Capturing Custom Link Semantics among Heterogeneous Artifacts and Tools

Hazeline U. Asuncion Richard N. Taylor
Institute for Software Research
University of California, Irvine
Irvine, California 92697-3455 USA
{hasuncion, taylor}@ics.uci.edu

Abstract

Automated techniques aid in minimizing the overhead associated with the capture and maintenance of trace links. However, many challenges to automated traceability remain, such as linking heterogeneous artifacts and capturing custom link semantics. In this position paper, we propose a combination of techniques, including prospective link capture, open hypermedia, and rules, in order to address these challenges and complement current automated techniques. Our approach borrows ideas from e-Science, a domain in which tracing data plays a crucial role in the repeatability of experiments.

1. Introduction

Researchers and practitioners agree that software traceability can improve software development by increasing accessibility to related artifacts [13, 15]. The high overhead in establishing and maintaining trace links [32] has prompted research in the automated recovery of trace links [27, 28, 36]. However, these techniques fall short of linking to heterogeneous artifacts and capturing custom link semantics, such as the trace relationship type or purpose of the link [20].

To address these difficulties, we propose a novel combination of automated techniques for capturing trace links, involving the prospective capture of links along with concepts from open hypermedia and rules. The prevalent approach to automatically capturing trace links is to perform *retrospective* techniques that recover trace links from existing artifacts. In contrast, *prospective* trace link generation captures links *in situ*, while artifacts are generated or modified, to enable the capture of temporal and contextual relationships that are not captured through other trace recovery techniques. Open hypermedia concepts like first class n-ary links and hypermedia adapters aid in modeling captured link semantics and in enabling traceability across tool boundaries. Finally, externally pluggable rules enable the automatic addition and maintenance of custom trace semantics.

Our approach borrows ideas from the domain of e-Science, an area where traceability, referred to as data provenance, plays a key role in the repeatability of experiments. While traceability within this domain is more constrained than traceability between artifacts in software, we are able to glean insights that are generally applicable to software traceability.

We center our discussion on the mechanisms for automatically capturing link semantics across artifacts that are of varying levels of formality, abstraction, and representation and that are generated by different tools. While we do not concentrate on defining an ontology for link semantics, we do focus on supporting *user-defined* link semantics, similar to those in [20, 37].

2. The e-Science perspective

A domain that shares many similarities with software engineering, e-Science is a burgeoning field where scientific experiments are conducted on computers. *In silico* experiments, performed on the computer or via computer simulation [10], enable data analysis on existing data as well as the formulation of hypotheses to be tested in the laboratory [30]. The field is characterized by distributed global collaborations among scientists using large-scale data sets and heterogeneous computational resources [34].

Like software development, an *in silico* experiment also has a lifecycle. Interestingly, the phases in an *in silico* lifecycle may be compared to the phases in a software development lifecycle. Experiment design, running, and publication are analogous to software design and implementation, testing, and deployment. (see Figure 1).

Data provenance techniques in e-Science enable tracing data products across an entire experiment life cycle to support the repeatability of experiments. The

TEFSE'09, May 18, 2009, Vancouver, Canada
978-1-4244-3741-2/09/$25.00 © 2009 IEEE

main approach of capturing data provenance is through scientific workflows [38] since these not only capture the individual workflow execution, but they also codify the design of the experiment or scientific analysis. While many data provenance challenges remain, provenance systems have been used to further scientific research [2, 14, 16, 39, 42]. Given this measured success, insights from data provenance techniques can potentially improve software traceability.

The following insights guide our approach.

2.1 Automated capture of data provenance involves *in situ* recording of data manipulation

In e-Science, automated data provenance collection takes place while data sets are being processed by various transformation functions, which are usually directed by a scientific workflow. Examples include recording user interaction with data [7], recording component interaction in an executing workflow [1], recording service invocations [41], and recording workflow execution [42]. Lower level recording is also used, as in recording interactions between service objects [18] or recording operating system level events [8]. These events are stored in log files which are automatically analyzed at a later stage.

2.2. There is a tradeoff between tool "openness" and automatic semantic capture

"Openness" is the ability to capture provenance among heterogeneous tools [18]. A higher level of openness allows for the increased ability to capture provenance across heterogeneous tools. However, a higher level of openness also lowers the level of semantics that can be automatically captured. For example, CAVES [7] is not capable of capturing provenance with external tools, but the provenance captured has a high level of semantics since it is directly related to its internal data analysis tool. On the other end of the spectrum, PASS [8] has a high level of openness (i.e. it can capture provenance across any tool via operating system events), but the captured semantics is nil.

