Multi-Modal Modeling, Analysis, and Validation of Open Source Software Development Processes*

Walt Scacchi[1], Chris Jensen[1], John Noll[1,2], and Margaret Elliott[1]
[1]*Institute for Software Research*
University of California, Irvine
Irvine, CA, USA 92697-3455
[2]*Santa Clara University*
Santa Clara, CA
Wscacchi@uci.edu

Appears in *J. Internet Technology and Web Engineering*, 1(3), 49-63, 2006.
Previous version appeared in *Proc. First Intern. Conf. Open Source Software*, 1-8, Genova, Italy, July 2005.
Received *Best Paper* Award. Also reprinted in G. Alkhatib and D. Rine, (Eds.), *Integrated Approaches in Information Technology and Web Engineering: Advancing Organizational Knowledge Sharing*, Information Science Reference, Hershey, PA, 51-65, 2009.

Abstract

Understanding the context, structure, activities, and content of software development processes found in practice has been and remains a challenging problem. In the world of free/open source software development, discovering and understanding what processes are used in particular projects is important in determining how they are similar to or different from those advocated by the software engineering community. Prior studies have revealed that development processes in F/OSSD projects are different in a number of ways. In this paper, we describe how a variety of modeling perspectives and techniques are used to elicit, analyze, and validate software development processes found in F/OSSD projects, with examples drawn from studies of the software requirements process found in the NetBeans.org *project.*

Keywords: Software process, process modeling, open source software development, requirements processes, empirical studies of software engineering

1. Introduction

In the world of globally dispersed, free/open source software development (F/OSSD), discovering and understanding what processes are used in particular projects is important in determining how they are similar to or different from those advocated by the software engineering community. For example, in our studies of software requirements engineering processes in F/OSSD projects across domains like Internet infrastructure, astrophysics, networked computer games, and software design systems [25,26,27], we generally find there are no explicit software requirements specifications or documents. However, we readily find numerous examples of sustained, successful, and apparently high-quality F/OSS systems being deployed on a world-wide basis. Thus, the process of software requirements engineering in F/OSSD projects must be different that the standard model of requirements elicitation, specification, modeling, analysis, communication, and management [22]. But if the process is different, how is it different, or more directly, how can we best observe and discover the context, structure, activities, and content software requirements processes in F/OSSD projects? This is the question addressed here.

Our approach to answering this question uses multi-modal modeling of the observed processes, artifacts, and other evidence composed as an ethnographic hypermedia that provides a set of informal and formal models of the software development processes we observe, codify, and document. Why? First, our research question spans two realms of activity in software engineering, namely, software development and software process modeling. So we will need to address multiple perspectives or viewpoints, yet provide a traceable basis of evidence and analysis that supports model validation. Second, given there are already thousands of self-declared F/OSSD projects affiliated with OSS portals like SourceForge.net, Freshmeat.net and

Savannah.gnu.org, then our answer will be constrained and limited in scope to the particular F/OSSD project(s) examined. Producing a more generalized model of the F/OSS development process being studied requires multiple, comparative project case studies, so our approach should be compatible with such a goal [25]. Last, we want an approach to process modeling that is open to independent analysis, validation, communication, and evolution, yet be traceable to the source data materials that serve as evidence of the discovered process in the F/OSSD projects examined [cf. 15].

Accordingly, to reveal how we use our proposed multi-model approach to model requirements processes in F/OSSD projects, we first review related research to provide the foundational basis for our approach. Second, we describe and provide examples of the modeling modes we use to elicit and analyze the processes under study. Last, we examine what each modeling mode is good for, and what kind of analysis and reasoning it supports.

2. Related Research and Approach

There is growing recognition that software requirements engineering can effectively incorporate multi-viewpoint [7,16,22] and ethnographic techniques [22,31] for eliciting, analyzing, and validating functional and non-functional software system *product* requirements. However, it appears that many in the software engineering community treat the *process* of requirements engineering as transparent and prescriptive, though perhaps difficult to practice successfully. However, we do not know how large distributed F/OSSD projects perform their development processes [cf. 3].

Initial studies of requirements development across multiple types of F/OSSD projects [25,26] find that OSS product requirements are continuously emerging [8,9,30] and asserted after they have been implemented, rather than relatively stable and elicited before being implemented. Similarly, these findings reveal requirements practice centers about reading and writing many types of communications and development artifacts as "informalisms" [25], as well as addressing new kinds of non-functional requirements like project community development, freedom of expression and choice, and ease of information space navigation. Elsewhere, there is widespread recognition that F/OSSD projects differ from their traditional software engineering counterparts in that F/OSSD projects do not in general operate under the constraints of budget, schedule, and project management constraints. In addition, OSS developers are also end-users or administrators of the software products they develop, rather than conventionally separated as developers and/versus users. Consequently, it appears that F/OSSD projects create different types of software requirements using a different kind of requirements engineering process, than compared to what the software engineering community has addressed. Thus, there is a fundamental need to discover and understand the process of requirements development in different types of F/OSSD projects.

We need an appropriate mix of concepts, techniques, and tools to discover and understand F/OSSD processes. We and others have found that process ethnographies must be empirically grounded, evidence-based, and subject to comparative, multi-perspective analysis [3,7,10,15,22,25,28]. However, we also recognize that our effort to discover and understand F/OSSD processes should reveal the experience of software development newcomers who want to join and figure out how things get done in the project [27].

