INCIDENT AND PROBLEM MANAGEMENT USING A SEMANTIC WIKI-ENABLED ITSM PLATFORM

Frank Kleiner[1], Andreas Abecker[2] and Marco Mauritczat[3]
[1]*FZI Forschungszentrum Informatik, Haid-und-Neu-Str. 10-14, Karlsruhe, Germany*
[2]*disy Informationssysteme GmbH, Erbprinzenstr. 4-12, Karlsruhe, Germany*
[3]*Infomotion GmbH, Ludwigstr. 33-37, Frankfurt am Main, Germany*

Keywords: IT service management, Semantic Wiki, Ontology-based information systems, Collaborative ITSM.

Abstract: IT Service Management (ITSM) is concerned with providing IT services to customers. In order to improve the provision of services, ITSM frameworks (e.g., ITIL) mandate the storage of all IT-relevant information in a central Configuration Management System (CMS). This paper describes our Semantic Incident and Problem Analyzer, which builds on a Semantic Wiki-based Configuration Management System. The Semantic Incident and Problem Analyzer assists IT-support personnel in tracking down the causes of incidents and problems in complex IT landscapes. It covers two use cases: (1) by analyzing the similarities between two or more system configurations with problems, it suggests possible locations of the problem; (2) by analyzing changes over time of a component with a problem, possible configuration changes are reported which might have led to the problem.

1 INTRODUCTION

The increasing complexity of IT landscapes, paired with an increasing dependency on IT services from almost all functions within an organization has led to new paradigms for managing information technology. While for a long time, the technical aspects of IT were the center of attention, the **IT Service Management (ITSM)** approach centers around services, while putting technical aspects into the background. Focusing on the customer helps to align services provided by the IT department with an organization's business goals (Addy, 2007; Clacy and Jennings, 2007). There exist a number of frameworks which give guidelines for the implementation of ITSM-relevant functions and processes within organizations. Currently, the **IT Infrastructure Library (ITIL)** (Caetlidge et al., 2008; Cannon and Wheeldon, 2007) is the most widely used general-purpose IT Service Management framework. ITIL consists of five volumes, which describe the lifecycle of IT services. The topics addressed in this paper make use of and build on the following ITIL processes:

- **Service Asset and Configuration Management (SACM)** deals with maintaining a system for managing all entities used for providing IT services, including their relations to and dependencies from each other. Entities are referred to as Configuration Items (CIs) and are stored in the Configuration Management System (CMS). Information about CIs is stored in one or more Configuration Management Database(s) (CMDB), which are part of the Configuration Management System (Lacy and Macfarlane, 2007). A simple example for CMDB entries goes as follows: a service which is responsible for providing an organization's Web site is provided by a Web Content Management System running on an instance of the Apache Web server on a certain computer. It uses a MySQL database instance, which runs on another computer. In order to communicate, the computers are networked together by using a network switch. The network switch is connected to a router, which provides Internet access. As can be seen, in order for the Web site to be available (customer perspective), a number of services and systems have to be running and reachable via the network (technical perspective).
- **Change Management** is concerned with the application of changes to an IT infrastructure (e.g., hardware, software, services) in a controlled manner. The goal is to assess the potentials for problems with planned changes and to mitigate the

Kleiner F., Abecker A. and Mauritczat M..
INCIDENT AND PROBLEM MANAGEMENT USING A SEMANTIC WIKI-ENABLED ITSM PLATFORM.
DOI: 10.5220/0003751303630372
In *Proceedings of the 4th International Conference on Agents and Artificial Intelligence* (ICAART-2012), pages 363-372
ISBN: 978-989-8425-95-9

associated risks. Change Management depends heavily on Service Asset and Configuration Management for understanding dependencies between IT components when planning changes (Lacy and Macfarlane, 2007). Resuming the example, this means that before upgrading the MySQL database to a new version, it has to be made sure that it does not break the database access of the Web Content Management System before performing the upgrade.

- **Incident Management** is the process within the ITIL framework which is responsible for fixing the causes of service interruptions. An incident in ITIL is defined as an "unplanned interruption to an IT service or reduction in the quality of an IT service" (Cannon and Wheeldon, 2007). The focus of the Incident Management process is clearly on restoring the service as soon as possible, without necessarily being concerned with the underlying cause of the incident (Cannon and Wheeldon, 2007). For example, if the Web site stops working, a solution within Incident Management would be to restart the Web server or to rebuild a broken database.
- **Problem Management** is concerned with finding the underlying causes of service failures. Even if a service was successfully restored within the Incident Management process, it has to be made sure that the error which caused the outage will not reoccur in the future (Cannon and Wheeldon, 2007). For example, the cause of the database crash could be a relatively rare error in the underlying hardware, which has to be detected and which leads to replacing the problematic hardware component.

