UNIVERSITY OF GOTHENBURG

Visualization of Log Files of Embedded Broadband Modules

Master of Science Thesis in Software Engineering and Management

ILYA BELIANKA
ALEXANDER BELYAKOV

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Göteborg, Sweden, March 2012

Visualization of Log Files of Embedded Broadband Modules

ILYA BELIANKA
ALEXANDER BELYAKOV

Supervisor: MIROSLAW STARON
Examiner: MATTHIAS TICHY

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Cover:
Visualization is important for data analysis.

Department of Computer Science and Engineering
Göteborg, Sweden. March 2012

Visualization of Log-Files of Embedded Broadband Modules

Alexander Belyakov
IT University of Göteborg,
SE-412 96 Göteborg, Sweden
gusbelyal@student.gu.se

Ilya Belianka
IT University of Göteborg,
SE-412 96 Göteborg, Sweden
gusbelil@student.gu.se

ABSTRACT

BACKGROUND: Exponential increase in the amount of software in consumer telecom products has resulted in growing needs for resolving maintenance issues together with customers. The resolutions often require understanding systems status through understanding its log-files. As the log-files contain sensitive information, new ways of visualizations are needed to allow for efficient dialog between customers and software maintenance engineers without disclosing the sensitive content.

METHOD: The research presented in this paper has been conducted together with our industrial partner – Ericsson – and follows on the action research methodology.

RESULTS: The visualization tool and MS Windows Sidebar Gadget were developed during the first circle of action research. The tool is applicable for log analysis and over 90% of respondents estimated time savings as minimum 20%. The potential of using gadget for real-time and acceptance testing was also revealed.

CONCLUSION: The overall impact of the tool's utilization at the studied department leads to effort reductions and time savings as minimum 20% during the key activities. Applying the gadget to real-time testing has the potential to enable immediate problem notification, simultaneous monitoring and concentrating testing efforts on machines where errors occur

General Terms

Management, Documentation, Design. Security, Human Factors, Theory, Verification.

Keywords

Software Engineering, Action Research, Integration, Maintenance, Visualization.

1. INTRODUCTION

The research performed by Analysis Mason Limited [1] forecasts the growth of the worldwide telecoms software market from USD 20.1 billion in 2009 to USD 29.5 billion in 2014, meaning the exponential increase of the software component in the telecoms industry. This trend means increasing complexity of the telecoms software and introduces a strong need for applying software engineering techniques for resolving problems specific to this industry. The growth is a consequence of the rise of mobile networking and increased availability of the Internet during the last two decades. Being one of the largest players on the telecoms market, Ericsson has been affected by the trend of increasing the amount of software in the telecoms products and services.

While Ericsson is a world-leading provider of telecommunications equipment and services to mobile and fixed network operators, it is a vendor for the *Information Communications Technology* (ICT) hardware manufacturing sector. The general strategy in this sector is based on outsourcing components' development to different vendors and that is why their annual reports reveal complex web of suppliers [2]. In many cases one particular component could be supplied by two or even more different vendors. As an example, [2] describes supply chain of Acer Inc., which consists of 17-18 component manufacturers. This strategy in the ICT hardware manufacturing sector has created more opportunities for new vendors to enter the market.

The increased complexity of the telecoms products and services leads to development and maintenance of large software systems. Since software is developed on the vendor's side for specific components, *Original Equipment Manufacturers* (OEMs) and *Original Design Manufacturer* (ODMs) have to integrate different hardware and software components into complete systems and to maintain their software systems. ISO/IEC 12207 [94] mentions software integration and software maintenance among the most significant and potentially problematic processes during pre- and post-release phases of the software life cycle. Integration typically introduces problems at the OEM's and ODM's side because of construction flaws and compatibility issues, which are the results of a wide range of components and their vendors that are currently on the market [2]. Additional problems occur during software maintenance stage because of the increasing complexity of software in telecoms products. Therefore, continuous communication between vendors, OEMs and ODMs for collaborative work on improvement of vendors' components becomes an essential part of the integration and maintenance processes. Despite log-files generated by components are employed as means for communication among vendors, OEMs and ODMs, their use introduces other problems, which have to be identified and resolved.