As participant observers in such a project, we find that it is common practice for newcomers to navigate and browse the project's Web site, development artifacts, and computer-mediated communication systems (e.g., discussion forums, online chat, project Wikis), as well as to download and try out the current software product release. Such traversal and engagement with multiple types of hyperlinked information provide a basis for making modest contributions (e.g., bug reports) before more substantial contributions (code patches, new modules) are offered, with the eventual possibility of proposing changing or sustaining the OSS system's architecture. These interactive experiences reflect a progressive validation of a participant's understanding of current F/OSSD process and product requirements [1,19]. Thus, we seek a process discovery and modeling scheme that elicits, analyzes, and validates multi-mode, hypertext descriptions of a F/OSSD project's requirements process. Furthermore, these process descriptions we construct should span informal through formal process models, and accommodate graphic, textual, and computationally enactable process media. Finally, our results should be in a form open to independent analysis, validation, extension, and redistribution by the project's participants.

3. Multi-Mode Process Modeling, Analysis and Validation using Ethnographic Hypermedia

An ethnographic hypermedia [4] is a hypertext that supports comparative, cross-linked analysis of multiple types of qualitative ethnographic data [cf. 28]. They are a kind of semantic hypertext used in coding, modeling, documenting, and explaining patterns of social interaction data and analysis arising in contemporary anthropological, sociological, and distributed cognition studies. The media can include discourse records, indigenous texts, interview transcripts, graphic or photographic images, audio/video recordings, and other related information artifacts. Ideally, they also preserve the form and some of the context in which the data appear, which is important for subsequent (re)analysis, documentation, explanation, presentation and validation.

Ethnographic studies of software development processes within Web-based F/OSSD projects are the focus here. Ethnographic studies that observe and explain social action through online participant observation and data collection have come to be called "virtual ethnography" [12]. Virtual ethnography techniques have been used to observe the work practices, compare the artifacts produced, and discover the processes of F/OSSD projects found on and across the Web [5,6,13,14,23,25,26,27]. In particular, an important source of data that is examined in such studies of F/OSSD projects is the interrelated web of online documents and artifacts that embody and characterize the medium and continuously emerging outcomes of F/OSSD work. These documents and artifacts constitute a particular narrative/textual genre ecology [29] that situate the work practices and characterize the problem solving media found within F/OSSD projects.

We have employed ethnographic hypermedia in our virtual ethnographic studies of F/OSSD projects. What does this mean, and what challenges or opportunities for requirements elicitation, analysis, and validation have emerged along the way? These questions are addressed below through examples drawn from a case study of the NetBeans.org OSSD project [13,14], which is one of the largest F/OSSD projects we have studied. The NetBeans.org project is a corporate sponsored OSSD project [13] focused on the development of an interactive development environment (IDE) for constructing application systems using Java enterprise beans technology. It is similar is size and scope to the Eclipse project (formerly) sponsored by IBM, which is also developing a Java-based IDE.

As noted, the F/OSSD projects we study are found on the Web. Web sites for these projects consist of a network of hyperlinked documents or artifacts. Samples of sites we have studied

include NetBeans.org, Mozilla.org, Apache.org, and GNUenterprise.org among others [5,6,13,14,25,26]. A team of 2-5 researchers examines a project site (via browsing, search, download, and cross-linking) over a period of 4-6 weeks initially, then periodically thereafter. The artifacts we examine include Web pages, email discussion lists, bug reports, project to-do lists, source code files and directories, site maps, and more. These artifacts are an important part of the data we collect, examine, study, code, and analyze in order to identify F/OSSD work practices and development processes that arise in a given project.

We create a hypermedia of these artifacts in ways that allow us to locate the originating source(s) of data within the focal project's Web site. This allows us to maintain links to the source data materials that we observe as evidence of the process at hand, as well as to allow us to detect when these data sources have been updated or removed. (We also archive a local copy of all such data). However, we create annotated and assembled artifacts that embed hyperlinks to these documents as part of our ethnographic hypermedia. As a result, multiple kinds of ethnographic records are created including annotated artifacts, rich hypermedia pictures, and ethnographic narratives. Juxtaposed about these records are other kinds of models including a process meta-model, attributed directed graph model, process domain ontology, and a formal, computationally enactable process model. Each is described next, and each is hyperlinked into an overall ethnographic hypermedia that provides cross-cutting evidence for the observed OSS requirements processes.

Annotated artifacts

Annotated artifacts represent original software development artifacts like (publicly available) online chat transcripts that record the dialogue, discussions, and debate that emerge between

OSS developers. These artifacts record basic design rationale in an online conversation form. The textual content of these artifacts can be tagged, analyzed, hyperlinked, and categorized manually or automatically [24]. However, these conversational contents also reveal much about how OSS developers interact at a distance to articulate, debate, and refine the continuously emerging requirements for the software system they are developing. For example, Elliott and Scacchi [5,6] provide conversational transcripts among developers engaged in a debate over what the most important properties of software development tools and components to use when building free software. They provide annotations that identify and bracket how ideological beliefs, social values, and community building norms constrain and ultimately determine the technical choices for what tools to use and what components to reuse when developing OSS.

Navigational rich pictures

Rich pictures [18] provide an informal graphical scheme for identifying and modeling stakeholders, their concerns, objects and patterns of interaction. We extend this scheme to form navigational rich pictures constructed as an Web-compatible hypertext image map that denotes the overall context as the composition and relationships observed among the stakeholder-roles, activities, tools, and document types (resources) found in a F/OSSD project. Figure 1 displays such a rich picture constructed for NetBeans.org. Associated with each relationship is a hyperlink to a *use case* [2] that we have constructed to denote an observable activity performed by an actor-role using a tool that consumes or produces a document type. An example use case is shown in Figure 2. Each other type of data also is hyperlinked to either a descriptive annotation or to a Web site/page where further information on the object type can be found.