In (Kleiner and Abecker, 2009; Kleiner et al., 2009a), we motivated and described a collaborative, semantics-enabled ITSM software support based on the Semantic MediaWiki. In the works published so far, we focussed on the Service Asset and Configuration Management process above. The value-added of that ITSM software infrastructure mainly comes from the use of a Semantic Wiki as a "single point of information" that collects and integrates manifold kinds of information and knowledge about an organization's hardware and software environment. In this paper, we aim at drawing further benefit from the collected information in order to support additional ITSM processes, namely Incident Management and Problem Management; by examining the stored knowledge about configuration items, their structure and relationships, we want to analyse the causes for occurring incidents or documented fault situations. If such an analysis tool works well, the created new knowledge about problem causes may also be exploited for Change Management. In this paper, we present a prototypical tool with the desired functionality, called Semantic Incident and Problem Analyzer (SemPA).

This paper is organized as follows: first, a detailed problem description is given in section 2, followed by a description of our design in section 3. The implementation of the Semantic Incident and Problem Analyzer is described in section 4. In section 5, a conclusion is given, followed by a discussion of related work and an outlook on future work.

2 PROBLEM DESCRIPTION

2.1 Application Environment

The solution presented in this paper addresses a number of real-life problems encountered in the daily operations within the IT department of an SME organization with about 150 full-time employees as well as 250 part-time employees. Because the organization's core business is mainly IT-centric, the knowledge of the majority of its employees can be rated high or very high. The IT department, which consists of four full-time and eight part-time employees, provides IT services mostly for in-house customers. Services include the design and maintenance of the network infrastructure (ethernet, wireless networks, telephone, VoIP), email services, Web services, and database services. Furthermore, it includes the coverage of IT equipment through its life-cycle, e.g., the acquisition, testing, commissioning, maintenance and decommissioning of servers, desktop computers, and notebooks. While key infrastructure components and services are maintained by the IT department, employees are free to install software required for their work on their workstations, as well as to set up services for testing purposes within the internal network.

2.2 Incident Classes

After analyzing the incidents reported to our IT department's help desk system and their underlying causes, it was observed that there are three main classes of incidents:

- **Class 1: Multiple incidents with a common cause.** Two or more incidents occur, which are related to each other by a common cause. An example for this kind of incident is the failure of a network switch, which leads to a number of users having network problems.

- **Class 2: Single incident evolved over time.** An incident occurs, where a single computer evolves a problem which can be traced back to a change which was applied to the computer by the user or the IT department. An example is an upgrade to a Web browser, which crashes when visiting a certain Web page, which loaded perfectly fine before the upgrade.
- **Class 3: Stand-alone incident.** An incident occurs on a single system without a previous change to any component. Examples are mostly found when looking at hardware failures (e.g., a failed harddisk or main-board).

Because incidents generally have an underlying problem, which has to be found and fixed in order to prevent an error from reoccurring, tracking down the cause of an incident means finding the incident's underlying problem. Depending on the class of the incident, different techniques have to be applied and different knowledge of the support personnel is required. In order to find the cause of a class 1 incident, a detailed knowledge of similarities between IT components has to be possessed by the person assigned to find the problem. If this is not the case, information has to be gathered from the CMDB, which can be cumbersome if done manually. For class 2 incidents, in some cases the user of a system can give valuable hints by narrowing down the time interval when the undesired behavior occurred for the first time. Class 3 incidents are usually relatively easy to detect, because they are limited to a single system.

In addition to the different classes, incidents with the same class of underlying problem can occur on different systems independently from each other and distributed over time. This means that in order to speed up locating problems, it is necessary to document fixed problems and make them searchable for further use.

2.3 Requirements Analysis

After studying the different classes of incidents, the requirements for a tool which helps in tracking down the underlying problems were formulated. Because class 3 incidents are on the one hand restricted to a single system, which makes locating them relatively easy, and on the other hand not detectable via comparing CIs, class 3 incidents are not pursued further and partly left to mechanisms used for detecting failing equipment.

The requirements for the Semantic Incident and Problem Analyzer are as follows:

- Ability to find the cause of class 1 incidents by comparing a given list of IT components for similarities. E.g., to detect a failing network switch from incidents reported by independent users indicating a problem.
- Ability to find the cause of class 2 incidents by comparing configurations in time. For example, to detect the cause of an incident report which states, that a program was running fine two days ago, was not used yesterday and does not start today.
- Ability to find problems which were fixed on the same or other computers in the past, e.g., a browser update caused problems with a browser plug-in, which happened again on another computer, with another browser version and another plug-in.

Before presenting the implementation of the Semantic Incident and Problem Analyzer, an overview of the Semantic Wiki-based IT Service Management platform and of the ontology of the Semantic Incident and Problem Analyzer domain, is given.

3 DESIGN

3.1 Wikis and Semantic Wikis

Wikis are Web sites which enable visitors to contribute to their content by editing the content from within their Web browsers. The Wikipedia[1] encyclopedia is a shining example for the possibilities of Wikis. In the corporate context, Wikis are often used for knowledge management for projects, as portals, and as tools for project management. More information about Wikis can be found in (Barrett, 2008; Ebersbach et al., 2007).