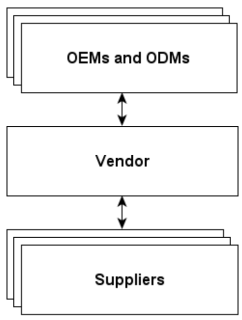

Figure 1. Vendor's collaboration chain.

Every aspect of the collaboration chain between vendor and external participants (Fig. 1), namely OEMs, ODMs and own suppliers, has its own specific issues. The current study involves analysis of problems at all levels of the collaboration chain:

1. The main problem is to make sure that every detail for finding root causes of reported issues is captured in the log. As it happens in practice, OEMs often send log files without this information thus making further investigation impossible and considerably increasing the turnaround time between submission of a failure report and acceptance of its solution.

   Since the size of network log-files can grow rapidly it becomes problematic to quickly analyze such a large amount of text [3, 4, 5]. Vendors generally face this problem in situations when OEMs send log files capturing hours of component's work. This complicates the process of failure detection and further increases turnaround time.

   Another problem is the sensitivity of information, which could be used to compromise security aspects of networks and therefore must not be disclosed [3, 6]. Consequently, it becomes necessary to provide OEMs with possibility to conclude that log actually contains information about the problem without disclosing log itself.

2. In order for vendor's developers check and analyze the failures report obtained from OEMs, understanding requires in-depth knowledge of log-files' content, which might be difficult to find at any given moment. Therefore, developers need additional help to identify root causes of failures [7].

3. The most crucial problem at the stage of communication with vendor's suppliers is finding the exact area in the log where the problem is actually manifested and to extract that area for further analysis. The reason for this problem is the fact that the current tools are not able to work with large files since their performance degrades severely.

In order to improve the quality of products by enhancing faults elimination on the earliest stages of the development process vendors continuously search for additional methods, tools and actions that could minimize the number of faults discovered after release.

The strong potential for facing integration and maintenance issues with OEM-specific platforms during the development of embedded broadband modules for notebooks, smart phones and other mobile devices became the main driver for Ericsson to organize this research project. Needs to find the imperceptible balance between non-disclosure of sensitive information to OEMs and to capture any detail for finding root causes of reported problems introduced the necessity to provide OEMs with the possibility to conclude that the log actually captured the problem without disclosing the log itself. Therefore, presenting selected non-sensitive parts of the log to OEMs in a user-friendly way emerged as the basis for fulfilling the goals of the current research project:

- to improve feedback from the customers of the studied software development unit;
- to decrease number of faults discovered after release;
- to enhance error localization without disclosing sensitive information;
- to research possibility to use the visualization tool for improvement of other stages of the development process;
- to decrease the amount of time spent on maintenance of developed software.

In this paper we present the results of the research conducted at one of Ericsson's departments, which has the aim to discover and analyze problems in the process of work with log-files from different perspectives. We present two applications that were developed as improvement suggestions for the current business processes. The implementation and further usage of them had an intention to decrease the turnaround time between submission of a failure report and acceptance of its solution, to prevent disclosure of any sensitive information and to decrease a number of faults discovered after release. We compare these applications with already existing ones and their usage at the company using semi-structured interviews and a survey. Finally, we analyze possible impacts of these tools on the business processes at the studied software development department and suggest further ways of improvements and the aims for the second circle of the action research.

The rest of the paper is organized as follows: Section 2 provides information about related work. Section 3 describes the action research method we used. Section 4 describes the diagnosis of the studied software development department. Section 5 describes the suggested way to resolve identified problems. Section 6 describes the implemented log visualization systems and demonstrates usage of the available visualizations. Section 7 describes the evaluation of those systems at the studied department and in Section 8 the results of this evaluation are analyzed. Finally, Section 9 presents the conclusion and Section 10 discusses future work.

2. RELATED WORK

Humans are effective at analysis and searching for patterns in images rather then in text data [8]. Additionally, visual methods can provide valuable assistance for data analysis and improve decision making process [9] and reveal important information that supplements the knowledge obtained from the application of more generally-used statistical approaches [10]. Therefore, the graphical representation of the information could considerably accelerate the process of data mining from log files.