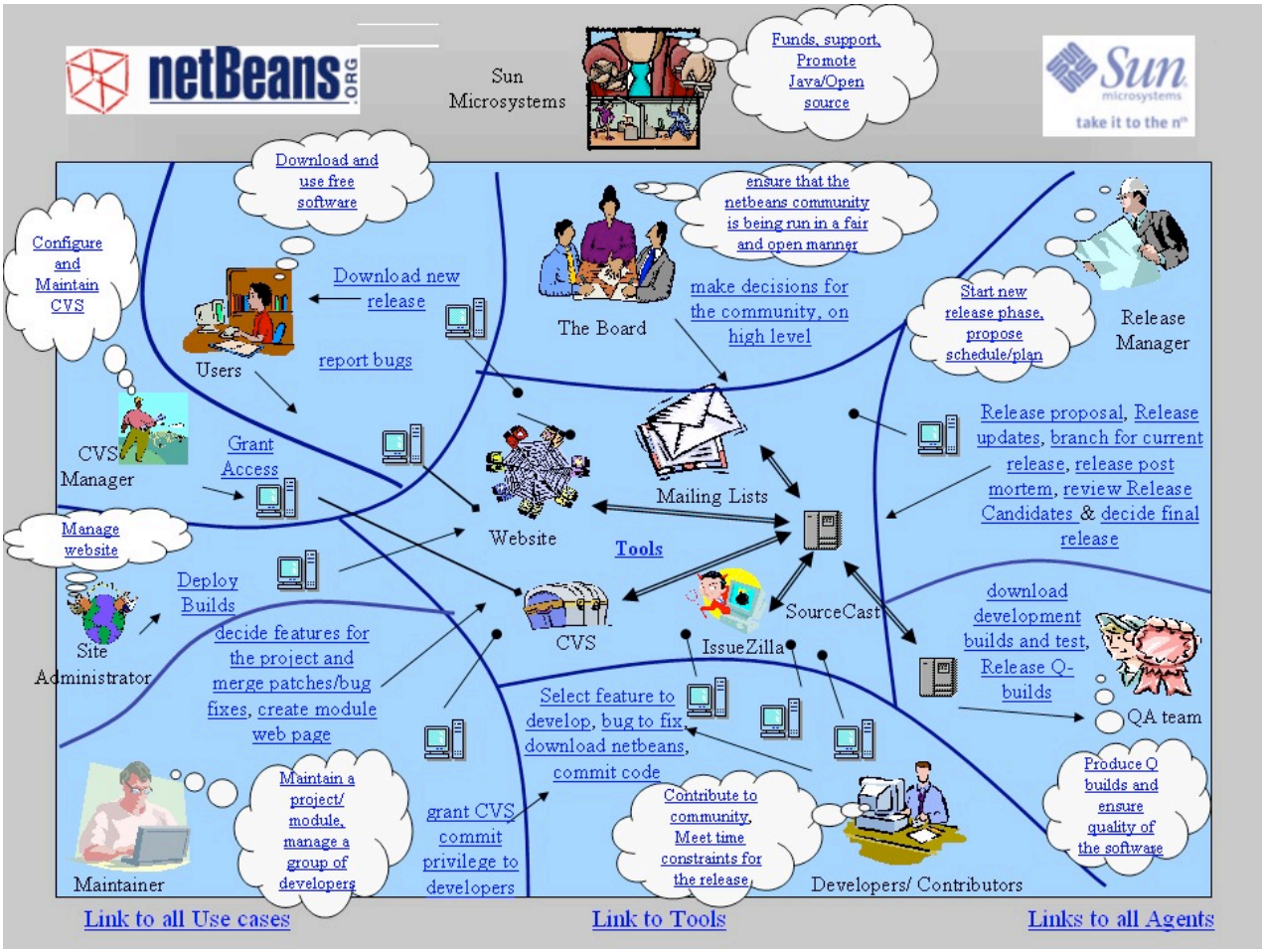

Figure 1. A rich picture image map of the requirements and release process in NetBeans.org