Semantic Wikis extend Wikis by adding features to express semantic statements. These explicit semantic statements allow a better processing of Wiki articles and their relations by a computer. In (Schaffert et al., 2008), an overview of characteristics of Semantic Wikis as well as some examples for implementations are given. The main characteristics of a Semantic Wiki are (Schaffert et al., 2008):

- Semantic Wikis extend non-semantic Wikis by adding means to express structured data, while keeping the Wiki's flexibility and collaborative working style. In order to achieve this goal, "meta-data in the form of semantic annotations of the Wiki pages themselves and of the link relations between Wiki pages" (Schaffert et al., 2008) is supported.

[1] http://www.wikipedia.org/

- An ontology (Staab and Studer, 2009) forms the underlying knowledge model of the Wiki. Added and changed annotations in the Semantic Wiki are reflected in the ontology. Usually, Ontologies used by Semantic Wikis are represented in RDF Schema or OWL (Allemang and Hendler, 2008), which simplifies the exchange of data with external applications.
- Additional implicit information can be derived from information present in the Wiki by using deductive reasoning (Krötzsch et al., 2007).
- Semantic Wikis provide simple means for annotating links, articles and other content.
- Attributes and relations can be used in queries which extend the possibilities of queries from simple keyword searches to more complex ones.

There is a number of different implementations of Semantic Wikis[2]. The work presented in this paper builds on top of the Semantic MediaWiki software (Krötzsch et al., 2007; Vrandečić and Krötzsch, 2009). Semantic MediaWiki is an extension for the popular MediaWiki software which is the technical basis for Wikipedia and numerous other Wiki sites. MediaWiki and Semantic MediaWiki were selected because of their reliability, extensibility, and the quality of their documentation.

3.2 Semantic Wiki-based IT Service Management Platform

When looking at the IT landscape of organizations, it can be seen that there exists a multitude of different hardware components (e.g., desktop computers, servers, notebooks, network switches, routers, printers), software components (e.g., operating systems, Web server software, database server software) and services (e.g., an organization's Web site, email service). In addition, there exist dependencies between these CIs, e.g., an instance of an operating system is running on a certain physical server, which is connected to a certain network switch and is providing a number of specific services. As shown in (Kleiner and Abecker, 2009; Kleiner et al., 2009b), a Semantic Wiki can be used as a Configuration Management System for managing information about CIs as well as the relations between the CIs. In our scenario, each CI is described in a Wiki article which means that each computer and other hardware component has an associated Wiki page which lists its properties. In the case of a computer, properties range from its name to its serial number. Relations to other CIs are expressed as relations between Wiki articles, e.g., the manufacturer of a computer has its own Wiki page, which is linked to by all relevant CIs and which includes information about the manufacturer relevant for delivering IT services. In order to simplify the editing of structured information, the Semantic Forms[3] extension is used, which provides a forms-based interface which abstracts from the underlying semantic relations.

3.2.1 Architecture

Figure 1 gives an overview of the architecture of the Semantic Wiki-based IT Service Management platform. The core of the system is the Semantic Wiki-based Configuration Management System, where all information relevant for providing IT services is stored. This ranges from formal statements about IT components in the form of attributes and relations (e.g., a computer has a certain kind of graphics adapter and is connected to a network switch) to free text used to describe, for example, work processes and best practices (Kleiner and Abecker, 2009). In order to provide additional functionalities, extensions can be added to the IT Service Management platform. Figure 1 shows three extensions which are used to interact with hardware and software components that are part of our organization's IT landscape. Furthermore, the Semantic Incident and Problem Analyzer is shown, which is described in detail in this paper.

3.2.2 Configuration Gathering

The first extension is used for Configuration Gathering, i.e., the automatic acquisition of information from computers and other IT components over the network. At the current implementation stage, information can be read from Windows computers via the Windows Management Instrumentation (WMI) (Jones, 2007) infrastructure. This enables to automatically read information about hardware components of a computer (e.g., its graphics adapter, CPU, RAM, harddisk, network adapter), installed software, and configurations (e.g., the computer's network addresses). Another supported protocol is the Simple Network Management Protocol (SNMP) (Case et al., 1990) which is used to communicate with network hardware (e.g., network switches) and other components (e.g., printers) (Kleiner et al., 2009b).

3.2.3 Intrusion Detection

A part of ensuring the security of an organization's computer networks is the detection of suspicious ac-

