Discovery and Analysis of Processes in Virtual Enterprises

Chris Jensen and Walt Scacchi
Institute for Software Research
School of Information and Computer Science
University of California, Irvine
Irvine, CA USA 92697-3425
{cjensen, wscacchi}@ics.uci.edu
August 2003

(submitted for review)

Abstract

Intelligence informatics has led to techniques for mining Web data, but it has yet to relate how we can combine usage, content, tool, and resource data discovered from online information spaces into enterprise processes. Intelligence and security agencies may operate as virtual enterprises that perform such processes. This paper addresses the discovery and analysis of processes operating within virtual enterprises. The process discovery method employs a process meta-model to organize and filter information gathered from the public information space of a virtual enterprise. This gives rise to construction of an informal flow-graph model, semi-structured rich hypermedia model, and a formal computationally enactable model of the process being analyzed.

Keywords

Web-based Intelligence Mining, Process Discovery, Virtual Enterprise

Discovery and Analysis of Processes in Virtual Enterprises

1. Introduction

One important line of research in intelligence and security informatics addresses the discovery, management, and protection of knowledge-intensive information artifacts for a complex enterprise. These artifacts may include plans for action, email messages, images, Web pages, reports, executable application programs, and others. The distribution, access control, presentation, and maintenance of these artifacts are also part of such endeavors. While much work has investigated these phenomena individually, these issues are manifest within enterprise processes--processes that people or computational agents perform to create, update, or consume information artifacts. These processes require information to be shared among large numbers of knowledge customers acting in different roles that require varying degrees of knowledge for how the processes operate. These actors use information processing tools/application programs (email, Web browsers, instant messaging services, document compilers, publication servers, etc.) to create, update, consume, or otherwise share this information. However, current informatics efforts directed at knowledge discovery and data mining from text or Web sites do not yet reveal what these processes are, or the enterprise contexts in which they occur.

The research problem addressed in this paper is how to discover and analyze complex processes operating within a virtual enterprise (VE). VEs can be characterized as those whose decision-making or resource allocation activities are logically (hence administratively) centralized, but who activities, actors, and artifacts are physically decentralized, and all are primarily brought together or made accessible over the Internet. For example, data fusion, intelligence analysis, and executive briefings are often centralized, whereas the agents and sensors that collect the data or

affect local action in response to an analysis or directive from an executive decision maker are geographically dispersed. Many types of enterprises including intelligence and security agencies, "organized threats" like terrorist groups, and business competitor intelligence units operate as VEs. Consequently, knowledge and explicit representation of the processes operating within such VEs becomes a key enabler for collective process analysis and interpretation, inferred prediction of process enactment trajectories, process interdiction and disruption, or process redesign and continuous process improvement. However, these types of VEs and their processes are generally not available for empirical study with public dissemination of results.

The solution to the research problem presented here entails development and demonstration of an approach to the discovery and analysis of processes operating within a VE. It presents a case study to demonstrate and examine the approach and results. The approach employs a variety of automated tools and knowledge-based techniques to codify and analyze process data from Web-based information resources. However, the approach is specifically designed with the goal of being more fully automated.

Once a VE process is discovered and explicitly represented and modeled, it may provide insights to support decision making, resource allocation, policy setting, and more. The discovery process also yields insight of its own as a means of competitive intelligence by demonstrating the ability of third parties to discover a competitor's processes through mining and analysis of publicly accessible information spaces. Since a VE may rely on an Internet application infrastructure like the Web to communicate and post (updates to) information artifacts/contents, it may also engage in encryption and other information hiding actions. We do not address encryption/decryption

issues here, though actors performing automated encryption, routing, or decryption of information artifacts may engage in detectable actions that denote their creation, usage, update, or consumption within a VE process. As a result, the study presented here for mining process data and related artifacts examines an open VE whose source data is publicly available[1] as a surrogate of those VEs that we cannot directly study, or whose results may not be published due to confidentiality or security constraints. Identifying and rationalizing how such a surrogate is selected will be described as part of the research method presentation.

The remaining sections of this paper are organized as follows. First is a discussion of related research that helps identify some important contributions that may be compared to the approach presented here. Next is an overview of the research method that was used to examine the data, conduct the analysis, and to present results from our approach to process discovery in a VE. Following this is a more detailed examination of the approach. This includes discussion of the criteria used to select a surrogate VE and process to examine in this study. The selected surrogate VE is the NetBeans enterprise that develops a large open source software system and related artifacts. Next is a presentation of the analysis that elaborates the purpose and role of a process meta-model to help discover and construct a semi-structured hypermedia model, and then a formal enactable version of the process model that follows in a case study. The selected process being examined is what we call the requirements and release process at NetBeans. This process entails a sequence of activities that collect, review, and progressively refines a group of related artifacts that are progressively stabilized and incrementally released. A discussion of issues and challenges that arise in this approach to discovery and analysis of processes within a VE, as well

[1] As prior research reveals that most enterprises do not possess or maintain explicit models of their processes [33], we examine a public VE that also does not have such process representations at its disposal.

as identification of remaining steps in moving toward a more fully automated process discovery capability then completes this paper, prior to presenting conclusions as to what was accomplished, what's missing, and what research activities may follow.

2. Related Work

Our research methods incorporate Web-based information mining, process modeling, and process meta-modeling [16]. Other representative approaches to Web data mining include the following. Cooley, *et al*. [6] are among those examining tools and methods for data mining on the Web where analysis must be done on remote objects accessed from a Web server. This is the case with process discovery in virtual enterprises. They propose a system architecture for Web usage mining and a taxonomy of efforts in Web mining. While this does not solve the process discovery problem, the approach of Cooley and associates does give us some foundational techniques needed to discover process information. The drawback of their approach is that it requires the participation of Web users, albeit mildly, but operates as the equivalent of a phone tap on the user's Web browser. Cadez, *et al*. [2] attempted to uncover Web site usage patterns without such a tap, but their underlying Markov Model, did not allow for assertions of Web tour patterns to be made. Finally, other approaches to Web usage data mining, such as WebQuilt [12] seek to discover and visualize the navigation patterns of end-users that visit and traverse a Web site. WebQuilt relies on use of a proxy server that effectively encapsulates or wraps the designated Web site to monitor and discovery Web site usability patterns, but does not seek to identify or associate these patterns into processes.

Chen, *et al*. [3], present a powerful technique for monitoring Web information spaces for counterintelligence of decision-making process data. However, their work does not attempt to

construct a process model using the information they discover. Similarly, though link analysis [35] is a central activity to process discovery, current link analysis work seeks merely to extract and discover where relationships exist between data, but does not extend to discovering sequences of events that relate the people, resources, objects, and locations involved. While Zhao, *et al.* [38] construct a meta-model to describe collaborative efforts among law enforcement agencies, current research in workflow automation [36] does not shed light on recovering a process from enactment evidence. Thus, it is somewhat unclear whether an outside organization could piece the results together to form a model of the process by simply using available Web mining techniques.

Probabilistic relational modeling [11] is a data mining technique for extracting paths, flows, or trajectories through a network of (partially) ordered states or events. Such states or events may be found in a corpus of text like a collection of documents within a domain-specific digital library through textual data mining techniques. PRM and text data mining thus offer the potential for automated construction of process models from textual sources that collectively describe one or more process flows or knowledge flows. However, existing efforts have not applied these techniques to the problem of discovery and analysis of processes within a virtual enterprise, as is the focus here. In contrast, Hwang and Yang [13] propose an approach similar to PRM to discover and model processes derived from a recorded sequence of process activities or events that possess a timestamp attribute. With this information, Hwang and Yang demonstrate how it is possible to construct complex process flow sequences including iterative loops and conditional actions using their reconstruction algorithms. However, they do not provide nor claim how their approach might be used to collect and analyze data from Web sites or to processes within a VE.

