Artifact: Cassandra Source Code, Feature Descriptions across 27 versions, with Starting and Ending Version Trace Matrices

Mona Rahimi and Jane Cleland-Huang
Department of Computer Science and Engineering, University of Notre Dame
South Bend IN, USA
m.rahimi@acm.org, JaneClelandHuang@nd.edu

Artifact Description

Name of Artifact:
Evolution of Trace Links across 27 versions of Features to Code in Cassandra.

Artifact License:
Cassandra Source code is licensed under Apache Software License v2.0. We release our artifacts under the same license.

Download Site:
If our artifact is accepted we will release it prior to publication on the PROMISE repository, thereby providing stable, long-term access. We have released it via Dropbox (http://tinyurl.com/TLEArtifacts) for review purposes, as we would like to perform a final review before its permanent public release.

Brief Description of Artifact:
Our data set includes 27 versions of the Cassandra Distributed Database System ranging from Version 1.0.0-beta1,Sep 2011 (V_{start}) and ending at Version 1.2.1,Jan 2013 (V_{end}). All versions include *source code* downloaded directly from the Cassandra project in its and transformed into SrcML as well as *Feature descriptions* manually extracted from Cassandra's documentation and release notes. V_{start} and V_{end} also include a trace matrix showing trace links from feature requests to source code classes.

Scorecard

Insightful
Software traceability is an essential element in developing and certifying safety-critical systems. It also supports numerous software engineering activities such as impact analysis, compliance verification, and coverage analysis. Given the cost and effort involved in creating and maintaining trace links, researchers have focused significant effort on developing automated trace link creation solutions. However, while community effort has resulted in approximately 20 publicly available artifact sets for evaluating trace link creation, there are no existing publicly available data sets for evaluating trace link evolution. Such data sets require multiple, sequential versions of *source* artifacts (e.g. requirements), *target* artifacts (e.g. source code), and trace matrices to provide an *answer set* of trace links between source and target artifacts. Our artifacts provide researchers with data needed to evaluate trace link evolution algorithms and tackle the previously under-addressed problem of trace link maintenance.

Useful
Our multi-version artifact set is a first step in filling the need for data sets that provide multiple versions of requirements, source code, and trace matrices. It has limitations because we only provide two trace matrices for V_{start} and V_{end} and not for each of the 25 intermediate versions. Each matrix includes trace links between the 48 feature descriptions associated with V_{start} and the complete set of source code for V_{start} and V_{end} respectively. Researchers can therefore evolve trace links across all versions, but must perform the evaluation against final version V_{end}. Nonetheless, we believe this first-of-its kind publicly available artifact set will be useful for the research community and will facilitate renewed interest in research related to trace link maintenance and evolution. Unlike many other types of data sets which can be mined directly from open-source projects, the construction of our data set required very significant human effort. While we followed a systematic and rigorous process, we cannot guarantee correctness and completeness of our trace matrices; however, exposing it to public scrutiny will allow future refinements to occur.

Usable
Data is stored in a standard format that is easily parsable. Our artifact includes the following:

- **A detailed explanation** of the artifacts provided in the form of a *readme* file.
- **Source Code:** Raw source code as well as several processed formats including: a call graph, a list of classes, a list of methods, and inheritance relationships are provided for each version.
- **Feature descriptions:** A list of feature descriptions for each version.
- **Delta between pairs of sequential versions:** Lists of modified, added, and deleted methods.
- **Trace Matrices:** Trace matrices between features and class files for V_{start} and V_{end}.

The standard format of the data and the instructions we provide, make our artifacts eminently usable by other researchers.