[2] http://semanticweb.org/wiki/Semantic_wiki_projects

[3] http://www.mediawiki.org/wiki/Extension:Semantic_Forms

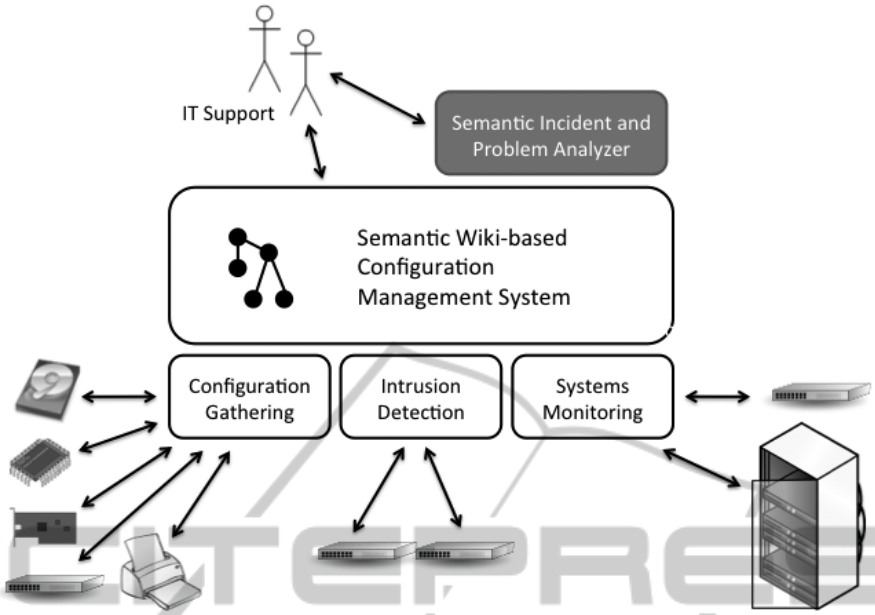

Figure 1: Overview of the Semantic Wiki-based IT Service Management Platform.

tivities by using a network intrusion detection system. These systems scan network traffic for patterns which indicate malicious activities (Northcutt and Novak, 2002). The Intrusion Detection extension interacts with the Open Source network intrusion detection tool Snort (Roesch, 1999) in order to integrate intrusion detection events into the IT Service Management Wiki. By using the knowledge stored in the Semantic Wiki-based Configuration Management System, events from the intrusion detection system can be better classified as relevant or not relevant than without this knowledge. Rules are used to sort out events which do not represent a threat to the attacked systems. At the moment, the intrusion detection extension is under development, an evaluation of the extension will be performed within the next six months.

3.2.4 Systems Monitoring

Another aspect of ensuring the delivery of services is the monitoring of service availability. The Systems Monitoring extension interacts with Nagios (Barth, 2005), an Open Source systems monitoring tool. Nagios tests services for availability by sending requests and analyzing the answers. Our systems monitoring extension simplifies the administration of Nagios by creating Nagios configuration files from information stored in the Semantic Wiki-based Configuration Management System. Furthermore, information about failed services is presented within the Wiki interface (Kleiner et al., 2009a).

3.2.5 Semantic Incident and Problem Analyzer

The Semantic Incident and Problem Analyzer, which is presented in detail in this paper, uses information about CIs which is stored in the Semantic Wiki-based Configuration Management System. IT support personnel is able to query the Semantic Incident and Problem Analyzer when suspecting class 1 or class 2 incidents. The Semantic Incident and Problem Analyzer is embedded into the Wiki interface, which means that support personnel uses the same interface as when documenting changes or looking up information about CIs.

3.3 Ontology

This paragraph describes the structure of the ontology, which was developed as the data model for the Semantic Incident and Problem Analyzer. In the text, class names are printed in bold, while relations are printed in italic. One of the main classes of the ontology is **Computer System**. It represents any computer in operation in an organization, either physical or virtual. Subclasses are **Desktop Computer**, **Notebook Computer**, **Server Computer**, and **Virtual Computer**. Computer systems are composed of hardware components, which are modeled in the **Hardware Component** class. The relation *has Hardware* is used for stating which hardware components comprise which computer system. Examples for hardware components are **CPU**, **Graphics Card**, **Main Board**, **RAM**, **Harddisk**, and **Network Adapter**. The **Net-**

work Adapter class is part of the domain of the *connected to Network Component* property, which has the class **Network Equipment** as its range. Another part of the domain of the property *connected to Network Component* is the class **Network Equipment** itself, which enables the modeling of network equipment being connected to other network equipment, e.g., network switches connected to each other or a router. Subclasses of **Network Equipment** are **Firewall**, **Network Switch**, **Router**, and **Wireless Access Point**. Computer systems and network equipment are located at a certain location, which is expressed by the *located in* relation. Locations are represented by the **Location** class, which has **Building**, **Room**, and **Server Rack** as subclasses. Server racks are located in rooms, which is expressed by *located in Room*, while rooms are located in buildings, which is expressed by the relation *located in Building*. Software is modeled in the **Software** class, with **Application Software** and **Operating System** as subclasses. The relation *has installed Software* between the **Computer System** and the **Application Software** class indicates, which software is installed on a computer. The property *has Operating System* has **Computer System** as domain and the class **Operating System** as its range. The relation was introduced in addition to the *has installed Software* property to address the special status of operating systems on computers. Services are modeled in the **Service** class. Computer systems provide services, which is expressed by the use of the *provides Service* relation. Computer systems and services have one or more owners, which is stated by the *has Owner* relation. The **Standard** class is used to model all kinds of standards, e.g., RAM standards, or network standards. The associated relation is *has Standard*, with the classes **Hardware Component**, **Service**, **Network Equipment**, **Computer System**, and **Software** as domain. The manufacturer of hardware and software is expressed in the class **Manufacturer** and the relation *has Manufacturer*. Incidents and problems are modeled in the classes **Incident** and **Problem**. Incidents can be related to other incidents, or to problems, which is expressed by the *related to Incident* relation. Accordingly, problems can be related to other problems or incidents. Problems and their solutions are modeled with the *has Solution* relation, with the class **Problem** as domain, and the class **Solution** as range.