The explicit modeling and analysis of enterprise processes in forms suitable for enactment in a computer-based or Web-based manner continues to receive growing research and commercial attention. For example, the RosettaNet [31] industry consortium has invested years of effort to produce visual models of common, recurring business processes that enable inter-enterprise commerce between trading partners. Elsewhere, Microsoft, IBM, SAP and other companies are exploring the development of business process execution language that can be automated and enacted through the invocation of Web services, called BPEL4WS [15]. Such a language might be used to model and enact the processes associated with the RosettaNet effort. However, neither of these efforts addresses process discovery or analysis of such processes.

In contrast, Noll and Scacchi [23] describe a Web-based process modeling language and semantic Web-based enactment framework for process coordinated artifact creation, update, and sharing among multiple actors and information resources in a physically distributed VE. Such processes can be modeled, analyzed, simulated, redesigned, visualized, and enacted across the Web by actors/agents in different enterprise roles to create, update, or consume information artifacts given the automated tools and related information resources at hand. However, their approach assumes the enterprise process is already known or previously prescribed, in order to specify a process suitable for automated support. Similarly, their approach does not address how to discover a process from enterprise intelligence or associated information resources.

Last, some initial efforts at process discovery from logs or event/transaction records have been explored by Cook and Wolf [5], and Hwang and Yang [13]. In both of these approaches, the

records for analysis are comparatively simple and lack any contextual artifacts or enterprise modeling clues, and characterize non-Web-based processes found in centralized enterprises.

As such, the approach presented here seeks to build from the prior related research efforts such as those described above, but seeks to incorporate or fill-in where these other efforts have not yet prevailed. No attempt is made at this time to fully automate the process discovery and analysis activity. Instead, effort focuses on determining how to perform such an endeavor as a step to help guide where different kinds of knowledge discovery and data mining techniques and automation may be effectively applied.

3. Research Method

As already noted, it may not be possible or appropriate to study the operations, artifacts, or processes of intelligence or security agencies, organized threats, or business competitors in a way where the results can be presented in the public for review or reuse. Accordingly, the first step in our research method must therefore address how an appropriate surrogate VE and one or more of its processes can be selected for close study.

The next step, once a study site has been selected, is to describe what kinds of data or information must be gathered or remotely acquired, as well as how to organize it. For this we employ a proven and well-refined process meta-model scheme that has been developed and used for a number of years in a variety of enterprise process application domains [16, 22, 23, 33]. This process meta-model provides an ontological framework that organizes, filters, and represents that collected data and related information artifacts into a taxonomic scheme suitable for progressively constructing explicit process models with increasing formality. In this work, we

use a contemporary knowledge editing and ontology modeling tool, Protégé-2000 [29], to implement our process meta-model as a domain ontology, which will then enable us to construct, update, and export process models in a process markup language [23].

With the selected VE and process modeling framework in hand to help guide the effort, the next activity is data mining the information artifacts found on the VE's Web site. This entails developing a taxonomic classification of the types of artifacts, actor roles, recurring actions, and tools that can be observed at a distance, as well as mining the artifacts content for timestamp attributes or update transactions. For example, email messages appearing in a discussion forum or threaded email bulletin board are typically marked with a user-id/email address and a timestamp for when it was posted for public access. Email discussion threads may then identify or discuss the actions of one or more actors or actor roles, together with the artifacts they manipulate and update. Other artifacts like Web pages may be updated with changes to objects that are mentioned or hyperlinked from the page to the object's location (URL). Monitoring these email lists and Web pages may then reveal a series of updates or transactions on artifact contents that provide an externally observable record of process activities or process fragments.

Next, the VE process under study can be modeled with a semi-structured hypermedia model of the context of the process being analyzed within its VE. Here construction of a rich picture [17] serves as the starting point, though it is augmented with typed (multi-media) objects and hyperlinks. Rich pictures are a descriptive visualization and modeling technique that focuses attention to capture and depiction of an enterprise setting, stakeholder/actor roles and concerns (goals and constraints), tools, resources, and the process action flows that interlink them.

Associated with the process objects appearing in our rich hypermedia picture are use cases [4], which are employed to extend the representational and semantic richness of the picture. *Use cases* serve as a semi-structured notational form for capturing and recording scenarios of activities performed by actors in some role that use one or more tools to manipulate artifacts associated with an enterprise process or activity within it. Use cases can be modeled and represented using the unified modeling language, UML [8], though that is not our choice here. Instead we simply seek to create process models using a language that can be computationally enacted across a semantic Web. Therefore, in this approach, use cases are hyperlinked to the rich picture via use case link types, and to a use sub-case hierarchy providing further detail to arbitrary depth. These use cases may also include hyperlinks to information resources or multi-media artifacts in the VE under study.

Given the rich hypermedia model of an enterprise and process context, we can next create a model of a process flow-graph. This flow-graph models the flow of resources between actors and actions, and is detailed to account for information captured in the use cases. In the visual representation of the flow-graph, these details appear as different types of node objects (e.g., artifacts appear as ellipses, actions as boxes, actor roles as labeled text pointing to the action boxes, and so forth), and the types correspond to those in the process meta-model [16]. This results in a process flow-graph that can be represented as a labeled or attributed directed graph structure. These semantic process graph structures can then be entered and edited using the Protégé-2000 tool that we have tailored for process modeling, through instantiation of our process meta-model as the modeling ontology.

Finally, we use our process meta-modeling and model editing tool to transform the formal process models that have been created into a computationally enactable process markup language that enables modeled processes to be analyzed, simulated, or deployed for enactment on a semantic web run-time environment [23]. This completes the basics of our research method by producing a model of a process discovered from a VE. In turn, such a model can also be deployed for reuse in another enterprise, or subject to further process engineering or redesign [32]. As such, the next section details this research method through a case study of process discovery in a VE.

4. Finding Appropriate Enterprises and Process to Examine

First, we need to find an appropriate surrogate VE to study, and then discover and analyze the details of one of its substantial processes.

4.1 Selecting a Surrogate Organization

As studying actual security agencies or large organized threats may be inappropriate for various reasons, we look instead to find alternate accessible organizations that operate in similar fashion to those we would desire to know. To do so, we first identify which situations and contexts are essential to emulate. Security organizations, and in some cases their illicit counterparts, are geographically distributed and decentralized. They have central authorities that distribute assignments, set policies, and allocate resources to their dispersed members. These agents act semi-autonomously and are loosely coupled to both the central authority and other members. In security organizations, a high degree of explicit collaboration would risk exposing sensitive information critical to their operations. With field operations ranging from highly coordinated to highly autonomous, coordination is done either directly through private channels, or indirectly

via public channels where their communication may be difficult to distinguish from that of others. Due to the considerable availability of public information and open participation, we explore open source software development (OSSD) projects as a sampling space for determining an appropriate surrogate VE to study.

OSSD projects operate as globally dispersed communities that can be found in many shapes and sizes. Most OSSD projects consist of small groups of people motivated by an otherwise unmet need for a particular software application, or by the desire to sharpen one's engineering skills for the sake of career advancement. While interesting on their own [34], these communities do not exhibit many of the properties we seek in a surrogate VE for study. Instead, we look at large scale OSSD projects which act as VEs, meaning that they have a central administrative authority or corporate sponsor that acts to coordinate and provide critical resources to insure the overall success of their decentralized software development work efforts.