2.3. Reasoners aid in automatically inferring relationships between scientific artifacts

Since the automated collection of data provenance can potentially produce large amounts of data, some provenance systems support reasoning about the collected provenance information, i.e. the ability to analyze, query, and browse captured provenance [31]. Reasoners determine the type of relationship between artifacts using the *context*, which can be broadly defined as "anything that was true" during the experiment [29]. The two kinds of contextual information in e-Science are the provenance information captured during an experiment and the external information surrounding an experiment. The first kind aids in understanding how data sets are manipulated by computational objects. The types of relationships between scientific objects may be classified as dependency, temporal, or contributor relationships [18, 38]. The context in an experiment frames assumptions about the environment (e.g. hardware environment, loaded libraries) and how entities are related (e.g. "input data X and intermediate data Y are fed into computation object A to produce output data Z"). Meanwhile, the contextual information surrounding an experiment links the objects in an experiment to their real world representation. This information may be categorized as organizational (e.g. user name, hypothesis) and knowledge-based (e.g. notes) [38]. The context may also include the rationale behind the experiment [1].

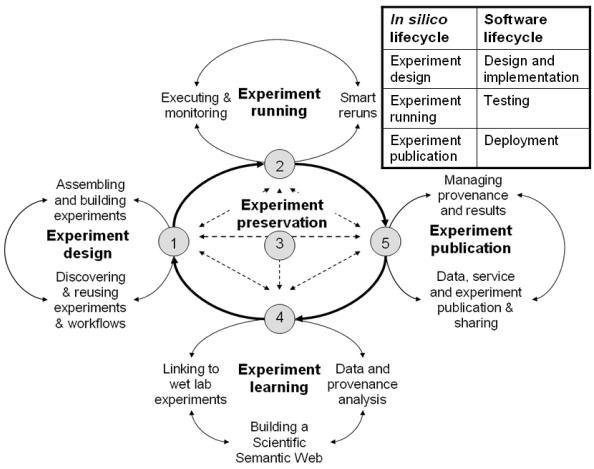

Figure 1: *In silico* experiment lifecycle [38]

3. Applying insights to traceability

The preceding insights provide the basis for our approach of using prospective link capture with concepts from open hypermedia and rules. Prospective link capture enables the capture of links across artifacts with varying levels of representation. Open hypermedia enables the capture of links across tool boundaries. Rules enable the capture of custom link semantics. The details of our approach follow.

3.1 Prospective link capture

Automated capture of data provenance involves *in situ* recording of data manipulation. Likewise, we pro-

pose the prospective capture of trace links. The prospective approach creates links as a side-effect to stakeholders' development tasks. Trace links can be automatically added by analyzing user input events such as keyboard and mouse events. Links can be created between artifacts that are simultaneously or sequentially accessed. In contrast to existing retrospective techniques, the prospective approach enables the capture of contextual information (e.g. causal relationships) and temporal relationships between artifacts that are not captured otherwise. We hypothesize that 1) related artifacts are generated, edited, or accessed either concurrently or sequentially of each other and 2) the prospective approach yields usable links. Interestingly, other research areas take a similar stance, stating that the navigation path of a user among the artifacts indicate how the artifacts are related to each other [35, 40]. Unlike information retrieval techniques that rely on mining repository logs, this approach enables the capture of links between artifacts regardless of whether the related artifacts are modified or not. Since the prospective capture of links is not dependent on textual similarities between artifacts, this approach enables links to be captured across heterogeneous artifacts (e.g. between a design document and a video recording of the deliberation over the design). Prospective link capture can be used in conjunction with rules to filter noise and to automatically add semantic information. Note that the prospective approach can be coupled with a retrospective approach to create high quality links.

3.2 Open hypermedia

Concepts from open hypermedia can be used to increase the "openness" of tools, facilitating the capture and management of trace links between heterogeneous artifacts. In e-Science, we observed that there is a tradeoff between the level of openness and the level of semantic capture. In our approach, we will try to overcome this tradeoff by infusing "knowledge" in the form of rules. Rules assign link semantics whenever a pattern of user interaction with artifacts is detected (see Section 3.3).

We use first class links with n-ary endpoints to represent semantically rich links; thus, links may contain link metadata, endpoint metadata, and mechanisms for managing link endpoints. The link metadata encodes the trace relationship and the endpoints. The endpoint metadata may include the native artifact editor and timestamp of the most recent traversal. Mechanisms for managing link endpoints include the use of a specialized search engine such as [5] to locate the relevant portions within an artifact or the use of a notification wrapper to send modification events to related artifacts (see Figure 2). Notification adapters can be used to automatically update link metadata (e.g. update endpoint status to obsolete when all bug reports related to a component are closed).