4 IMPLEMENTATION

The implementation of the Semantic Incident and Problem Analyzer is built on top of Semantic MediaWiki's extension architecture. A Special Page extension, realized in PHP provides the foundation for the process of finding similarities between configuration items. Two different *operation modes* have been implemented to find problems of configuration items:

- The *first idea* relies on the assumption that once two or more different Configuration Items show an identical problem, the cause is possibly identical as well. A comparison of equalities of all problem-affected Configuration Items will lead to a set of properties which all affected Configuration Items have in common. While more than one Configuration Item is needed for this operation mode of the Semantic Incident and Problem Analyzer to work, it is of special interest for Problem Management.
- The *second approach* to find possible causes for a specific problem can be used especially in cases, where only a single Configuration Item shows a certain problem. By comparing a single Configuration Item's properly working configuration with the configuration after a problem emerged, changes which may have caused the malfunction should become obvious. This procedure is of particular interest for Incident Management, as a single malfunctioning Configuration Item is sufficient to use this structured approach for tracking down the incident's cause.

Independent of which approach is used, the set of identified properties can be used as a structured entry point for further manual troubleshooting. The *implementation* of these two ideas is mainly identical:

1. Semantic relations and individuals are recursively retrieved from the Semantic MediaWiki in order to build a tree-like data structure of every originating CI and all succeeding CIs. Whether a CI succeeds another CI is defined through semantic properties inside the Semantic MediaWiki. The nodes of such a tree structure hold the name of an individual and the class it is an instance of. The connecting edges between the nodes represent a property between two individuals, thus forming a set of semantic triples between a node and all succeeding nodes, as provided by the Semantic MediaWiki's annotated links. Care was taken to detect cyclic relations between individuals, in order to prevent the generation of infinite trees.
2. Each tree structure is successively compared with an initially empty compare tree in an depth-first search manner. For each node it is checked whether the triple it forms together with its succeeding node and the property-edge is already present at the actual level in the compare tree. If

the triple is found, the compare tree increases a counter for the object node of that triple, in order to represent a match with a previously added node. In case the triple is not yet present in the compare tree, the algorithm clones the triple and adds it to the compare tree. At this point it is inspected if the preceding node already has a relation to an individual of the same class with the same property like the newly added triple. In Figure 2 this is the case when adding the node marked with "B", because node "A" was added previously. This step is necessary to detect similarities between the trees at a deeper level. Even though the recently added node (B) might not be equal to any already present nodes at that level, it is possible that another individual of the same class has a similar subtree (A). Therefore the subtree of the recently added node (B) would be compared to all similar node's subtrees as well (A). An example for similarities in subtrees are two graphics cards, which have the same chip vendor, but use a different chip set and are manufactured by different graphics card manufacturers. While in the presented example, the tree depth is two, trees with a greater depth occur in more complex scenarios. For example, when comparing hosts in a network, including connections between network switches and routers, trees depths of seven were found in the test environment. In more complex environments, in some cases it might make sense to limit the tree depth in order to limit the processing time when using the analyzer.

3. The compare tree contains the union of all triples of all trees which have been compared to each other. The occurrence of correspondences between triples can easily be identified through the size of the matches counter. During the final third step, the compare tree is visualized using the MediaWiki extension GraphViz[4]. DOT[5] code is generated and gets passed to GraphViz which returns an appropriate bitmap file containing the tree visualization. It can be parameterized which nodes should be displayed according to their matches counter. After a comparison of n different configuration items, first of all one would be interested in nodes which have a matches counter of n-1, because these nodes were part of all n trees. Thus it can easily be achieved to hide potentially uninteresting nodes and improve the readability of the graphical tree representation as shown in Figure 2. The more matches a node has, the stronger it gets colorized, dependent on the selected color scheme. For improved usability, nodes can be clicked to open their Wiki page.

Figure 2 shows the screenshot of a comparison between two different configuration items. The visualized tree indicates through the matches counter inside the node *MDT 1024* equaling 1 (and all its succeeding nodes), that the two previously compared configuration items are both connected with this configuration item through the same property. The two nodes names *Asus* with the matches counter set to 1 both bear the information, that both configuration items have the property *has hardware* connecting them with an instance of *Graphics card*. While the instances of *Graphics card* are different from each other, each instance in return has an semantic relation to the same instance of the class *Manufacturer*, namely *Asus*. One could therefore read the matches counter equaling 1 in the *Asus* node as "The compared configuration items have a mutually different graphics card of the same manufacturer." In order not to loose the information about the actual instances of the class *Graphics card*, both unique instances *Asus ATI Radeon HD 2400 XT* and *Asus Pro Gamer* are present in the tree.

In case a single configuration item was compared with itself at a different point of time, the visualization needs to emphasize those nodes with zero matches. While the properties of the configuration which did not change will have exactly one match, zero matches indicate that either before or after the problem occurred this property of the configuration was different.