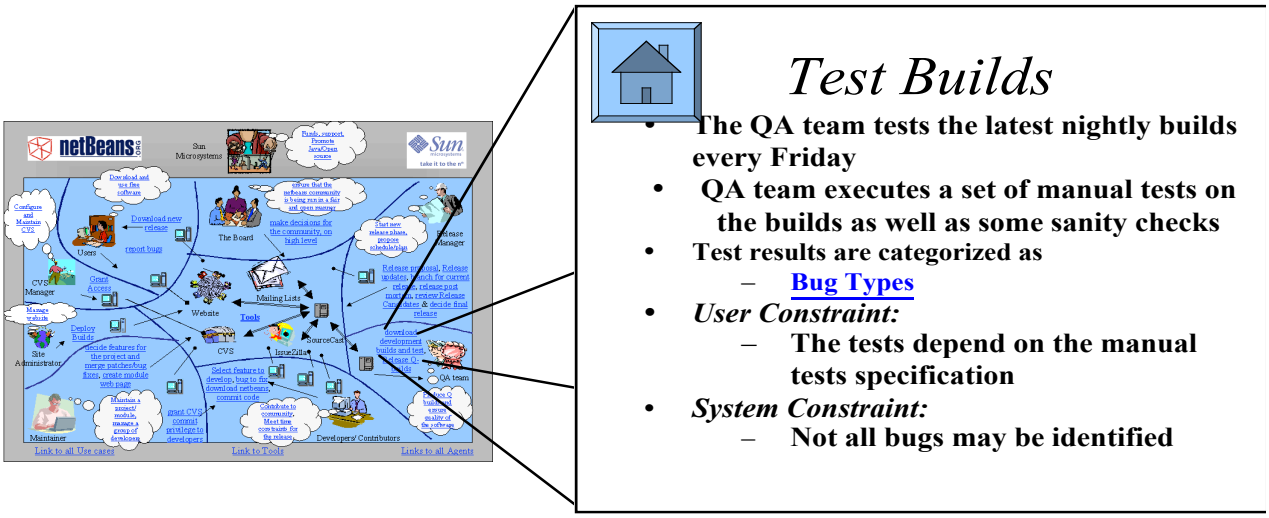

Figure 2. A hyperlink selection within a rich hypermedia presentation that reveals a corresponding use case.

Directed resource flow graph

A directed resource flow graph denotes a recurring workflow pattern that has been discovered in an F/OSSD project. These workflows order the dependencies among the activities that actor-roles perform on a recurring basis to the objects/resources within their project work. These resources appear as or within Web pages on an F/OSSD project's Web site. For example, in the NetBeans.org project, we found that software product requirements are intertwined with software build and release management. Thus, the "requirements and release process" entails identifying and programming new/updated system functions or features in the course of compiling, integrating, testing, and progressively releasing a stable composition of source code files as an executable software build version for evaluation or use by other NetBeans.org developers [5,6,23]. An example flow graph for this appears in Figure 3. The code files, executable software, updated directories, and associated email postings announcing the completion and posting the results of the testing are among the types of resources that are involved. Last, the rendering of the flow graph can serve as an image map to the online (i.e., on the NetBeans.org Web site) data sources from where they are observed.

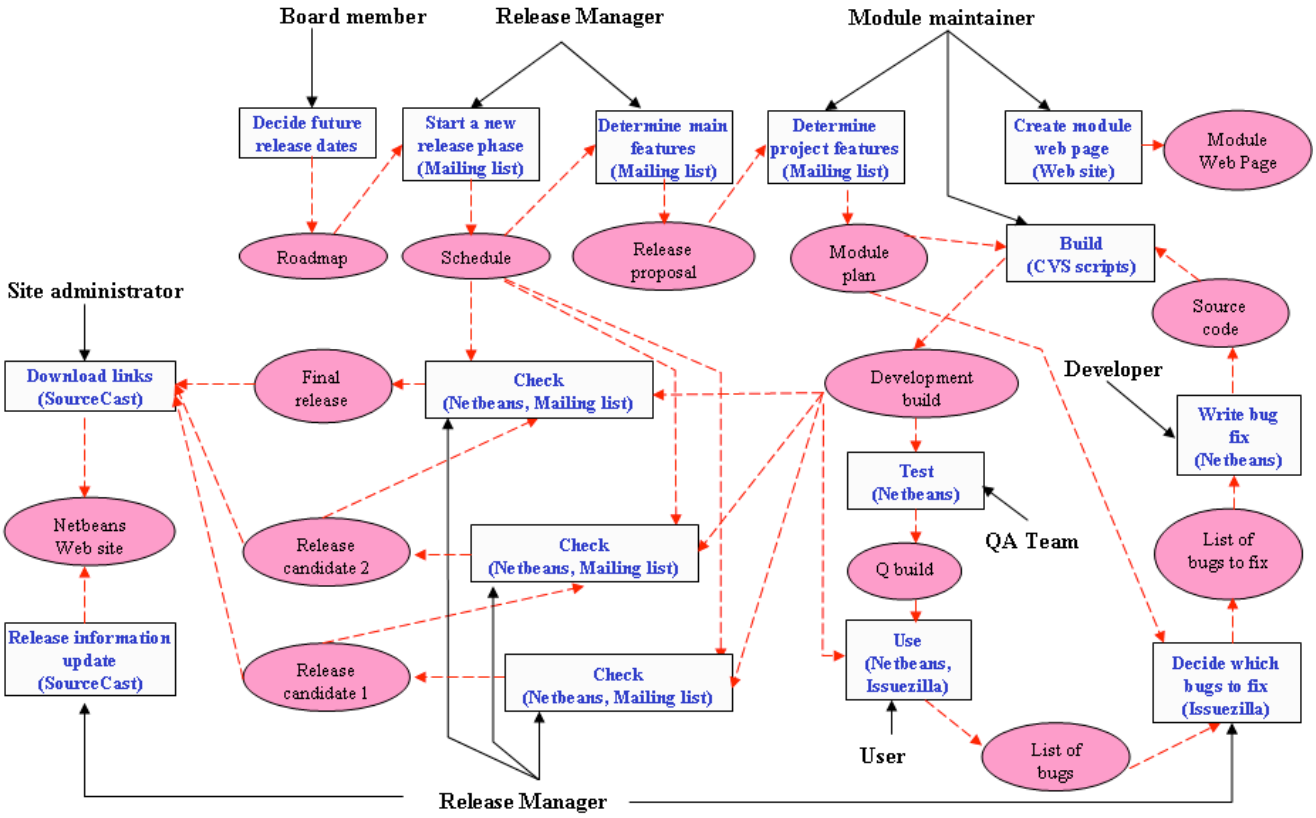

Figure 3. An attributed directed graph of the resource flow for the NetBeans.org requirement and release process. Boxes denote tasks/actions, ellipses denote resources/objects, dashed lines denote resource flows, and solid lines and labels denote agent/stakeholder roles performing tasks that transform input resources into output resources.