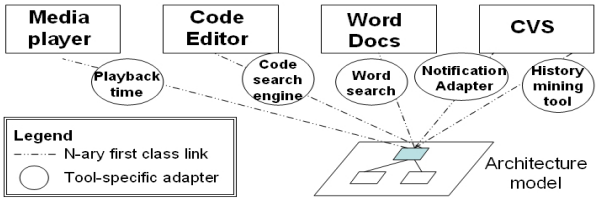

Figure 2: Open hypermedia

We also leverage existing tool capabilities by integrating them into an open hypermedia system; hence we are currently building a representative set of tool-specific adapters that either leverage the native hypertext capabilities already present in some tools (e.g. Adobe Acrobat, MS Word) or may be custom built through plug-ins, macro programs, or other extension mechanisms (e.g. Eclipse). We believe that building these adapters will provide an effective means for tracing across heterogeneous tool boundaries.

3.3 Rules

Reasoners in data provenance systems use contextual information to infer relationships between pieces of data. Similarly, we propose using rules to infer relationships between artifacts. Since the context can uncover the assumptions made when artifacts are generated or modified, it is important to take advantage of the different types of context: organization-specific, project-specific, and role-specific (e.g. "Architects create use cases during requirements analysis"). Rules specified by users can encode this contextual information, and these rules can be used to 1) guide the prospective capture of links and 2) automatically identify relationship types between artifacts. For instance, a rule may check if a component is selected in a design diagram and if the user enters information into a Wiki during the "record" mode. If these conditions are met, then a trace link between the component and the Wiki page can be captured, with the relationship type "rationale". Since users can specify custom rules, we hypothesize that this approach increases the relevancy of captured links to the user and/or development tasks. Because rules can guide the prospective capture of links, less irrelevant links will be captured.

4. Related work

Automatically capturing link semantics between different artifacts has been tackled by the areas of natu-

ral language processing (NLP) and information retrieval (IR). For instance, Basili et al. use co-occurrences of concepts in documents to generate typed hyperlinks [6]; Spanoudakis et al. use rules to detect patterns of terms to create links between requirements specifications, use cases, and object model, all of which have text representations [36]; Jiang et al., Lucia et al., and Marcus and Maletic use latent semantic indexing to identify links between documents [25, 27, 28]; Camacho-Guerrero et al. use NLP techniques with latent semantic indexing to automatically create semantic hyperlinks [9]. These approaches are geared toward the recovery of link semantics based on the *textual* content of the artifacts. We aim to trace artifacts that are not only represented as text, by using the user's interaction with artifacts as the basis for creating links. Our approach can also be nicely integrated with these text-based techniques.

Capturing links across different artifacts and tools has also been tackled by various research areas. One class of program comprehension (PC) tools, recommender systems [11, 22, 35], uses software repository mining to automatically identify possible links within code and between code and other artifacts. For example, Hipikat [11], uses various sources (e.g. email archives, software repositories, activity logs) to create links between code and other artifacts. Meanwhile, Jazz, a collaborative software development tool, creates links between the current work context and files from collaboration tools (e.g. source code with chat) [23]. Software Concordance, an open hypermedia tool, links source code and documentation using a uniform object model [19]. Finally, Infinité, enables linking across various artifacts by translating the artifacts into a homogeneous representation and creating links within this environment [3]. While these approaches use different linking heuristics, none allows users to specify their own custom linking heuristics.

Recording user interaction with artifacts has also been studied in different research areas. Computer human interaction and computer-supported cooperative work employ user interaction to raise awareness [21, 33, 40]. Recently, the PC community has studied the capture of user interaction to aid in program comprehension: using a team's interaction with the code to create links between source code files [12], using a developer's navigation patterns between an IDE and a browser to create links between code and documentation [17], and using navigation patterns in the code to create links between tasks and source code [26, 43]. The recording of user interaction in these different research areas suggests that this is a viable approach to capturing links between artifacts. While PC techniques have focused on user interaction with the artifacts represented as text, namely source code, we posit that links can also be created across heterogeneously represented artifacts. We can use the PC linking heuristics as a starting point for creating links between heterogeneously represented artifacts.