5 CONCLUDING REMARKS

5.1 Summary and Related Work

This paper presented our Semantic Incident and Problem Analyzer which assists IT-support personnel in tracking down problems in complex IT landscapes. Built on top of Semantic MediaWiki, it makes use of the information about Configuration Items stored in the Wiki. The application scope of the Semantic Incident and Problem Analyzer can be divided into two main scenarios: first, problems which occur at roughly the same time period on different hardware components and can be narrowed down to a common cause; second, problems which occur on the same hardware component and can be tracked down by comparing the configurations of the component at different times. We have designed an ontology for problem analysis, which will be extended towards a comprehensive ontology for IT landscapes and IT Service Management support.

[4]http://www.mediawiki.org/wiki/Extension:GraphViz
[5]http://www.graphviz.org/doc/info/lang.html

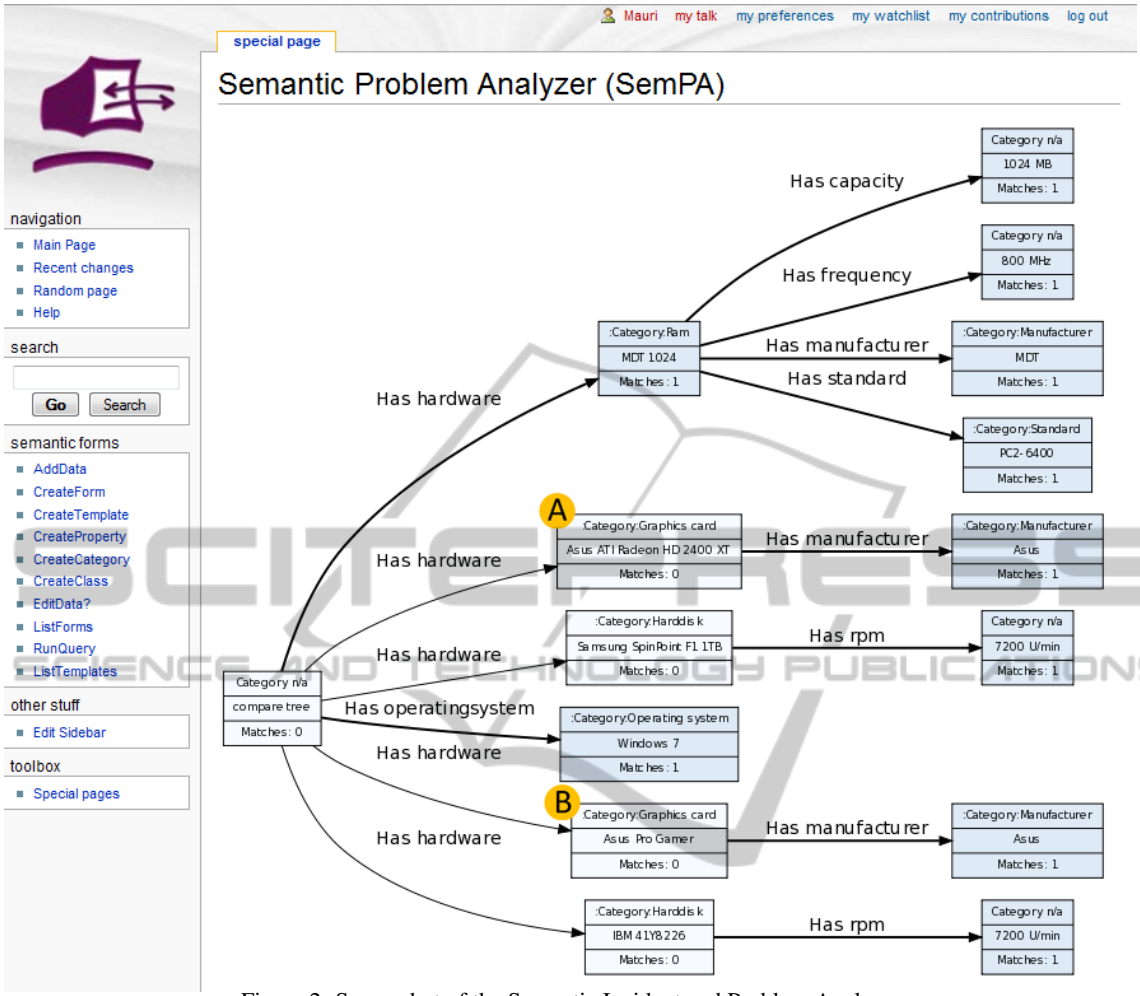

Figure 2: Screenshot of the Semantic Incident and Problem Analyzer.

With regard to using a Semantic Wiki in the IT Service Management domain, the work presented in (Alquier et al., 2009) describes a Semantic Wiki-based Knowledge Management System, which is used for Asset and Configuration Management, documentation, as a self-help system, and for system outage tracking. As far as documented in that paper, the authors follow exactly the same line of thinking as we do in our prior work (Kleiner and Abecker, 2009), i.e. they support the SACM process of ITIL; but they do not offer any help in Incident or Problem Management which is our focus with the SemPA extension. In a recent communication, (Lane, 2010b) also supported the idea of an SMW-based ITSM infrastructure; in (Lane, 2010a), an SMW-integrated ticketing system is discussed which could be a useful complement for our approach. However, for the time being, our organization's requirements are satisfied by the existing OTRS ticketing system. In the future, the benefits of a deeper semantic integration of the ticketing system may be examined.