Artifact: Cassandra Source Code, Feature Descriptions across 27 versions, with Starting and Ending Version Trace Matrices

Mona Rahimi and Jane Cleland-Huang
Department of Computer Science and Engineering, University of Notre Dame
South Bend IN, USA
m.rahimi@acm.org, JaneClelandHuang@nd.edu

***Abstract*—To facilitate research into trace link evolution we present 27 versions of Cassandra source code, feature descriptions for each version, deltas between versions, structured descriptions of each version, and trace links between a subset of 48 features and source code for the starting and ending versions.**

I. Motivation

Software traceability supports a variety of software engineering activities such as safety analysis, impact analysis, regression testing, and compliance verification [2]. Researchers have invested significant effort developing automated techniques for trace link creation [1], [4], while little attention has been paid to trace link evolution [5], [7]. The lack of data sets for evaluating link evolution algorithms is a major inhibiting factor. Data sets for evaluating trace link creation require *source* artifacts (e.g. requirements), *target* artifacts (e.g. source code), and a *trace matrix*, and there are currently over 20 data sets in the public domain [3]. In contrast, trace link evolution data-sets require *multiple, sequential versions* of source artifacts, target artifacts, and trace matrices, and to the best of our knowledge there are no such data sets currently available in the public domain.

II. Cassandra-27 Version Features-to-Code

To facilitate research into trace link evolution [7] we mined and constructed a data set from the Cassandra Open Source system. The data set includes the following:

- **Source code** for 27 versions of Cassandra starting from Version 1.0.0-beta1,Sep 2011 (V_{start}) and ending at Version 1.2.1,Jan 2013 (V_{end}). V_{start} had 488 classes and 86,852 Lines of Code (LOC) while V_{final} had 702 classes and 132,111 LOC. Our artifact also includes srcML representations [6], lists of classes and methods, call-graphs, and hierarchical associations for each version.
- **Feature descriptions** for each of the 27 versions were manually extracted from Cassandra's initial- and release-documentation. We identified 48 features for V_{start} growing to 594 in V_{final}. Feature descriptions are also provided for all intermediate versions. An example of a V_{start} feature is "An authenticator handles the authentication challenge/response cycles of a single connection."
- **Deltas between sequential versions** showing added, deleted, and modified classes.
- **Trace links** from the original 48 features to source code are provided at the feature-to-file level in two trace matrices for V_{start} and V_{end} respectively. Approximately 93% of V_{start} links were identified directly through references to classes found in the documentation, while the remaining 7% were found through evaluating candidate links automatically generated using the Vector Space Model [4].

Our data set is available under Apache Software License v2.0 from http://tinyurl.com/TLEArtifacts. We will upload to the PROMISE repository for public release.

III. Research Challenge

The facilitated research challenge is: "Given an initial version V_{start} of feature descriptions, source code, and trace links, and also a subsequent version V_{end} of feature descriptions and source code; can trace links be evolved to reflect the feature-to-code associations in V_{end}?

IV. Acknowledgments

The work of constructing this artifact was primarily funded by US National Science Foundation grant CCF-1647342.

References

[1] G. Antoniol, G. Canfora, G. Casazza, A. De Lucia, and E. Merlo. Recovering traceability links between code and documentation. *IEEE Transactions on Software Engineering*, 28(10):970–983, 2002.

[2] J. Cleland-Huang, O. Gotel, J. H. Hayes, P. Mäder, and A. Zisman. Software traceability: trends and future directions. In *Future of Software Engineering, FOSE 2014,*, pages 55–69, 2014.

[3] J. Guo, M. Rahimi, J. Cleland-Huang, A. Rasin, J. H. Hayes, and M. Vierhauser. Cold-start software analytics. In *Intn'l Conf. on Mining Software Repositories, MSR 2016, Austin, TX, USA, May 14-22, 2016*, pages 142–153, 2016.

[4] J. Huffman Hayes, A. Dekhtyar, and S. K. Sundaram. Advancing candidate link generation for requirements tracing: The study of methods. *IEEE Transactions on Software Engineering*, 32(1):4–19, 2006.

[5] P. Mäder and O. Gotel. Towards automated traceability maintenance. *Journal of Systems and Software*, 85(10):2205–2227, 2012.

[6] J. I. Maletic and M. L. Collard. Exploration, analysis, and manipulation of source code using srcml. In *Intn'l Conf. on Software Engineering, ICSE 2015, Vol. 2*, pages 951–952, 2015.

[7] M. Rahimi, W. Goss, and J. Cleland-Huang. In *International Conference on Software Maintenance and Evolution (ICSME), Title = Evolving Requirements-to-Code Trace Links across Versions of a Software System, Year = 2016.*