5. Challenges and research directions

We have implemented a traceability tool based on these ideas, within ArchStudio4 [4, 24]. While initial results are promising, we are investigating whether the prospective approach can create links with an acceptable level of accuracy. The standard metrics for evaluating automated traceability systems are precision and recall [28], and they have mainly been applied to retrospectively captured links. One potential difficulty with these metrics is that they rely on labels that specify whether two artifacts should be related. However, the relevancy between two artifacts may differ with different users. In our approach, users are allowed to specify custom rules, which can potentially boost precision. Furthermore, the user's sequence of interactions with the system could affect the recall. We conjecture that combining prospective capture with retrospective capture may increase both precision and recall rates. Another important research question is "Where is the threshold by which captured links are usable and can support development tasks for users?" In addition, it is important to use link accuracy metrics that also measure the correctness of assigned link semantics.

Since overhead in capturing trace links plays a major role in the lack of adoption of a traceability technique [32], we are studying whether the overhead incurred with our approach is within an acceptable threshold for users. We are interested in studying two specific types of overhead: overhead associated with extending and customizing the tool (e.g. creating tool adapters, creating custom rules) and overhead associated with using the tool (e.g. applying rules, filtering noise). One approach to reducing overhead is to integrate our prospective capture technique with retrospective techniques and assign higher link quality to links captured using multiple techniques, in order minimize the manual post-analysis of links.

6. Acknowledgements

Effort funded by the US Nat'l Science Foundation grants IIS-0205724, CNS-0438996, & CCF-0820222.

7. References

[1] Altintas, I., Barney, O., et al. Provenance Collection Support in the Kepler Scientific Workflow System. In *Int'l Provenance & Annotation Workshop (IPAW)*. Chicago, 2006.