Regarding related work in the application task, we are, of course, well aware that the task of diagnosing problems in complex technical systems has been thoroughly investigated in the area of knowledge-based systems (KBS) for many years (see, e.g., (Darlington, 1999)). However, it is also a long-standing and not yet satisfactorily solved problem to widen KBSs's knowledge acquisition bottleneck. This is the reason that many technology providers and researchers in the KBS area also produce, since a number of years, *intelligent advisory and assistant systems* in order to find the sweet spot in the trade-off between power and maintainability of software systems. In this sense, the Semantic Incident and Problem Analyzer represents a light-weight approach which totally avoids the problem of upfront knowledge engineering and simply exploits the already existing IT Service Management knowledge base to locate potential problem causes—thus helping the human problem-solver to analyze the

situation and navigate through the problem space. In this way, the system can start-up almost without any specific additional effort. If the system is in place for some time, it may be an option to gradually enrich the system's capabilities, for instance, by fault diagnosis heuristics derived from prior problem cases.

5.2 Implementation Status and Next Steps

The Semantic Incident and Problem Analyzer is currently in productive use and evaluated in a productive environment with about 500 computers. An early qualitative evaluation analyzing a few problems previously encountered in productive use, has shown that the time needed for tracking down problems can be reduced by the Semantic Incident and Problem Analyzer. The amount of time reduction is related to the knowledge about the details of the IT landscape of the person working on the problem, with inexperienced personnel benefitting more from the Problem Analyzer than experienced employees.

However, before having a reasonable critical mass of usage experience for a valid quantitative evaluation, a several-months period of operational use will probably be required. At the moment, a handful of full-time and part-time employees of our IT-support personnel is using the whole Semantic Wiki-based IT Service Management solution (sketched above in section 3) for their daily work. This leads already to a more collaborative work-style and allows for more agile and light-weight IT Service Management processes than typically asked for in IT Service Management endeavors. Next steps of the system roll-out will comprise to open the platform for technologically knowledgeable end users, thus enabling some IT Service Management self-service offerings. This must be accompanied by appropriate considerations about usage incentives such that the self-service IT Service Management portal will offer win-win situations for end users and for IT support personnel as well. In the long-term, we hope that we can study kind of collaborative knowledge creation and exchange through the IT Service Management Wiki, especially regarding observed system failures and problematic configurations. The so-collected experience about observed problems will be the basis for an increasing usefulness of the Semantic Incident and Problem Analyzer.

Seen from the Artificial Intelligence point of view, the presented solution in its current status is certainly a "lightweight semantics" solution profiting not so much from deep and sophisticated ontologies and automated inferences; instead, it is a first, pragmatic and practically useful solution which puts a Semantic Wiki into a daily-business context, addressing a widespread application problem; before coming to the sophisticated stuff, we first implemented all necessary interfaces and connectors to integrate the system into a real-world environment in its full complexity.

In this application context, our first benefits are based on the simple and easy-to-use features for collaboratively creating and editing in a browser-based style, a knowledge base, in the case of a Semantic Wiki containing both structured (i.e., data) and unstructured (i.e., text and multimedia documents) information. The second benefit is realized through the "integrative power" of the Semantic Wiki which is an extremely flexible and open tool with an expressive data model that allows to integrate practically all kinds of ITSM-relevant data, information, and knowledge in the appropriate way. To do so, a first reusable result of our work is the ITSM ontology that combines all ITSM-related aspects of hardware, software, infrastructure and organizational aspects and which can serve of kind of a reference ontology for semantic ITSM / CMDB applications.

In the presented application for finding the possible causes of system problems, simple tree-comparison algorithms have been applied to the formally represented CMDB data. In a traditional AI-approach, one would probably have tackled such problems by a rule-based or a case-based expert system. While the former can only be built when a reasonable account of expert knowledge is available and formalized (which is not always the case, but is always expensive), the latter is also applicable in cases where the problem-analysis knowledge is just slowly growing with experience and may change often over time; and this is exactly the case for our application scenario which can very easily be sorted into the case-based reasoning paradigm, but with a very lightweight knowledge-representation approach that allows for simple and efficient analysis algorithms. In later extensions of our system, we may also investigate the usefulness of complex similarity measures to assess the usefulness of slightly different stored information. But this only makes sense when some longer user experience will have shown which kinds of problem causes can be found by the system and which ones cannot.

One further system extension is probably more nearby and more obviously useful: If longer use of the system will have identified a number of problematic system configurations, these configurations (or, abstractions of them) can be converted into forbidden configuration patterns which could proactively be tested for when new systems are configured or new software is installed. In this manner, known prob-

lems can be avoided before they are repeated. If we are analysing system change logs over time, we might even be able to identify forbidden configuration paths (sequences of actions) that might proactively be tested for by Complex-Event-Processing machinery.