Large OSSD project have upwards of hundreds to thousands of contributors who collectively construct complex information systems and webs of hyperlinked content consisting of thousands artifacts (design diagrams, source code files, system builds, online help documents, Web pages, etc.) that are posted for global access, review, or modification on the Web. O'Mahony [25] compares several large OSSD project enterprises and reveals differences in the scope of activities performed by their central authorities. These authorities have a broad range of control over the loosely coupled activities of geographically and temporally dispersed OSSD participants.

In the Linux Debian community, for example, the central authority has no control over the technical direction or development timeline of the Linux kernel or overall operating system. It serves merely as a "legal steward" to the project. It has no power to set policies or distribute assets and is a poor match for our purposes. Likewise, in the Apache project, its central authority, the Apache Software Foundation, cannot assign which tasks must be completed, when, or by whom. However, there are a number of large OSSD projects that have a centralized authority with the power to assign which tasks must be completed by when and by whom, that is also geographically distributed with the power to set policies governing the community. Examples here include the Eclipse project supported by IBM, the Gelato project supported by HP, and the Darwin project supported by Apple. Another example is NetBeans [20], which is sponsored by SUN Microsystems.

NetBeans is focused on the development of a software application platform for building enterprise-scale information system applications using the Java programming language and Enterprise Java Beans as system components. While many of the core developers are SUN employees based in Prague, it also benefits from an open-ended set of 2500 contributors who are based across the globe. The governance board of NetBeans keeps SUN's influence in check and ensures fairness in the community. One of the three members of the governance board is appointed by SUN. Technical direction in NetBeans is guided in part by SUN through its community members and also by the release manager. The release manager volunteers to coordinate efforts of the community members and ensure that the project stays on schedule and meets the goals set forth in the project roadmap. Development tasks are then assigned to persons of authority in the relevant subprojects, who are left to delegate and complete the work. Though

the argument could be made that many private corporations operate in such fashion, they may be encumbered by the same confidentiality constraints as security organizations that OSSD communities are less subject to. Thus, a VE such as the NetBeans OSSD project can be selected as suitable surrogate organization for study, though we make no claim that it is a replica of intelligence or security agencies, or their counterparts.

4.2 Selecting Appropriate Processes to Examine

As important as selecting a surrogate organization may be, equal care must be put into choosing which of its processes provide useful insights for intelligence and security informatics. Since we cannot look at all operations in the VE, we will explore the software requirements and release process at NetBeans [27]. This process is central to the overall mission of NetBeans, it is enacted on a frequent and ongoing basis, and it allows globally dispersed actors to work in a mostly autonomous manner on tasks that are coordinated and scheduled within the VE's central authority. This requirements and release process requires independent work spanning the enterprise to assess the feasibility of completing work on NetBeans software subsystems by the assigned date and addressing the importance of their inclusion in the prospective release and then later on, testing the subsystem to assure its correct functionality and operation. Based on this analysis, the central authority can then plan its development and release strategies based on the current status of each subsystem.

Thus, NetBeans is the VE being studied, and its requirements and release process is the focus for discovery and analysis.

5. Establishing a Meta-Model for Processes in OSSD Projects

Our process meta-model describes the interplay between agents, tools, resources required to complete a task, resources provided by its completion, and a description of the actions ("script") involved in completing the task [16]. A conceptual view of process object taxonomy appears in Figure 1. It suggests that a VE consists of processes that manipulate its resources. The minimum set of resource types associated with a process [23] includes its agents (actor roles), the tools they use, the action script they may perform/enact, and the information artifacts and other VE resources they consume or produce. Each process can also be hierarchically decomposed to arbitrary levels into component sub-processes, which are also resources. Furthermore, each process or sub-process consists of activities (not shown in the figure) that are partially ordered as a set of successor activities. The overall ordering of activities denotes the workflow, and the structure of the ordering denotes process control flow. So process or sub-process activities can be appear as a linear sequence, an iterative loop, a conditional selection among alternatives, or one or more concurrent branches. Figure 2 provides a view of process control taxonomy as coded and visualized in the Protégé-2000 knowledge editing environment. Further details for how this process meta-model is constructed in Protégé-2000 can be found elsewhere [10].

Though details of the process meta-model appear elsewhere [16, 23, 32], it should be noted that none of the constructs or formulation described above is specific to NetBeans, OSSD projects, or software development enterprises of any kind. Instead, they are generic and therefore applicable for modeling processes of any type for any kind of enterprise. In fact, the cited works provide studies of its use to support process modeling in domains like large-scale software development, government acquisition, and corporate financial operations.

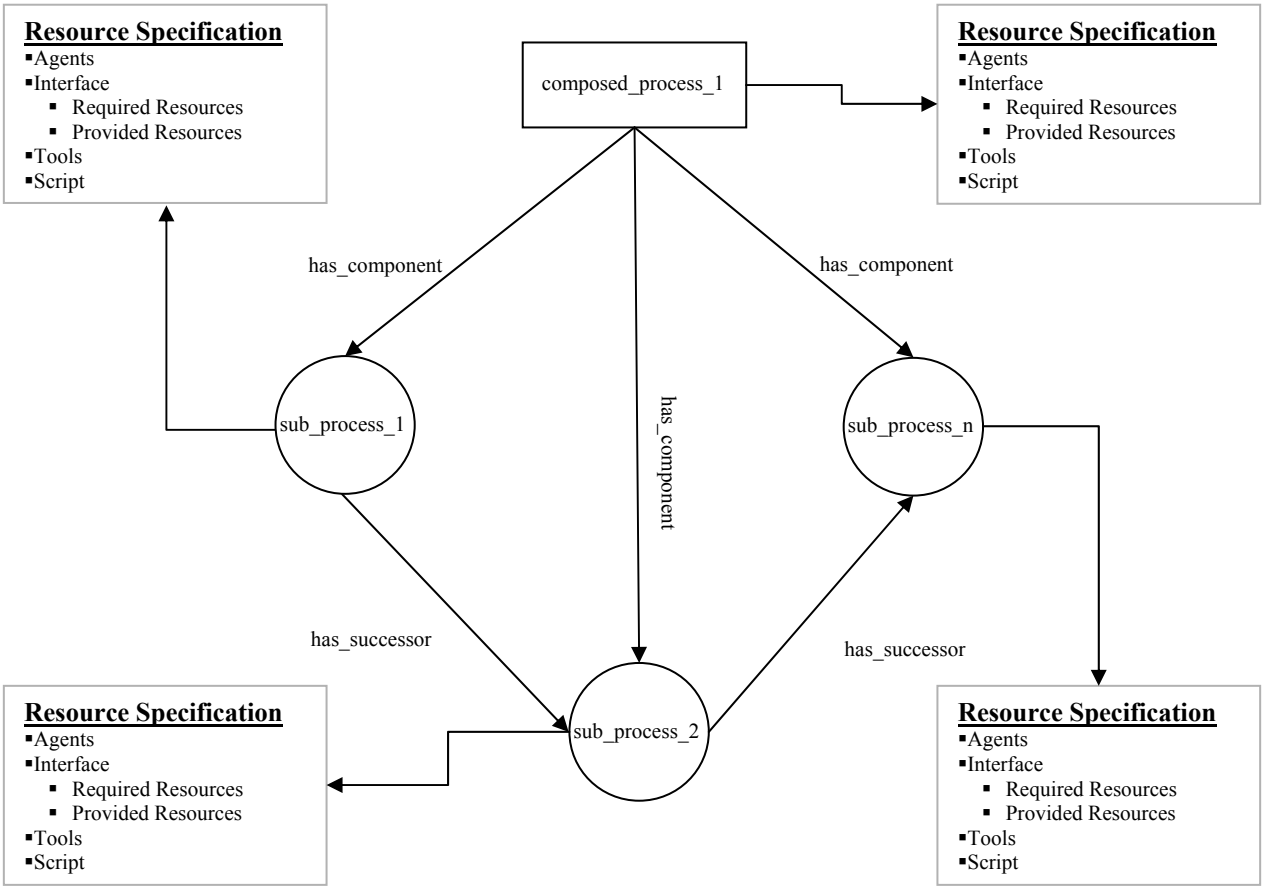

Figure 1. Conceptual view of a process object taxonomy.