The technologies of text visualization and visual text analysis originated in 1960s information science researches. Doyle [11] proposed the idea of development of automated systems for creation and interaction with the graphical "association maps" of the libraries' content. Nevertheless, systems for visual text analysis became useful in practice only in 1990s, when computational processing and human-computer interaction interface technology reached the required level of technology development [12].

One of the first practical attempts to implement text visualization software was the SPIRE system developed in 1995 [13]. This application includes the functionality for creation and interaction with the graphical representation of the document's content. Since log files are sequences of time-stamped events, which are "corresponding to input and output from the system, as well as internal state transitions and state readings" [14], they can be considered as simple text files. The research in the area of log-files' visualization took place from the early 1990s till present resulting in a number of the systems: LOGSCOPE [14], SeeLog [15], EESoft [16], VISUAL [17], Spotfire [18], Webviz [19], MieLog [20], LogView[21], SnortView[22], Session Viewer [23], Eventbrowser [24], PaintingClass [10], ADVIZOR [25], etc.

The most important for us in the conducted studies are the goals those studies pursued. By generalizing information from different science papers and conferences we could distinguish the following purposes of log-files visualization:

- Intrusion detection [10, 22, 26];
- Anomaly detection [10];
- Faults detections [22];
- Visually analyzing the user's behavior [27, 28, 29];
- Detection of unusual system's behavior [20, 23];
- Multi-node monitoring in big network systems [29, 30];
- Understanding and analysis of patterns in data [17, 18, 31];
- Checking execution traces against formal specifications [14];
- Accelerating data analysis during system's downtimes [21];
- To "support session log analysis at both the statistical-aggregate and the detailed-session analysis levels" [23].

Researchers agree on the idea that log-files are too large [8, 14, 15, 20, 21, 32, 33] to be analyzed effectively without additional help provided by special tools. Furthermore, additional problems exists such as log noise [15], which could cause human to miss important information during analysis, disregarding common fault messages in a file [15] or inability for a human to distinguish patterns in a large amount of data[33]. Therefore, the main domain of usage of log files' graphical representation is the defining of user's or system's behavior and anomaly or intrusion detection.

There are many visualization techniques described in the science papers. Specific approaches were identified in different areas of application. The taxonomy made by Kasemisri [32] clearly defines the most common techniques for network security visualization, which could be definitely used for the purposes of visualization of log-files generated by embedded broadband modules: scatter plots [17, 34, 35, 36, 37, 38], color maps [39, 40, 41, 42, 43], glyph [26, 29, 37, 44, 45 46], histograms [20, 27, 42], parallel coordinate plot systems [30, 38, 47, 48, 49], and some others [50, 51, 52, 53, 54]. With regards to visualization for verification purposes some traditional types of visualization are used: state diagrams [14, 55, 56], tree-view diagrams [21], 2-D time diagrams[22]. Additionally, there is a number of studies that propose new methods for logged data visualization, for example Hierarchical Network Maps (HNMaps) [57], Identifier Graphs [10], Graphs Bridges [58], Histogram Matrix [49]. Meanwhile, researches pay attention not only to the structure of a graphical representation itself, but also to basic human-interaction techniques such as color-coding [18, 20] or interaction with visualized figures [20].

Not every application of those surveyed had possibility to set up visualizations. Most of them use predefined rules for visualization construction and allows only changing minor attributes, like color or size [10, 14, 19, 23]. Moreover, they do not provide users with an ability to select specific information from files by defining an area of text for visualization using specific language or regular expressions. Nevertheless, some works defined approaches that have been applied in the area of language definition for the visualization purposes such as data-parameterized temporal specification language [56], regular expressions-based language [59] and attribute-based language [21]. The significant result of the work by Baffinger et al [14] is that "the engineers find it very effective to write specifications in the pattern language and check their precise semantics observing the automaton visualizations" [14]. Additionally, authors in [14, 55, 56], state that it is important for pattern language to be so simple that engineers are able to learn it quickly and it should be useful for expressing required properties in visualization.