REFERENCES

Addy, R. (2007). *Effective IT Service Management: To ITIL and Beyond!* Springer, Berlin, 1st edition.

Allemang, D. and Hendler, J. A. (2008). *Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL*. Morgan Kaufmann, Burlington, 2nd edition.

Alquier, L., McCormick, K., and Jaeger, E. (2009). knowIT, a Semantic Informatics Knowledge Management System. In *Proceedings of the 5th International Symposium on Wikis and Open Collaboration*, WikiSym '09, pages 20:1–20:5, New York, NY, USA. ACM.

Barrett, D. J. (2008). *MediaWiki (Wikipedia and Beyond)*. O'Reilly, Sebastopol, 1st edition.

Barth, W. (2005). *Nagios: System and Network Monitoring*. No Starch, San Francisco, 1st edition.

Caetlidge, A., Hanna, A., Rudd, C., Macfarlane, I., Windebank, J., and Rance, S. (2008). An Introductory Overview of ITIL V3. http://www.itsmfi.org/content/introductory-overview-itil-v3-pdf.

Cannon, D. and Wheeldon, D. (2007). *Service Operation ITIL, Version 3 (ITIL)*. Stationery Office Books, Norwich.

Case, J., Fedor, M., Schoffstall, M., and Davin, J. (1990). Simple Network Management Protocol (SNMP). RFC 1157 (Historic).

Clacy, B. and Jennings, B. (2007). Service Management: Driving the Future of IT. *Computer*, 40(5):98–100.

Darlington, K. (1999). *The Essence of Expert Systems (The Essence of Computing Series)*. Prentice Hall, Upper Saddle River.

Ebersbach, A., Glaser, M., Heigl, R., and Warta, A. (2007). *Wiki: Web Collaboration*. Springer, Berlin, 2nd edition.

Jones, D. (2007). *VBScript, WMI, and ADSI Unleashed: Using VBScript, WMI, and ADSI to Automate Windows Administration (Unleashed)*. Addison-Wesley Longman, Amsterdam, 2nd edition.

Kleiner, F. and Abecker, A. (2009). Towards a Collaborative Semantic Wiki-based Approach to IT Service Management. In Paschke, A., Weigand, H., Behrendt, W., Tochtermann, K., and Pellegrini, T., editors, *Proceedings of I-SEMANTICS '09, 5th International Conference on Semantic Systems*.

Kleiner, F., Abecker, A., and Brinkmann, S. F. (2009a). WiSyMon – Managing Systems Monitoring Information in Semantic Wikis. In *Advances in Semantic Processing, 2009. SEMAPRO '09. Third International Conference on*, pages 77–85.

Kleiner, F., Abecker, A., and Liu, N. (2009b). Automatic Population and Updating of a Semantic Wiki-based Configuration Management Database. In Fischer, S., Maehle, E., and Reischuk, R., editors, *Informatik 2009 – Im Focus das Leben*, volume P-154. Köllen, Bonn.

Krötzsch, M., Schaffert, S., and Vrandečić, D. (2007). Reasoning in Semantic Wikis. In Antoniou, G., Aßmann, U., Baroglio, C., Decker, S., Henze, N., Patranjan, P.-L., and Tolksdorf, R., editors, *Reasoning Web*, volume 4636 of *Lecture Notes in Computer Science*, pages 310–329. Springer.

Krötzsch, M., Vrandečić, D., Völkel, M., Haller, H., and Studer, R. (2007). Semantic Wikipedia. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(4):251–261. World Wide Web Conference 2006, Semantic Web Track.

Lacy, S. and Macfarlane, I. (2007). *Service Transition, ITIL, Version 3 (ITIL)*. Stationery Office Books, Norwich.

Lane, R. (2010a). Creating a simple ticketing system with Semantic MediaWiki. http://ryandlane.com/blog/2010/04/01/creating-a-simple-ticketing-system-with-semantic-mediawiki/.

Lane, R. (2010b). Helpdesk system and datacenter inventory Semantic MediaWiki prototypes added to my prototype wiki. http://ryandlane.com/blog/2010/03/29/helpdesk-system-and-datacenter-inventory-semantic-mediawiki-prototypes-added-to-my-prototype-wiki/.

Northcutt, S. and Novak, J. (2002). *Network Intrusion Detection: An Analysts' Handbook*. New Riders, Berkeley, 3rd edition.

Roesch, M. (1999). Snort - Lightweight Intrusion Detection for Networks. In *LISA '99: Proceedings of the 13th USENIX Conference on System Administration*, pages 229–238, Berkeley, CA, USA. USENIX Association.

Schaffert, S., Bry, F., Baumeister, J., and Kiesel, M. (2008). Semantic Wikis. *IEEE Software*, 25(4):8–11.

Staab, S. and Studer, R. (2009). *Handbook on Ontologies. (International Handbooks on Information Systems)*. Springer, Berlin, 2nd edition.

Vrandečić, D. and Krötzsch, M. (2009). Semantic MediaWiki. In Davies, J., Mladenic, D., and Grobelnik, M., editors, *Semantic Knowledge Management*, pages 171–179. Springer, Berlin.