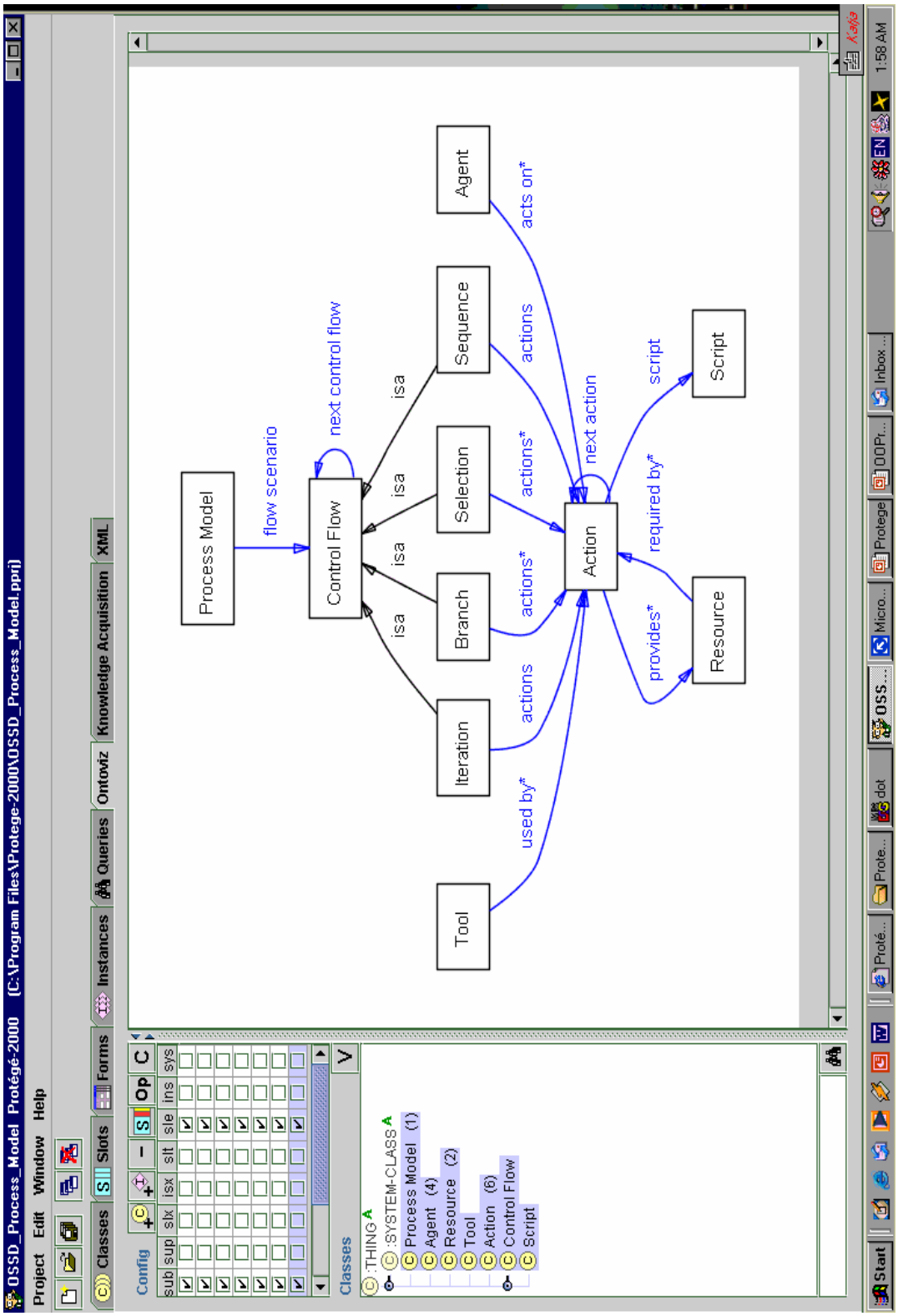

Figure 2. A representation view of a process meta-model created with Protégé-2000 [10].

We next turn to examine what types of information artifacts and resources can be found in support of the requirements and release process at NetBeans.

6. Information artifacts and resources found at NetBeans

Now that a meta-model has been established, we can begin to excavate the community's online information space. This step in the discovery process surveys the site to determine which types of information resources are available. With NetBeans, the project site map provided not only a breakdown of pages within each section, but also a timestamp of the latest update. This timestamp provides empirical evidence gathered from project content that reflects the process as it is currently enacted, rather than what it has evolved from. Such clues can be used to expand the meta-model. At this point, we have detailed which areas of the site contain plausible data, but we have still to address recognizing the data contained within. That is the goal of the reference model, which is simply a list of process clues associated with certain process activities. The "XTest" example below helps demonstrates this.

We have identified several types of information artifacts on the NetBeans Web site. Some of these are:

- Web pages, including project status reports and task assignments
- Asynchronous communications among project participants posted in threaded email discussion lists
- Transcripts of synchronous communication via Internet chat
- Software problem/bug and issue reports
- Testing scripts and results
- Community newsletters

- Web accessible software product source code directories
- Build binaries and distribution packages
- OSS development tools
- OSS development resources, including other software development artifacts

Accordingly, it is possible to construct a taxonomy of these artifacts types together with their attributes, such as when they first observed or last updated, location/URL, author, embedded hyperlinks, sub-components, other artifacts on the Web site hyperlinked to it, and so forth, as well as a hyperlink from the modeled object to an instance of the artifact or embedded content anchor.

Different participants in NetBeans have established preferred methods of interaction and communication with other participants, whether explicitly or implicitly. These collaboration modes yield a high amount of process evidence, as well as a large degree of unrelated data. This information can be found by examining three kinds of data: the structure of the Web site (its information architecture [30]); Web site content; and events that signal updates or transactions on the content or structure of the Web site, and when they occur.

First, the *structure* of the Web site can give us initial clues to the types of activities carried out by the community and the focus that is placed on them in the development lifecycle. A directory on the site named "XTest," for example, is a suggestive indicator that some kind of testing activity may be executed on the system, and the type of testing that is performed. If its contents included a set of files labeled "xtest_daily_20030614," the claim might be reinforced with some added

frequency of execution information. If the "XTest" directory were also located inside a "modules" directory, we might have reason to refute the claim, but may offer an alternate explanation. The "xtest_daily_20030614" file might be a source file updated daily of a system module named "XTest", or notes on its development progress. Structural data is also the easiest to obtain, since it may be discovered through a site map or with a simple Web crawler.

Second, examining Web site *content* creates a larger problem. Information artifacts, such as release announcements, bug reports, source code, and in some cases, explicit enumeration of process/task activity fragments, can be readily found. These provide us with a wealth of information. However, they can be misleading and incomplete. Site updates may delete critical information or relocate it. While larger sites are now turning to content versioning systems to manage the entire content of their information spaces, many still lack this or do not make it publicly available. Nevertheless, in larger and more active OSSD project communities like NetBeans, we may rely on a third type of data to validate content.