The main studies in log data visualization for analysis purposes have been held in the field of network security [17, 32, 37, 43, 50, 60] and mostly refer to log monitoring, as well as in the related area of intrusion detection [26, 27, 29, 34, 35, 39, 44, 45, 47, 61, 62]. Meanwhile, use of visualization practices for data analysis is under studies in the field of runtime verification. These studies have an intention to check execution traces against formal specifications [14]. Researches make emphasis on time-stamped events that correspond to input and output operations in a system in conjunction with state transitions and state reading.

The purpose of visualization of log-files in this research, unlike the mentioned researches, is improvement of communication between vendors and OEMs with ODMs, achieved through accelerated error localization without disclosing sensitive information, and decreased number of faults discovered after release. The list and detailed descriptions of the visualization techniques are stated below. An attribute-based language was selected for setting up visualization parameters. Additionally, this research has been performed in the telecoms industry and the studied log-files were obtained from embedded broadband modules. There were no studies of visualization of log-files in this industry before.

3. RESEARCH METHODOLOGY

The research was conducted using the empirical approach [63, 64] based on the qualitative research method [65]. Action Research (AR) process [66, 67] had been chosen as the most appropriate methodology for combining theory and practice. Its application purposes and scientific value have been widely discussed and used for industry projects during the last decades [66, 68, 69, 70, 71, 72, 73, 74, 75, 76]. Since the research was conducted entirely at Ericsson, we have been given an opportunity to actually become a part of the company and to see the development process from inside, to learn it and to introduce its change with use of the visualization tool, and to reflect on its consequences. Therefore, the choice of AR deemed to be the ideal match for capturing the change and reflection on it from inside. Finally, the cyclical nature of this research method was also considered as the most appropriate due to the iterative development and deployment of the visualization tool and the gradual shift of the industry towards agile software development.

The major constraints for the research methodology have been defined by the goals of the conducted research: the organizational development of the industrial partner's project and scientific knowledge. Besides, strong involvement of academia determined the choice of rigorous structure [77] of the research. Finally, since the research work was done on site, both researchers and Ericsson's representatives worked together on the problem in close collaboration. All the above-mentioned together with the iterative nature of our research imposed the selection of Canonical Action Research (CAR) [67, 78] modification of the AR. This research model had been proposed by Susman and Evered [67], further supplemented in [78] and widely used in research projects [79, 80, 81]. We followed five traditional phases of action research: diagnosing, action planning, action taking, evaluating and specifying learning (Fig.2) [67].

At the diagnosis stage we have identified the major problems at the studied department at Ericsson and discovered the divisions that are mostly affected. Interviews were the main instrument of data collection and performed at the studied department with employees from different divisions. In our work we also used literature review technique that provided information about log-files visualization problems detected in other studies.

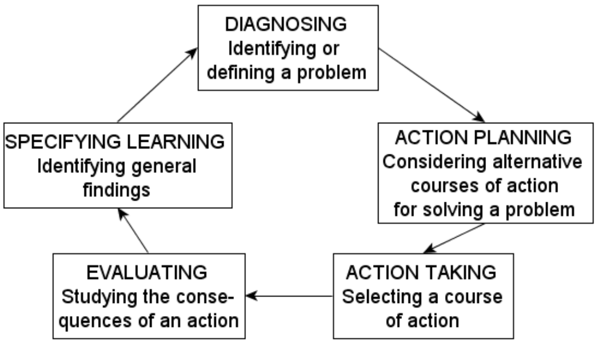

Figure 2. Action research iterative process.

After the thorough investigation of the main issues of working with log files we proposed a possible solution for the visualization system, which will be referenced further as MBLogVis. Firstly, together with our industrial partners from Ericsson we defined the boundaries for the development and produced the description of the system's requirements. Additionally, the development process was established and the list of actions to be taken was compiled.

The action taking phase of the project had stronger focus on development of applications for parsing and visualizing of the log-files. Since identification of problems at the company tended to be a permanent process, it was done in parallel with the actual development. The prototype of the system for visualization of log files was developed during this phase.