Process domain ontology

A process ontology represents the underlying *process meta-model* [17,20] that defines the semantics and syntax of the process modeling constructs we use to model discovered processes. It provides the base object classes for constructing the requirements process (domain) taxonomies of the object classes for all of the resource and relation types found in the rich picture and directed resource flow graph. However, each discovered process is specific to an F/OSSD project, and knowledge about this domain is also needed to help contextualize the possible meanings of the processes being modeled. This means that a process domain entails objects, resources or relations that may or may not be have been previously observed and

modeled, so that it may be necessary to extend to process modeling constructs to accommodate new types of objects, resources, and relations, as well as the attributes and (instance) values that characterize them, and attached methods that operationalize them.

We use an ontology modeling and editing tool, *Protégé-2000* [21], to maintain and update our domain ontology for OSS requirements processes. Using Protégé-2000, we can also visualize the structure of dependencies and relations [11] among the objects or resources in a semantic web manner. An example view can be seen in Figure 4. Furthermore, we can create translators that can transform syntactic form of the modeling representations into XML forms or SQL schema definitions, which enables further process modeling and tool integration options [cf. 14].

Formal process model and its enactment

A formal process model denotes a syntactically precise and semantically typed specification of the resource objects, flow dependencies, actor-roles, and associated tools that specifies an enactable (via interactive process-guided user navigation) hypertext representation we call an *organizational process hypertext* [20]. This semantic hypertext, and its supporting run-time environment, enables the ability to walkthrough or simulate enactment of the modeled F/OSSD process as a process-guided, navigational traversal across a set of process linked Web pages. The semantic hypertext is automatically rendered through compilation of the process models that are output from the ontology editor in a process modeling language called PML [20]. A PML-based model specification enables automated consistency checking at compile-time, and detection of inconsistencies at compile-time or run-time. An example of an excerpt from such a model is shown in Figure 5. The compiled version of the PML produced a non-linear sequence of process-linked Web pages, each one of which corresponds to one step in the modeled process. An

example showing the result of enacting a process (action) step specified at the bottom of Figure 5 appears in Figure 6.