Third, whereas structure and content can tell us what types of activities are performed, by monitoring *site usage* in terms of site/artifact updates or transaction events, we may discern how often they are performed and which activities are more essential and which are peripheral to the project. This can be detected by monitoring the timestamps on the modifications of each page, message boards and issue reports. The meta-model can then be extended to include clues such as these. Thus, to discover a process, it is not only necessary to determine the artifacts that may contain process clues, but also what types of clues may be found in which artifacts and what forms they may take.

7. Constructing a semi-structured process specification

Whereas Cook and Wolf [5] approached process discovery using a Markov Model approach [cf. 2], but without domain knowledge, we apply *a priori* knowledge of software development processes [16] to help us discover, analyze, and model processes within the NetBeans VE. Instead of randomly probing or exhaustively searching the NetBeans information space, we construct an informal reference model detailing possible indicators that a given process activity, artifact update, or tool use has occurred. This model will be represented as a rich picture. So how is this model created?

We start with an effort to determine common operational scenarios and their contexts within the enterprise and process under study. Conveniently, a history of the NetBeans project was found by searching the "about" sub-section of the project web, which provided information on the NetBeans technologies (its products) under development, as well as the project and the nature of its status as a OSSD project [20]. This project history is a basis for understanding current development work practices in the NetBeans project. This history also details ways of becoming involved in development activities and the project community at large [18]. The modes of contribution can be used to construct an initial set of use cases for project participation. The site map also shows a Web page that describes the NetBeans project governance buried three layers deep within the site. This page exposes the primary member types, their roles and responsibilities which constitute additional use case information. Unlike those found through the modes of contribution, project roles span the breadth of the process, though at a higher level of abstraction.

Each use case encodes an activity scenario which we treat as a process fragment. In collecting use cases, we can extract concrete actions that can then be assembled into a process description to be modeled, simulated, and enacted. When aggregated, these use cases can be coalesced into an informal process model that can be visually represented as a rich picture [17] that can be represented as a semi-structured object model with embedded hyperlinks to the corresponding use cases. The rich hypermedia picture shown in 3 identifies developer roles, tools, and artifacts of development and their interaction. In the rich hypermedia process specification, agents and resources are written with plain text, while their corresponding actions are underlined hyperlinked phrases which, when selected, traverse the hyperlink and display the corresponding use case for that action as shown in Figure 4. Such a model can be especially useful for both those looking to comprehend how this VE operates, how to become involved in the enterprise or its process scenario, as well as those decision making authorities looking to make resource allocation decisions by offering an overview of the process and its context in the project, while abstracting away the detail of its activities.

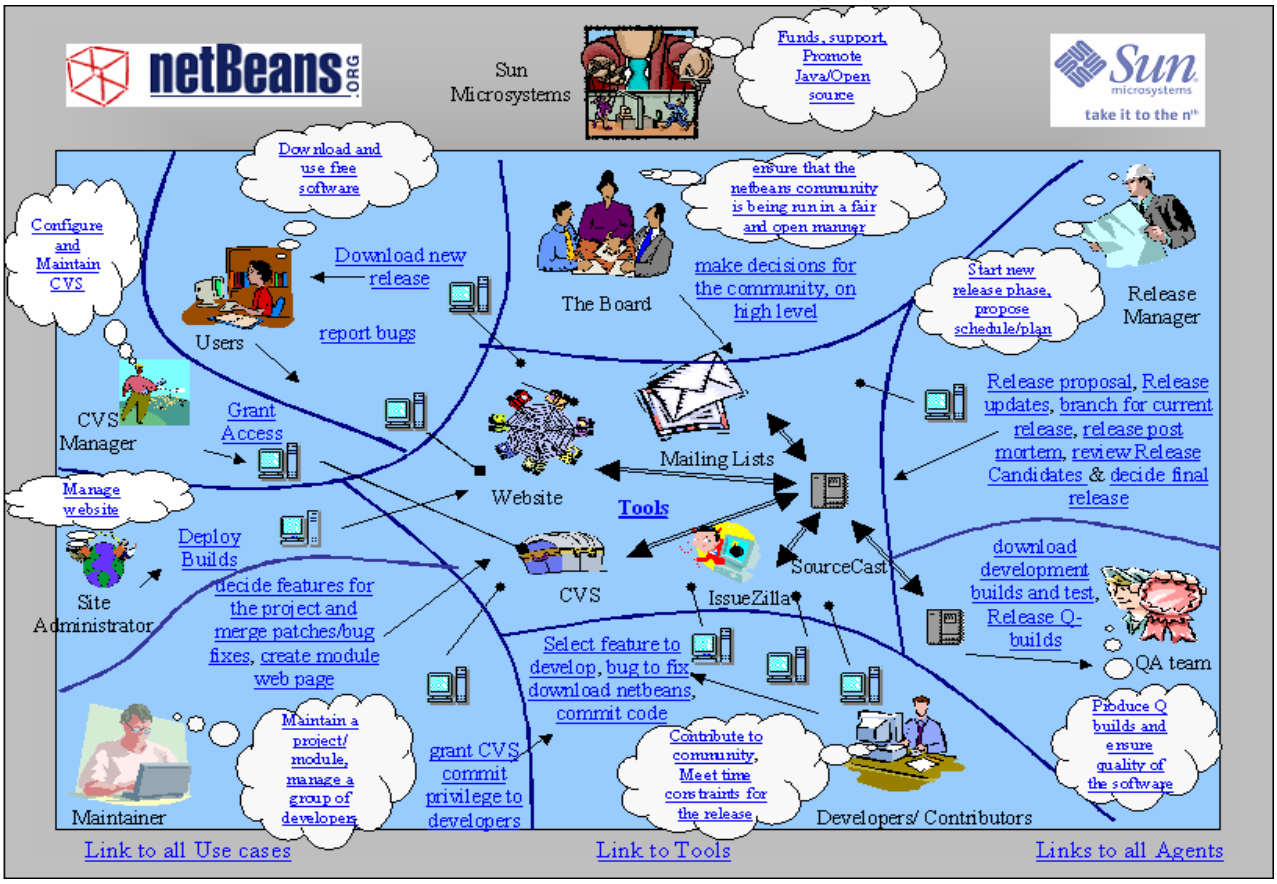

Figure 3. A rich hypermedia picture of the NetBeans context for their requirements and release process [27].

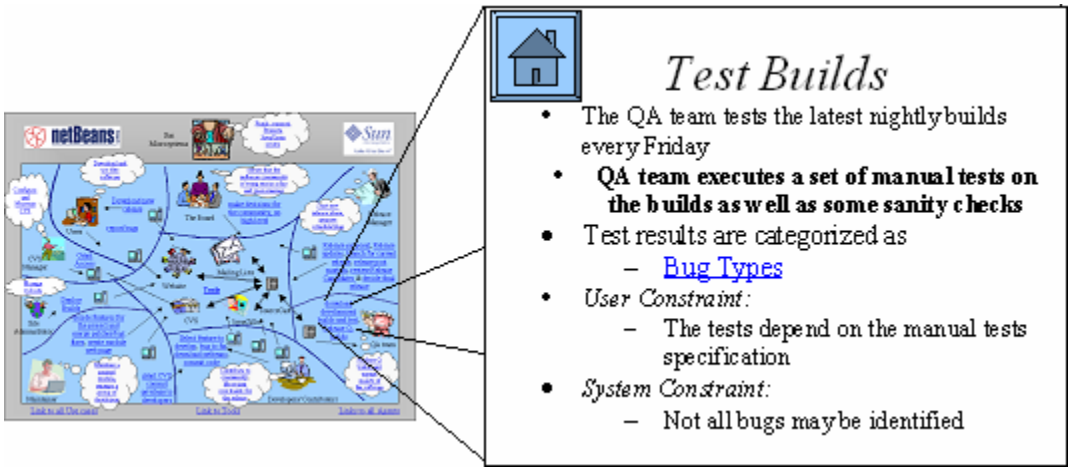

Figure 4. A hyperlink selection within a rich hypermedia presentation revealing details of a corresponding use case.