Afterwards, at the evaluation stage we assessed the applicability of the developed tools for the needs of our industrial partner. Additionally, we received valuable feedback on our work and in collaboration with Ericsson delineated the possible ways for further organizational development.

The work done in the action research circle from diagnosis till evaluation was additionally analyzed and its outcomes were reviewed at the specifying learning stage. We investigated both positive and negative aspects of the actions taken in this research. Results will become the basis for further development in the next circle of action research and for building a model of the situation under study.

Due to time limitations of the described research only the first circle of action research was performed.

4. DIAGNOSIS

4.1 Organization

Ericsson became a strong player among the vendors for ICT OEMs as a result of the launch of development of Mobile Broadband Modules (MBM) for notebooks, smart phones and other mobile devices following its decision to support the Networking Society initiative. Over the last years Ericsson has become a vendor to a number of large OEMs at the ICT hardware manufacturing sector while kept delivering developed devices to smaller customers and selling them through indirect distribution channels. This was a result of its strategy to drive the industry towards mass-market products, enabling wireless connectivity from any device at a reasonable cost [82].

The growing success of Ericsson in the ICT hardware manufacturing sector was a direct result of its close collaboration with OEM customers and involved adjustment of business processes. A division at the MBM department was established for communication with OEMs and resolving integration issues with MBM products on OEM's specific platforms. It operates both on the company and OEMs' sides and will be referenced further as COEM. Another division was established for the purpose of communication with hardware suppliers regarding the platform for MBM devices and will be referenced further as CS. Therefore, collaboration among the divisions of the MBM department and its suppliers and OEMs has the following structure:

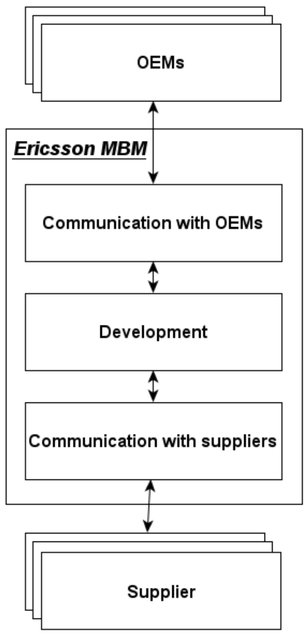

Figure 3. MBM department collaboration.

These divisions enable investigating, tracing and resolving communication issues at different levels from semiconductors to deployed networks.

It is worth noting that the development division (DD) plays an important role in this process. Developing drivers for the MBM devices and additional firmware for interaction between modules and operating systems, it acts as a transaction layer between the detected high-level failures on the OEMs' side and low-level investigation of errors on the vendor's side. The DD conducts decisive checks for failures obtained from the COEM searching for errors in firmware implementation and in case of their absence forwards the request to the CV for further investigation.

With regards to maintenance of products, the studied department follows the ISO/IEC 14764 standard [95], which defines six main activities: *Process Implementation*, *Problem and Modification*

Analysis, *Modification Implementation*, *Maintenance Review/Acceptance*, *Retirement* and *Migration*. However, from long-term maintenance perspective the most important are the three core recurring activities: *Problem and Modification Analysis*, *Modification Implementation*, *Maintenance Review/Acceptance*. The complete process structure is shown in Figure 4.

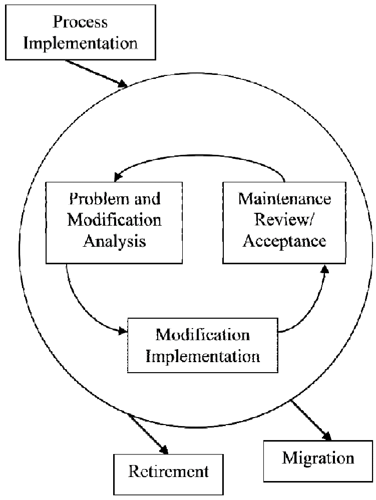

Figure 4. Maintenance process activities.