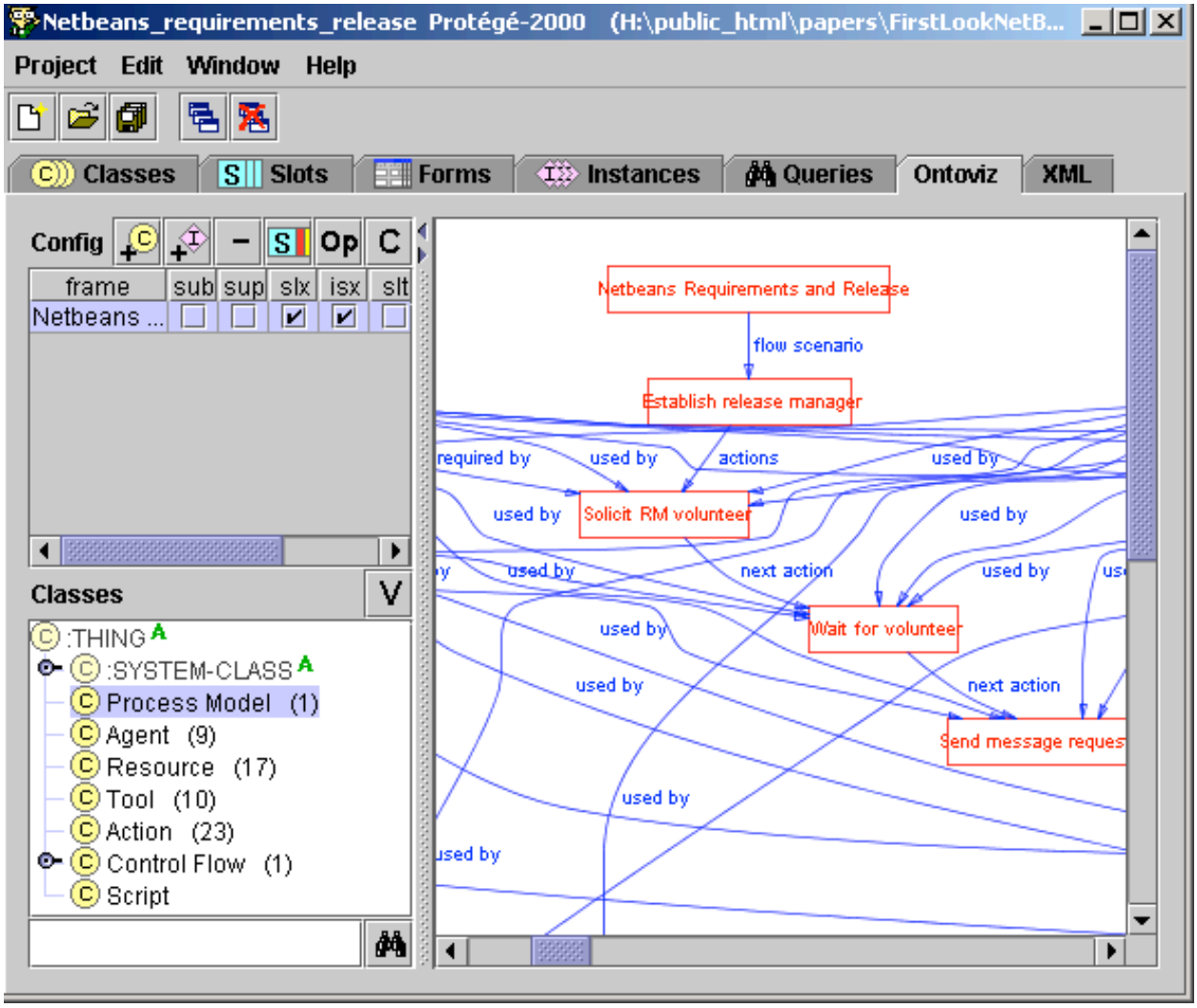

Figure 4. A view of the process domain ontology for the NetBeans.org software requirements and release process.

```
...
sequence Test {
  action Execute automatic test scripts {
  requires { Test scripts, release binaries }
  provides { Test results }
  tool { Automated test suite (xtest, others) }
  agent { Sun ONE Studio QA team }
  script { /* Executed off-site */ } }
action Execute manual test scripts {
  requires { Release binaries }
  provides { Test results }
  tool { NetBeans IDE }
  agent { users, developers, Sun ONE Studio QA team, Sun ONE Studio developers }
  script { /* Executed off-site */ } }
iteration Update Issuezilla {
  action Report issues to Issuezilla {
    requires { Test results }
    provides { Issuezilla entry }
    tool { Web browser }
    agent { users, developers, Sun ONE Studio QA team, Sun ONE Studio developers }
    script {
      <br><a href="http://www.netbeans.org/issues/">Navigate to Issuezilla </a>
      <br><a href="http://www.netbeans.org/issues/query.cgi">Query Issuezilla </a>
      <br><a href="http://www.netbeans.org/issues/enter_bug.cgi">Enter issue </a> } }
...
```

Figure 5. An excerpt of the formal model of the Netbeans.org requirements and release process coded in PML.

Ethnographic hypermedia narrative for validation

An ethnographic narrative denotes the final view ethnographic hypermedia. This is an analytical research narrative that is structured as a document that is (ideally) suitable for dissemination and publication in Web-based and printed forms. It is a composite derived from selections of the preceding representations in the form of a narrative with embedded hyperlinked objects, and hyperlinks to related materials. It embodies and explains the work practices, development processes, resource types and relations, and overall project context as a narrative, hyperlinked ethnographic account that discovered at play within a given F/OSSD project, such as we

documented for the NetBeans requirements and release process [23]. In printed form, the narratives we have produced so far are somewhere between 1/4 to 1/15 the number of pages compared to the overall set of project-specific data (documents) at the first two levels of hyperlink connectivity; said differently, if the ethnographic report is 30 or so printed pages (i.e., suitable for journal publication), the underlying ethnographic hypermedia will correspond to a hypermedia equivalent to 120-450 printed pages.