8. Formalizing the process model specification

The semi-structured model highlights the breadth of the process. Yet it does not provide the detail necessary required to perform analysis that may uncover potential for optimization, recovery from process breakdowns, process intervention, and other process engineering activities. If semi-structured and narrative models are devised to meet the needs of decision makers and process agents, formal models are needed to facilitate use by analytical tools. This requires a larger, finer grained set of process fragments than were needed in the semi-structured model.

In constructing the meta-model we noted contextual clues linking types of artifacts to the process data they might contain. These can be used to direct queries against the information space to draw out more subtle or hidden process data. By text-searching the NetBeans "defect" page [19], we can determine through the page location that quality assurance (QA) plays a role in the release process. We can enumerate some of the tools involved in the QA process (e.g. Bugzilla) given prior knowledge of the types of tools used in quality assurance, the information on the page, and the location of the page in the directory structure. We can tell that there is some computation of total defects, their priority levels, and component location. Going a step further, we notice that these defect totals are tallied daily from test results listed on another page [21], which is neither linked to the charts on the defect page, nor located under the QA module directory. If we consider another clue: that the link to the test results indicates that the tests are automated using XTest, we can construct a hypothetical sequence of events as follows. Every day, the build is tested against the XTest framework. Additional probing of the QA branch of the Web space turns up another set of tests: Q-Build verification. The Q-Build program is a hybrid

manual and automated testing process conducted by the SUN Microsystems QA team to perform sanity checks on periodic builds. The Q-Build Verification page documents these results and the priority in which defects should be addressed. The products of these actions are then posted on their respective Web pages. In this way, we have been able to infer a sequence of events given such evidence in bottom up fashion. Further, we have been able to associate tools that correspond to each event and their relationship as determined by their input and output resources. Casting our nets wider, we might query the entire information space with process related keywords using a search engine and attempt to construct relations between artifacts that may not be directly linked or for which there is an abundance of data.

The critical challenge in reconstructing process fragments from process enactment instances is in knowing whether or not the evidence at hand is related, unrelated, or anomalous. Reliability of associations constructed in this fashion may be strengthened by the frequency of association and, assuming the use of an unbiased search engine, the rank of artifacts carrying the association. In the above example, we can relate the "defects by priority" graph on the defect page to the defect priority results from the Q-Build verification. Likewise, the defect tallies and locations correlate to the error summaries in the XTest results. By looking at the curvature and dates listed on the defect graphs, we know which sets of results are charted and how often they are generated. This, in turn identifies the length of the defect chart generation process, and how often it is executed. The granularity of process discovered can be tuned by adjusting the search depth and the degree of inference to apply to the data gathered. An informal macro-view of the requirements and release process is shown in Figure 5.

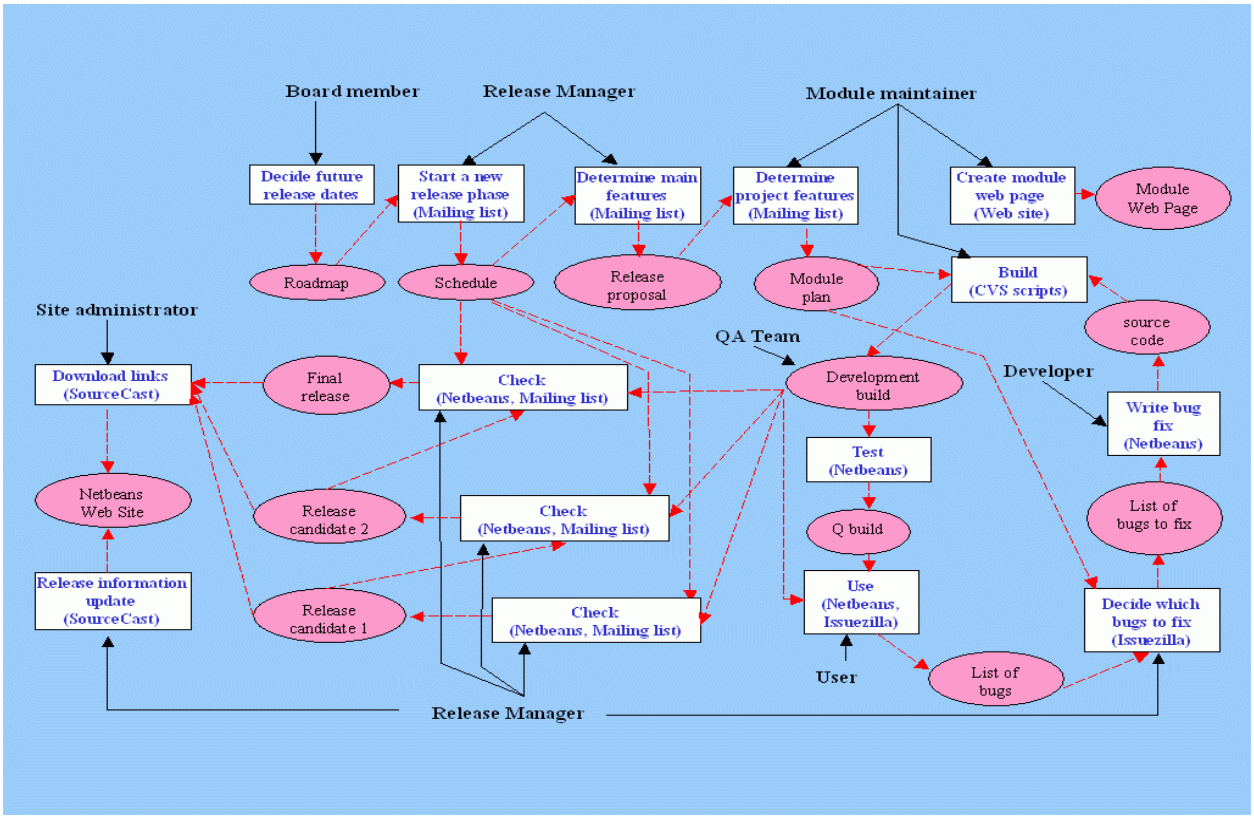

Figure 5. Process flow graph model of the NetBeans requirements and release process [27]

These process fragments can now be incrementally specified and composed into a formal PML description of the selected processes [23]. Using the PML grammar and process meta-model as a foundation [16], we created an ontology for process description with the Protégé-2000 knowledge-based ontology modeling tool [24]. The PML model builds from the use cases depicted in the rich hypermedia picture, as well as the hidden process data obtained through more in-depth mining of the information space, then distills them into a set of actions or sub-processes that comprise the process with its corresponding actor roles, tools, resources and the flow sequence in which they occur using the meta-model characterization operationalized in Protégé-2000 knowledge modeling and editing environment. A sample result of a process fragment coded in PML appears in Figure 6.

```
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 { }
   }
   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 { }
   }
   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 {
               Navigate to Issuezilla.
               Select issue number/component to update.
         }
   }
   action Update standing issue status {
         requires { Standing issue from Issuezilla, test results }
         provides { Updated Issuezilla issue repository }
         tool { Web browser }
         agent { users, developers, SUN ONE Studio QA team,
                 SUN ONE Studio developers }
         script { }
   }
   action Post bug stats {
         requires { Test results }
         provides { Bug status report, test result report }
         tool { Web editor, JFreeChart }
         agent { Release manager }
         script { }
   }
}
```

Figure 6. A PML description of the testing sequence for the NetBeans release process

Finally, using our Protégé-based PML modeling environment, we construct a formal representation of the modeled process with two external representations. The first is an XML description according to the ontology, while the second is a visual depiction generated by the

Ontoviz [24] an ontology graph visualization tool. The XML version serves as an exportable specification of the process that could be subsequently processed by other automated tools or Web services. The visualization, on the other hand, can be configured to incorporate relationships between process instance actions on varying depths and with different focal points (e.g., tools, roles, and resources), as suggested in Freeman [9] in his survey of methods for visualizing social or organizational networks.