4.2 Problems Definition

Such a profound complexity in the chain of interactions leads to different communication and collaboration problems. Despite using logged sequences of events produced automatically by capturing the type, content and time of every transaction made within the system [83] as an instrument of communication is a reasonable approach, using these logs for problem diagnosis and root cause analysis [7] still remains difficult. Since working with logs is spread among several divisions of the MBM department, the problem area at Ericsson could not be defined from just one point of view and needed to be analyzed from different perspectives instead.

One of the most crucial parts of the analysis was linking the problems at Ericsson with the relevant divisions of the MBM department in order to make the problematic area clearer thus enabling focusing on the most significant aspects. Interviews were the main instrument of data collection and performed at the studied department with employees from different divisions. We held 9 semi-structured interviews with average duration of around 45 minutes. The information was subsequently analyzed and the results were further utilized in the research. In our work we also used literature review technique that provided information about log files visualization problems detected in other studies. Furthermore, observation of the work at the department was conducted, which provided knowledge about the internal business processes.

During the first stage of the analysis it became clear that problems and needs related with sequences of events of MBM devices in operation are actually shared by the DD and COEM. Therefore, it became necessary to identify log-related problems for each of these groups.

DD's problems:

- In order for developers at Ericsson to check and analyze a failure report obtained from the division that communicates with OEMs, they need to have experience in working with log files and deep knowledge of communication standards, which might be difficult to find at any given moment at the DD. Therefore, developers might need additional help to identify root causes of failures, which leads to significant growth of time for problem identification.
- Another crucial need at the development division is to find the exact area in a log where a problem is actually manifested and to extract that area for further analysis. The fact that the tools currently used at the company are unable to work with large files, especially with files larger then 26 MB, since their performance degrades severely, considerably impedes the troubleshooting process. It leads to time expenses and increased turnaround time between submission of a failure report and acceptance of its solution.
- An integral part of log analysis is locating faulty changes of states for specific registers of the system. This process includes detection of sequences of state changes, which is time-consuming for developers.

COEM's problems:

- OEMs frequently send log files that describe the device's operation over a certain amount of time, on average around half an hour. However, since the size of MBM log-files can grow rapidly, up to 200-400 KB/min of raw data, depending on the settings of the log-capturing application, these files become large thus making quick analysis problematic [3, 4, 5]. It complicates the process of failure detection and further increases the turnaround time.
- The terminology and structure of the wireless communication protocols is strictly imposed by the industry standards. Consequently, the same terminology and abbreviations are used in log files and understanding them becomes problematic for users. It leads to time-consuming search of term definitions in the industry standards.
- For finding root causes of reported issues it is crucial that every detail must be captured in the log. However, as it happens in practice, OEMs often send log files without this information thus making further investigation impossible and considerably increasing the turnaround time between submission of a failure report and acceptance of its solution.
- Logs generally contain sensitive information, which could be used to compromise security aspects of a mobile network and therefore must not be disclosed [3, 6]. Consequently, it becomes necessary to provide OEMs with a possibility to conclude that the log actually contains the information about the problem without disclosing the log itself.

COEM's on the customers' side problems:

- To prevent the negative consequences of disclosure of sensitive information in log files, encryption of logs is performed. However, while solving the security issue, encryption makes it impossible to check if problems with

MBM devices in operation have actually been caught in log files.

- The process of log generation on the OEMs' side does not involve actual analysis of the log. Consequently, there is no possibility to check if there are failures at run-time. Therefore, employees have to spend time on log generation first and on log analysis afterwards, which leads to time being lost.

The most prominent example of a problem encountered at the OEM's side during the assembly stage is the situation when issues with MBM device in a laptop are caused by unacceptably high level of white noise, which makes an antenna incapable of distinguishing a signal from noise. As a result, the MBM device can no longer perform cell selection procedure. Even though the module could try and even succeed in connecting using another mode at different base frequency, the root cause of the issue still needs to be resolved. This issue can be found and resolved only by working with sequences of states, transitions and other events in the MBM device's log, which cannot be disclosed to the OEM. Otherwise, it might lead to further information leaks creating network vulnerabilities, damaging vendor's brand or causing unfair competition among different vendors.