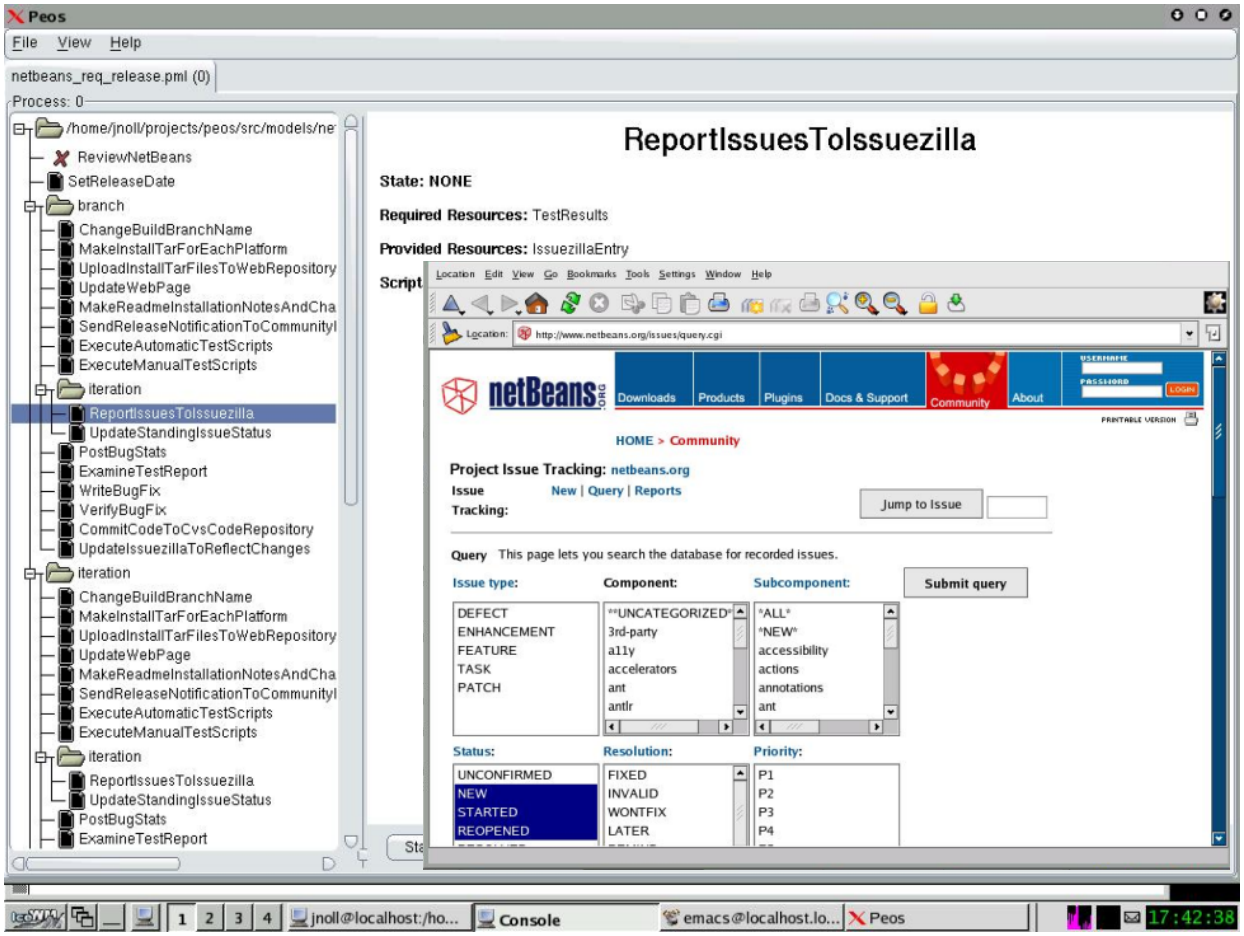

Figure 6. A screenshot displaying the result of the PML-based re-enactment of one step ("Action Report issues to Issuezilla") in the NetBeans,org requirements and release process.

The narrative is in a form intended for external review and validation by those not involved in the collection, modeling, and analysis activities, such as members of the project under study (NetBeans.org—see Figure 7). These external reviewers can read through the narrative during validation to see if there are gaps or inconsistencies, or to pose questions to the narrative's authors. When such shortfalls or queries are detected or reported, then the task is to determine if the problem arises from either a gap in the modeling effort, or in its narrative rendering. Finally, the narrative and its hypermedia components are envisioned as open and living documents, so that feedback from the community may serve to keep them consistent with current practice, or to detect and report inconsistencies that are in need of attention, update, or remediation, much like the software and artifacts found in the F/OSSD projects they describe.

4. Discussion

We have learned a number of things based on applying our approach to modeling development processes, such as those for software requirements, in different F/OSSD projects. First, no single mode of process description adequately subsumes the others, so there is no best process description scheme. Instead, different informal and formal descriptions respectively account for the shortcomings in the other, as do textual, graphic, and computationally enactable process representations. Second, incremental and progressive elicitation, analysis, and validation occur in the course of developing multi-mode requirements process models. Third, multi-mode process models are well-suited for discovery and understanding of complex software processes found in F/OSSD projects. However, it may not be a suitable approach for other software projects that do not organize, discuss, and perform software development activities in an online, persistent, open, free, and publicly accessible manner. Fourth, multi-mode process modeling has the potential to be applicable to the discovery and modeling of software product requirements, although the

motivation for investing such effort may not be clear or easily justified. Process discovery is a different kind of problem than product development, so different kinds of approaches are likely to be most effective.

Figure 7. Getting captured and analyzed process models out for validation and possible evolution by NetBeans.org project participants.

Last, we observed that the software product requirements in F/OSSD projects are continually emerging and evolving. Thus, it seems likely that the requirements process in such projects is

also continuously. Thus, supporting the evolution of multi-mode models of OSS requirements processes will require either automated techniques for process discovery and multi-mode update propagation techniques, or else the participation of the project community to treat these models as open source software process models, that can be continuously elicited, analyzed, and validated along with other F/OSSD project assets, as suggested in Figure 7, which are concepts we are currently investigating. However, it seems fair to note that ethnographic accounts are situated in time, and are not intended for evolution.

5. Conclusion

Ethnographic hypermedia are an important type of semantic hypertext that are well-suited to support the navigation, elicitation, modeling, analysis and report writing found in ethnographic studies of F/OSSD processes. We have described our approach to developing and using ethnographic hypermedia to support the modeling, analysis, and validation of software development processes in F/OSSD projects like NetBeans.org, where multiple modes of informal to formal representations are involved. We find that this hypermedia is well-suited for supporting qualitative research methods that associated different type of project data, together with comparative analysis of process descriptions rendered in graphic, textual and computationally enactable descriptions. We provided examples of the various kinds of hypertext-based process descriptions and linkages that we constructed in moving from abstract, informal representations of the data through a series of ever more formalized process models resulting from our studies.

Based on our efforts and results reported here, it appears that free/open source software development projects can benefit from the discovery, modeling, and validation of the development processes they practice, and that ethnographic hypermedia based representations of

these processes provides an innovative scheme for capturing, representing, and evolving these representations in a manner that can be maintained and evolved in an open source manner.

6. Acknowledgements

The research described in this report is supported by grants #0083075, #0205679, #0205724, and #0350754 from the U.S. National Science Foundation. No endorsement implied. Mark Ackerman at University of Michigan, Ann Arbor; Les Gasser at University of Illinois, Urbana-Champaign; and others at ISR are collaborators on the research described in this paper.

7. References

1. Bolchini, D. and Paolini, P., Goal-Driven Requirements Analysis for Hypermedia-Intensive Web Applications, *Requirements Engineering,* 9, 85-103, 2004.

2. Cockburn, A., *Writing Effective Use Cases*, Addison-Wesley, New York, 2001.

3. Curtis, B., Krasner, H., and Iscoe, N., A Field Study of the Software Design Process for Large Systems, *Communications ACM*, 31(11), 1268-1287, 1998.

