessment of the Methods Used

The dissertation primarily employs two complementary research methodologies:

1. *Design Science*, used to construct tools that are grounded in existing scientific knowledge while enabling new forms of investigation.
2. *Empirical Evaluation*, used to assess proposed techniques using metrics such as coverage, recall, and precision.

Both methodologies are applied consistently and appropriately. The author is also commendably transparent about the inherent difficulty of empirical evaluation in the absence of a well-defined ground truth. In dynamically typed languages, the “true” types of variables, expressions, and methods are often ambiguous or context-dependent, making definitive accuracy assessments inherently problematic. This limitation reflects the nature of the problem domain rather than a weakness of the research itself.

Evaluation of the Results and Contributions

The *TypeInfoTools* framework clearly constitutes the principal contribution of the dissertation. It provides a flexible and well-structured architecture for integrating multiple typing sources into a configurable pipeline. Depending on the use case, the pipeline can prioritize fast but coarse-grained techniques or combine several more expensive techniques to improve accuracy.

The framework's design and architecture are convincing, and the empirical evaluation—although necessarily limited by the lack of ground truth—is carefully executed using multiple case studies.

The *Name-to-Type* inference technique represents a strong secondary contribution. By generalizing earlier work and incorporating more fine-grained heuristics, it delivers useful type hints in practice. As with the main framework, the enduring value lies in the conceptual design rather than solely in the validating software artifact. The accompanying empirical evaluation is again convincing and well-motivated.

The final contribution, consisting of IDE-based demonstrators, should be understood not as an independent research result but as a practical illustration of the relevance and applicability of the first two contributions. In this role, it serves as a useful and appropriate complement to the thesis.

Parts of the research have undergone peer review and have been published in international workshops (notably IWST) as well as in an archival journal venue, the *Journal of Computer Languages*, which further attests to the quality of the work.

Remarks, Objections, and Questions for the Defense

The dissertation is generally clearly written, but I would like to raise several questions that could be addressed during the Ph.D. defense:

- The term *proper type* is used in several places (*e.g.*, Chapter 2, p. 9) but is never formally defined. The cited reference [67] also does not clarify this notion. What exactly is meant by a "proper type" in this context?
- The dissertation claims (p. 16) that cross-project aggregation of observed run-time types keyed by identifier or selector names is not reported in the literature. Why should such aggregation across projects yield significantly more reusable data, and is there empirical evidence to support this claim?
- The "type tree" is modeled as a forest (Chapter 4, p. 39). Why was this choice made instead of modeling it as a single tree with 'nil' as a global root?
- How are *weights* (p. 39) defined and determined in practice?
- When ordering sources for chaining (p. 41), how does one determine which sources are "fast"? Is this based on empirical measurement, historical data, or user intuition?
- During fusion (p. 41), can false positives produced by earlier sources be corrected by later ones?
- In some cases, 'Object' or 'UndefinedObject' may be valid inferred types (p. 41). How does the framework distinguish these cases from uninformative results?
- What is a *base weight* (p. 42), how does it differ from a weight, and how are weights combined during fusion?
- Were any experiments conducted against a manually established ground truth (p. 51)? If not, would it be feasible and useful to construct such a dataset with expert annotation?
- How is *coverage* defined for methods? Does it refer to statements, AST nodes, execution paths, or another unit?
- Why is the sender class recorded (Chapter 5, p. 69) rather than the sender method? What precisely is meant by "sender class"? Is it that of the sender instance, or of the class implementing the sender method?
- How expensive is the recursion check (p. 71) in terms of reification and stack inspection?
- I could not find test cases in the public GitHub repository of the project. Is there a reason?

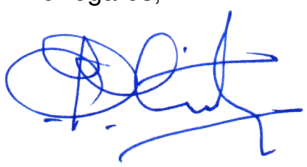
Overall Evaluation

This dissertation offers a distinctive and valuable contribution to the field of practical type inference for dynamic languages. By proposing a non-trivial and well-reasoned architecture for integrating heterogeneous typing sources, it advances both the scientific understanding and the engineering practice of type inference. The work is clearly written, methodologically sound, and convincingly evaluated within the inherent limitations of the domain.

I am confident that this research will have a lasting impact. I therefore *strongly recommend* this dissertation for acceptance and for admission of the candidate to the Ph.D. defense.

The author of the dissertation proved the ability to conduct research and achieve scientific results. In accordance with par. 47, letter (4) of the Law Nr. 111/1998 (The Higher Education Act) I do recommend the thesis for the presentation and defense with the aim of receiving the Ph.D. degree.

Kind regards,



Prof. Dr. em. O. Nierstrasz