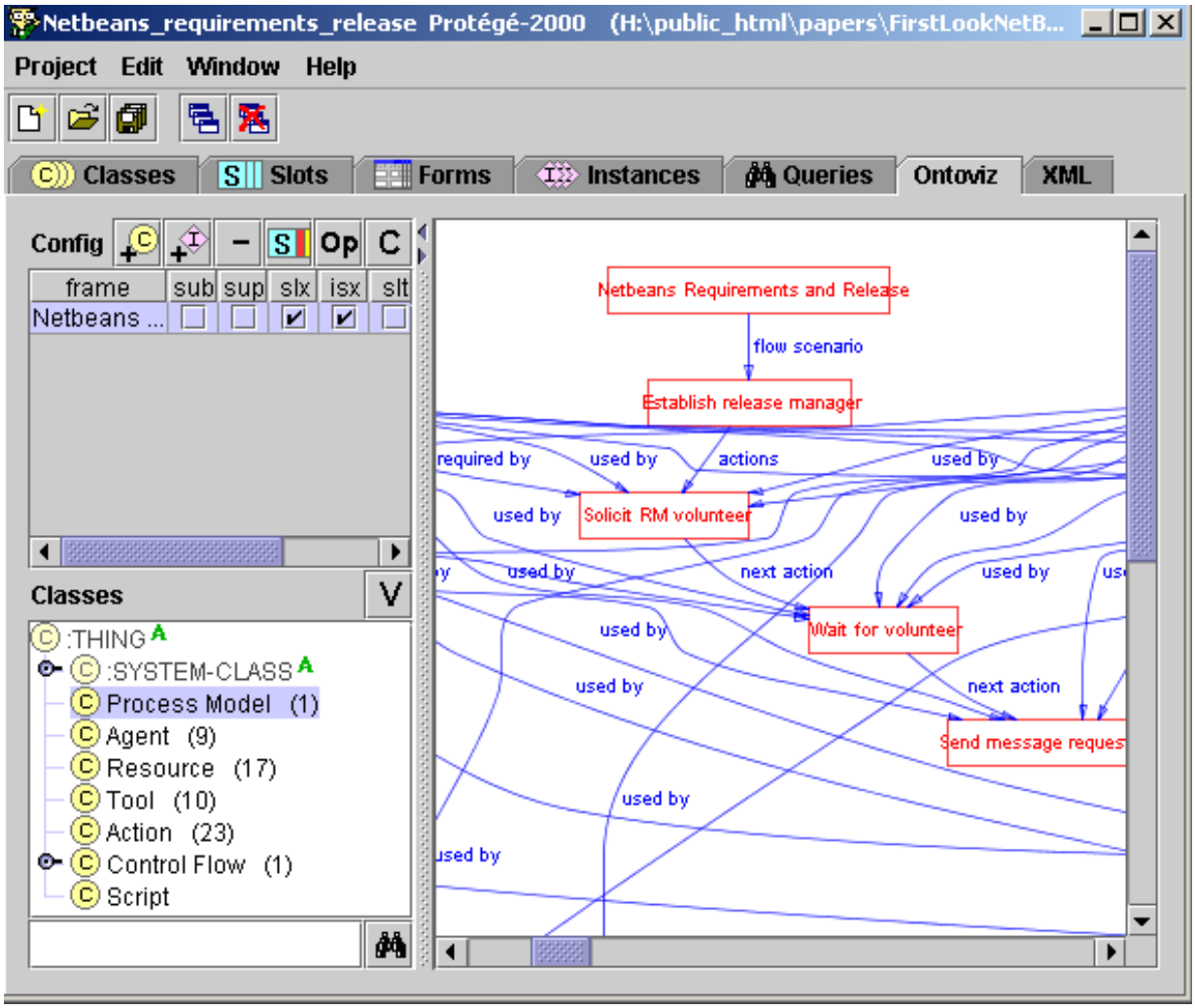

Figure 7. A partial view visualizing the process model for the requirements and release process at NetBeans [27].

The last activity in representing the process that has been discovered, analyzed, informally and formally modeled is packaging the results within an overall narrative description for future reference and secondary analysis. The process narrative describes a story of the process. An example of such a process narrative for the NetBeans requirements and release process is found elsewhere [27]. This report discusses the particulars of the meta-model, the finding of the semi-structured rich hypermedia, as well as the results from model formalizations and the process lifecycle activities thereby enabled. It describes and categorizes the tools, agents, and resources in the process. The goal of the process narrative is to provide a complete synthesis of the inputs and outputs of process discovery efforts in a form that can be more widely read, published, reviewed and digested. As such, it is more detailed than the rich hypermedia's use cases, but also less technical than the formal representation, but incorporates hyperlinks to both, as well as to further details within the Web site for the VE under study.

7. Discussion

As discussed in a companion report [14], this discovery approach can be more fully automated. We can take the results from our approach and build a base of informal and formal process assets that may be further deployed for analysis or reuse by other VEs. Such a repository can provide templates to enterprises as they form techniques to optimize, redesign, or continuously improve their processes [32], or to interdict or disrupt processes of organized threats. However, without an explicit understanding and representation of the process, it is difficult to know whether certain tasks may be removed from the critical workflow path, or where resource bottlenecks or points of interdiction may be located [33].

Finally, large VEs with central administrative authorities face issues with the intended or unintended transparency of their operational or decision making processes. Members who may be distant from the locus of authority often find it difficult to understand the changes in technical direction or organizational policy that may subsequently arise. This can cause delays or rejection of new policy directives, either due to communication breakdown or disenfranchisement, making policies difficult to enforce among loosely tied, semi-autonomous agents. On the other hand, transparency of sensitive processes can create a security risk. If processes used by organizations similar to those that desire to protect their confidentiality can be easily discovered, how feasible is process protection? So far, it appears that many VEs, such as large OSSD projects, are not aware of their processes, and lack formal or informal representations of these processes. Organizations may use this approach presented in this paper to assess their degree of process transparency, in a competitive intelligence fashion, and in doing so, assess the risk that such transparency carries with it. As easy as it is for a VE's own members to discover their online information work processes, it also begins to be possible for those they desire to protect processes from to do the same. Subsequently, this risk may be managed by regulating the transparency or obscurity of the VEs processes through policy decisions and activities that act to obscure or mis-direct the detailed analysis of these processes.

8. Conclusions

This paper introduces and describes a new approach to discovery and analysis of processes operating within geographically decentralized but logically centralized virtual enterprises. It is argued that this is an important problem in intelligence and security informatics, especially when it may not be possible to directly study and publish findings about the operational processes of different kinds of enterprises.