4. Dicks, B. and Mason, B., Hypermedia and Ethnography: Reflections on the Construction of a Research Approach, *Sociological Research Online*, 3(3), 1998. www.socresonline.org.uk

5. Elliott, M. and Scacchi, W., Free Software Developers as an Occupational Community: Resolving Conflicts and Fostering Collaboration, *Proc. ACM Int. Conf. Supporting Group Work*, 21-30, Sanibel Island, FL, November 2003.

6. Elliott, M. and Scacchi, W., Free Software Development: Cooperation and Conflict in A Virtual Organizational Culture, in S. Koch (ed.), *Free/Open Source Software Development*, Idea Group Publishing, Hershey, PA, 152-172, 2004.

7. Finkelstein, A.C.W., Gabbay, D., Hunter, A., Nuseibeh, B., Inconsistency Handling in Multi-perspective Specifications, *IEEE Trans. Software Engineering,* 20(8), 569-578, 1994.

8. Gans, G., Jarke, M., Kethers, S., and Lakemeyer, G., Continuous Requirements Management for Organisation Networks: A (Dis)Trust-Based Approach, *Requirements Engineering*, 8, 4-22, 2003.

9. Gasser, L., Scacchi, W., Penne, B., and Sandusky, R., Understanding Continuous Design in OSS Projects, *Proc. 16th. Int. Conf. Software & Systems Engineering and their Applications*, Paris, December 2003.

10. Glaser, B. and Strauss, A., *The Discovery of Grounded Theory: Strategies for Qualitative Research,* Aldine Publishing Co., Chicago, Il, 1967.

11. Grinter, R.E., Recomposition: Coordinating a Web of Software Dependencies, *Computer Supported Cooperative Work,* 12(3), 297-327, 2003.

12. Hine, C., *Virtual Ethnography*, Sage Publications, Newbury Park, CA, 2000.

13. Jensen, C. and Scacchi, W., Collaboration, Leadership, Control, and Conflict Management in the NetBeans.Org Community, *Proc. 38th. Hawaii Intern, Conf. Systems Science,* Waikola Village, HI, January 2005a.

14. Jensen, C. and Scacchi, W., Process Modeling Across the Web Information Infrastructure, *Software Process--Improvement and Practice,* 10(3), 255-272, July-September 2005b.

15. Kitchenham, B.A., Dyba, T., and Jorgensen, M., Evidence-based Software Engineering, *Proc. 26th Int. Conf. Software Engineering*, 273-281, Edinburgh, Scotland, IEEE Computer Society, 2004.

16. Leite, J.C.S.P. and Freeman, P.A., Requirements Validation through Viewpoint Resolution, *IEEE Trans. Software Engineering*, 17(12), 1253-1269, 1991.

17. Mi, P. and Scacchi, W., A Meta-Model for Formulating Knowledge-Based Models of Software Development, *Decision Support Systems*, 17(4), 313-330, 1996.

18. Monk, A. and Howard, S., The Rich Picture: A Tool for Reasoning about Work Context, *Interactions*, March-April 1998.

19. Narayanan, N.H. and Hegarty, M., Multimedia Design for Communication of Dynamic Information, *Int. J. Human-Computer Studies*, 57, 279-315, 2002.

20. Noll, J. and Scacchi, W., Specifying Process-Oriented Hypertext for Organizational Computing, *J. Network & Computer Applications*, 24(1), 39-61, 2001.

21. Noy, N.F., Sintek, M., Decker, S., Crubezy, M., Fergerson, R.W., and Musen, M.A., Creating Semantic Web Contents with Protégé-2000, *IEEE Intelligent Systems*, 16(2), 60-71, March/April 2001.

22. Nuseibeh, B. and Easterbrook, S., Requirements Engineering: A Roadmap, in Finkelstein, A. (ed.), *The Future of Software Engineering*, ACM and IEEE Computer Society Press, 2000.

23. Oza, M., Nistor, E., Hu, S. Jensen, C., and Scacchi, W. A First Look at the NetBeans Requirements and Release Process, http://www.ics.uci.edu/cjensen/papers/FirstLook

NetBeans/, February 2004 (Original May 2002).

24. Rao, R., From Unstructured Data to Actionable Intelligence, *IT Pro*, 29-35, November 2003.

25. Scacchi, W., Understanding the Requirements for Developing Open Source Software Systems, *IEE Proceedings—Software,* 149(1), 24-39, February 2002.

26. Scacchi, W., Free/Open Source Software Development Practices in the Computer Game Community, *IEEE Software*, 21(1), 59-67, Jan. 2004a.

27. Scacchi, W., Socio-Technical Interaction Networks in Free/Open Source Software Development Processes, in S.T. Acuña and N. Juristo (eds.), *Peopleware and the Software Process*, World Scientific Press, to appear, 2004b.

28. Seaman, C.B., Qualitative Methods in Empirical Studies of Software Engineering, *IEEE Trans. Software Engineering*, 25(4), 557-572, 1999.

29. Spinuzzi, C. and Zachry, M., Genre Ecologies: An Open System Approach to Understanding and Constructing Documentation, *ACM J. Computer Documentation,* 24(3), 169-181, August 2000.

30. Truex, D., Baskerville, R., and Klein, H., Growing Systems in an Emergent Organization, *Communications ACM*, 42(8), 117-123, 1999.

31. Viller, S. and Sommerville, I., Ethnographically Informed Analysis for Software Engineers, *International Journal Human-Computer Studies,* 53, 169-196, 2000.