[2] Altintas, I., Barney, O., et al. Accelerating the Scientific Exploration Process with Scientific Workflows. *Journal of Physics: Conf Series.* 46(1), p. 468-78, 2006.
[3] Anderson, K.M., Sherba, S.A., et al. Towards Large-Scale Information Integration. In *ICSE.* Orlando, FL, 2002.
[4] Asuncion, H. Towards Practical Software Traceability. In *ICSE Doctoral Symp.* Leipzig, Germany, 2008.
[5] Bajracharya, S., Ngo, T., et al. Sourcerer: A Search Engine for Open Source Code Supporting Structure-Based Search In *OOPSLA '06: Companion to the 21st ACM SIGPLAN Conf on Object-Oriented Programming Systems, Languages, and Apps.* 681-682, 2006.
[6] Basili, R., Pazienza, M.T., et al. Inducing Hyperlinking Rules in Text Collections. In *Recent Advances in Natural Language Processing.* p. 131-140, Borovets, Bulgaria, 2003.
[7] Bourilkov, D. THE CAVES Project. *Int'l Journal of Modern Physics.* 20, p. 3889-3892, 2005.
[8] Braun, U., Garfinkel, S., et al. Issues in Automatic Provenance Collection. In *IPAW.* Chicago, IL, 2006.
[9] Camacho-Guerrero, J.A., Carvalho, A.A., et al. Clustering as an Approach to Support the Automatic Definition of Semantic Hyperlinks. In *18th Conf on Hypertext and Hypermedia.* UK, 2007.
[10] Cavalcanti, M.C., Targino, R., et al. Managing Structural Genomic Workflows Using Web Services. *Data & Knowledge Eng.* 53(1), p. 45-74, 2005.
[11] Cubranic, D., Murphy, G.C., et al. Hipikat: a Project Memory for Software Developmt. *TSE.* 31(6) p446-65, 2005.
[12] DeLine, R., Czerwinski, M., et al. Easing Program Comprehension by Sharing Navigation Data. In *2005 IEEE Symp on Visual Languages and Human-Centric Comp.* 2005.
[13] Domges, R. and Pohl, K. Adapting Traceability Environments to Project Specific Needs. *CACM.* 41(12), p. 54-62, 1998.
[14] Ellison, A.M., Osterweil, L.J., et al. Analytic Webs Support the Synthesis of Ecological Data Sets. *Ecology.* 87(6), p. 1345-1358, June, 2006.
[15] Evans, M. SPMN Director Identifies 16 Critical Software Practices *CrossTalk, Journal of Defense S. Eng.* 2001.
[16] Freire, J., Silva, C.T., et al. Managing Rapidly-Evolving Scientific Workflows. In *IPAW.* Chicago, IL, 2006.
[17] Goldman, M. and Miller, R.C. Codetrail: Connecting Source Code and Web Resources. In *Visual Languages and Human-Centric Computing.* 2008.
[18] Groth, P., Miles, S., et al. Recording and Using Provenance in a Protein Compressibility Experiment. In *14th IEEE Int'l Symp on High Performance Distributed Computing.* NC, 2005.
[19] Gupta, S.C., Nguyen, T.N., et al. The Software Concordance: Using a Uniform Document Model to Integrate Program Analysis and Hypermedia. In *10th Asia-Pacific Software Eng Conf.* 2003.
[20] Hayes, J. and Dekhtyar, A. *Grand Challenges for Traceability.* Center of Excellence for Traceability, Tech Report COET-GCT-06-01-0.9, 2007.
[21] Hill, W.C., Hollan, J.D., et al. Edit Wear and Read Wear. In *SIGCHI Conf on Human Factors in Computing Systems.* CA, 1992.
[22] Holmes, R. and Murphy, G.C. Using Structural Context to Recommend Source Code Examp. In *ICSE. St Louis* 2005.
[23] Hupfer, S., Cheng, L.-T., et al. Introducing Collaboration into an Application Development Environment. In *CSCW.* Chicago, IL, 2004.
[24] Institute for Software Research. *ArchStudio 4*. http://www.isr.uci.edu/projects/archstudio/, UC Irvine, 2006.
[25] Jiang, H.-y., Nguyen, T.N., et al. Traceability Link Evolution Management with Incremental Latent Semantic Indexing. In *Proc. of the 31st Annual International Computer Software and Applications Conference.* 2007.
[26] Kersten, M. and Murphy, G.C. Mylar: A Degree-of-Interest Model for IDEs. In *4th Int'l Conf on Aspect-oriented Software Development.* p. 159-168, Chicago, IL, 2005.
[27] Lucia, A.D., Oliveto, R., et al. Adams Re-trace: Traceability Link Recovery via Latent Semantic Indexing. In *ICSE.* Leipzig, Germany, 2008.
[28] Marcus, A. and Maletic, J.I. Recovering Documentation-To-Source-Code Traceability Links Using Latent Semantic Indexing. In *ICSE.* Portland, OR, 2003.
[29] Miles, S., Groth, P., et al. *The Requirements of Recording and Using Provenance in e-Science Experiments.* Elec. & Comp Science, Univ of Southampton, T. Rep, 2005.
[30] Oinn, T., Greenwood, M., et al. Taverna: Lessons in Creating a Workflow Environment for the Life Sciences. *Concurrency and Computation: Practice and Experience.* 18(10), p. 1067-1100, 2006.
[31] Rajbhandari, S. and Walker, D.W. Support for Provenance in a Service-based Computing Grid. In *e-Science All-Hands Meeting* Nottingham, UK, 2004.
[32] Ramesh, B., Powers, T., et al. Implementing Requirements Traceability: A Case Study. In *RE '95.* p. 89-95, York, UK, 1995.
[33] Schummer, T. Lost and Found in Software Space. In *34th Annual Hawaii Int'l Conf on System Sciences.* 2001.
[34] Senior, I. *e-Science Definitions*. http://e-science.ox.ac.uk/ public/general/definitions.xml, 2002.
[35] Singer, J., Elves, R., et al. NavTracks: Supporting Navigation in Software Maintenance. In *21st Int'l Conf on Software Maintenance.* 2005.
[36] Spanoudakis, G., Zisman, A., et al. Rule-based Generation of Requirements Traceability Relations. *Journal of Systems and Software.* 72(2), p. 105-27, 2004.
[37] Spanoudakis, G. and Zisman, A. *Software Traceability: A Roadmap.* Advances in Software Eng and Knowledge Eng. Chang, S.K. ed. 3, World Scientific Publishing, 2005.
[38] Stevens, R., Zhao, J., et al. Using Provenance to Manage Knowledge of In Silico Experiments. *Briefings in Bioinformatics.* 8(3), p. 183-94, 2007.
[39] Stevens, R.D., Tipney, H.J., et al. Exploring Williams–Beuren Syndrome Using myGrid. *Bioinformatics.* 20(1), pi303-10.
[40] Wexelblat, A. and Maes, P. Footprints: History-rich Tools for Information Foraging. In *SIGCHI.*Pittsburgh, 1999.
[41] Zhao, J., Goble, C., et al. Semantically Linking and Browsing Provenance Logs for E-science. In *1st Int'l Federation for Info. Processing Conf.* Paris, France, 2004.
[42] Zhao, Y., Wilde, M., et al. Applying the Virtual Data Provenance Model. In *IPAW.* Chicago, IL, 2006.
[43] Zou, L., Godfrey, M.W., et al. Detecting Interaction Coupling from Task Interaction Histories. In *15th IEEE Int'l Conf on Program Comprehension.* 2007.