The research approach being investigated builds from existing results in Web data mining, process modeling, and process meta-modeling. It provides a case study using a surrogate VE to explain details of the methods involved, and to present representative results in the form of informal, semi-structured, and formal process models. Discovering, analyzing, and constructing these models from Web-based information artifacts, updates to them, other observable transaction events, access usage patterns (use cases), and emerging process fragments is shown to be a knowledge-intensive endeavor. Process meta-modeling and progressive model formalization are key capabilities to discovering process decomposition and control flow precedence relations within the context of the VE and its Web-based information artifacts, and how they evolve and change over time. The approach presented here seeks to outline this research problem and propose an initial solution that employs different knowledge capture, representation, and visualization techniques along the way. However, the solution at this time is partially automated, though additional automation for artifact content (text) data mining and probabilistic relational mining among artifact usage/update patterns appears as a promising next step. Nonetheless, the approach and results provide plausible evidence that further automation will be a productive investment that can yield more comprehensive process discovery and analysis, as well as a potential larger scale of deployment.

Overall, a successful approach to the discovery and analysis of processes operating within VEs will enable current approaches to mining Web site usage and textual (artifact content) data to expand to encompass process data mining as well, perhaps along the lines we suggest here.

Acknowledgements

The research described in this report is supported by grants from the National Science Foundation #ITR-0205679 and #ITR-0205724. No endorsement implied. Les Gasser at the University of Illinois, Urbana-Champaign and John Noll at Santa Clara University are also collaborators on the research project from which this article was derived.

References

[1] *Apache Software Foundation Web Site,* http://www.apache.org. (2003).

[2] I.V. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White, Visualization of Navigation Patterns on a Web Site Using Model Based Clustering. In *Proc. 2000 Knowledge Discovery and Data Mining Conference,* 280-284. (2000).

[3] H. Chen, M. Chau, and D. Zeng, CI Spider: a tool for competitive intelligence on the Web. *Decision Support Systems*. 34, 1-17, (Dec. 2002).

[4] A. Cockburn, *Writing Effective Use Cases*, Addison-Wesley: New York (2000).

[5] J. Cook and A.L. Wolf, Automating Process Discovery through Event-Data Analysis. In *Proc. 17th. International Conference on Software Engineering.* 373-386, (Seattle, Washington, USA, April, 1995).

[6] R. Cooley, B. Mobasher, and J. Srivastava, Web Mining: Information and Pattern Discovery on the World Wide Web. In *Proc. Ninth IEEE International Conference on Tools with Artificial Intelligence,* (Nov. 1997).

[7] *Debian GNU/Linux-- The Universal Operating System Web Page,* http://www.debian.org, (2003).

[8] M. Fowler, and K. Scott, *UML Distilled: A Brief Guide to the Standard Object Modeling Language*. Second ed. Addison Wesley: Reading, MA. (2000).

[9] L. Freeman, Visualizing Social Networks, *Journal of Social Structures*, 1(1), http://www.heinz.cmu.edu/project/INSNA/joss/vsn.html, (2000).

[10] J. Georgas, Software Process Modeling with Protégé. http://www.ics.uci.edu/~jgeorgas/ics225/example.htm, (June 2002).

[11] L. Getoor, N. Friedman, S. Dzeroski, and N. Lavrac, Learning Probabilistic Models. In *Relational Data Mining*. Springer-Verlag. (2001).

[12] J. Hong, J. Heer, S. Waterson, and J. Landay, WebQuilt: A proxy-based approach to remote web usability testing, *ACM Transactions on Information Systems*, 19(3), 263-285. (2001).

[13] S-Y. Hwang and W-S. Yang, On the discovery of process models from their instances, *Decision Support Systems*, 34(1), 41-57, (Dec. 2002).

[14] C. Jensen, and W. Scacchi, Simulating an Automated Approach to Discovery and Modeling of Open Source Software Development Processes. In *Proc. ProSim'03 Workshop on Software Process Simulation and Modeling,* (Portland, OR, May, 2003).

[15] F. Leymann, and D. Roller, Business processes in a Web services world, *IBM developerWorks*, http://www.ibm.com/developerworks/webservices/library/ws-bpelwp, (2002).

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

[17] A. Monk, and S. Howard, The Rich Picture: A Tool for Reasoning about Work Context. *Interactions,* 21-30, (Mar.-Apr. 1998).

[18] *NetBeans Contribute to The Community Web Page,* http://www.netbeans.org/about/community/contribute.htm, (2003).

[19] *NetBeans Defects in Graphics*, http://qa.netbeans.org/bugzzilla/graphs/summary.html, (2003).

[20] *NetBeans Open Source Project,* http://www.netbeans.org, (2003).

[21] *NetBeans XTest Results,* http://www.netbeans.org/download/xtest-results/index.html, (2003).

[22] J. Noll, and W. Scacchi, Supporting Software Development in Virtual Enterprises. *Journal of Digital Information.* 1(4), (Feb. 1999).

[23] J. Noll, and W. Scacchi, Specifying Process Oriented Hypertext for Organizational Computing. *Journal of Network and Computer Applications.* 24(1), 39-61. (2001).

[24] N.F. Noy, M. Sintek, S. Decker, M. Crubézy, R.W. Fergerson, and M.A. Musen, Creating Semantic Web Contents with Protégé-2000, *IEEE Intelligent Systems,* 16(2), 60-71. (2001).

[25] S. O'Mahony, Non-Profit Foundations and their Role in Community-Firm Software Collaboration. In *Proc. HBS-MIT Sloan Free/Open Source Software Conference: New Models of Software Development.* (Boston and Cambridge, Jun. 2003).

[26] *Ontoviz Web Site*, http://protege.stanford.edu/plugins/ontoviz/ontoviz.html, (2003).

[27] M. Oza, E. Nistor, S. Hu, C. Jensen, and W. Scacchi, *A First Look at the NetBeans Requirements and Release Process*. http://www.ics.uci.edu/~cjensen/papers/FirstLookNetBeans/, (June 2002).

[28] M. Petras, *NetBeans Weekly News.* (84), http://www.netbeans.org/community/news/newsletter/2002-09-23.html, (Sept. 23, 2002).

[29] *Protege-2000,* http://protege.stanford.edu, (2003).

[30] L. Rosenfield, and P. Morville, *Information Architecture for the World Wide Web (2nd Edition*), O'Reilly Press, Sebastopol, CA. (2002).

[31] RosettaNet, http://www.rosettanet.org, (2003).

[32] W. Scacchi, Modeling, Integrating, and Enacting Complex Organizational Processes, in K. Carley, L. Gasser, and M. Prietula (eds.), *Simulating Organizations: Computational Models of Institutions and Groups*, 153-168, MIT Press, Cambridge, MA. (1998).

[33] W. Scacchi, Redesigning Contracted Service Procurement for Internet-based Electronic Commerce: A Case Study. *Information Technology and Management*, 2(3), 313-334. (2001).

[34] W. Scacchi, Understanding the Requirements for Developing Open Source Software Systems, *IEE Proceedings – Software*, 149(1), 24-39, (Feb. 2002).

[35] J. Schroeder, J. Xu, and H. Chen, CrimeLink Explorer: Using Domain Knowledge to Facilitate Automated Crime Association Analysis. In *Proc. of the First NSF/NIJ Symposium on Intelligence and Security Informatics (ISI 2003)*, 168-180, Lecture Notes in Computer Science, Springer-Verlag. (2003).

[36] E. Stohr, and J.L. Zhao, Workflow Automation: Overview and Research Issues, *Information Systems Frontiers*, 3(3), 281-296. (2001).

[37] A.L. Wolf, and D.S. Rosenblum, A Study in Software Process Data Capture and Analysis. In *Proc. Second International Conference on the Software Process,* 115-124, IEEE Computer Society. (Feb. 1993).

[38] J.L. Zhao, H. Bi, and H. Chen, Collaborative Workflow Management for Interagency Crime Analysis. In H. Chen, *et al.* (eds.), *Intelligence and Security Informatics*, Lecture Notes in Computer Science, 2665, 266-280, Springer-Verlag. (2003).