Analysis of problems at the MBM department clearly identifies that all these issues are actually caused by difficulties in extracting meaningful information about every single MBM device operation error. Due to the fact that this information could typically be represented differently depending on a situation and spread among different parts of a long log file, it becomes extremely cumbersome to find error causes.

To a certain extent the described problems have been addressed by functionality of the tools currently in use at Ericsson. Analysis performed at the MBM department revealed the number of advantages and disadvantages of those tools.

- The most advanced tool used at the MBM department covers the needs of all MBM divisions including log-analysis, folding structure and pattern-based search. However, its biggest drawback is degrading performance when a log-file is larger than 26 MB to the extent of making the tool non-operational. Therefore, a way to limit the log size by leaving out irrelevant information was needed in order to maximize the impact of using MBLogVis.
- Another tool supports obtaining different kinds of log-files from devices and framing a basic overview. However, search possibilities of that tool are limited just to simple text without any search for patterns.
- The third tool supports analysis of small parts of log-files by organizing the decoded information into various drop-down subcategories. It is typically used in situations when the root cause of a problem is possible to identify by analyzing messages sent by devices to networks and vice versa.
- The fourth tool is a text editor with additional functionality that allows performing advanced search of sub-strings using regular expressions, saving search results and templates.
- The fifth tool is a description of the log-files provided by the hardware platform supplier with additional description of logged sequences of events of MBM devices in operation produced by a CV specialist with over 7 years of experience. It contains specification of the most problematic areas and provides examples of error patterns and possible ways of deeper log files analysis. Since this description could not automate log analysis process it might be useful only with additional tools for text analysis.

Nevertheless, while those tools provided some of the functionalities needed for log-files analysis, their limitations affected the possibility of their wide use. Therefore, ensuring the suitability of the tool being developed for working with wide range of log-files, especially larger ones, became the most necessary requirement for making the tool useful.

Therefore, the thorough investigation of the main issues of working with log-files at the MBM department as well as on the OEM's side resulted in the detailed description of log-related problems. Besides, the evaluation of the tools already in use revealed their advantages for particular aspects of log-analysis as well as disadvantages limiting their wide application. The evaluation results together with the problems definition served as the basis for the suggested improvements, which are described in the next section.

5. ACTION PLANNING

5.1 Visualization

The investigation of log-related problems at the studied department leads to the conclusion that they were caused by unsatisfied needs of its various divisions. Difficulties at the DD to detect sequences of state changes in large files are direct results of the need for advanced tools at the division while at the COEM time-consuming search for term definitions in the industry standards when working with large files is caused by lack of this functionality in the tools that are currently in use. Besides, impossibility to be sure that problems with MBM devices in operation have actually been caught in log files without disclosing logs to OEMs together with unavailability of run-time check for failures are consequences of the unsatisfied need for the required functionalities in the tools as well.

The problems of computer log files analysis have been studied during the last few decades. The currently growing interest in this area is caused by growing amount of networking equipment that contains a large amount of software that produces logs. Researchers have already agreed on the idea that the log files are too large [8, 14, 15, 20, 21, 32, 33] to be analyzed effectively without additional help provided by special tools. Additionally, log noise [15] causes missing important information by humans, disregarding common fault messages in files [15] and decreases ability to distinguish patterns in a large amount of data [33]. In the middle 1990s the idea of visual text analytics, which originated in early 1960-s [41], became widely used for the purposes of text analysis. Its effectiveness and applicability was confirmed by a number of practical experiments [10, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25].

We propose visual text analytics approach for solving log-related problems by using a group of information analysis techniques and processes that generate interactive graphical representations of textual data for further investigation [12]. Employing those techniques enables use of human visual pattern recognition and spatial reasoning capabilities for addressing log-related problems.

Visual text analytics encompasses techniques that employ visualization of "abstract" data in text files.

The behavior of MBM devices in operation can be exhibited using proposed visualization techniques, described below, and thus enabling more efficient comprehension of situations with devices in operation as well as faster problem localization and elimination. Since visualizations typically strive to graphically depict the overall conceptual structure of a document, they can become appropriate machine-generated "tables-of-contents" for log files that help avoiding disclosure of sensitive information. Furthermore, by enabling user's interaction with text visualizations and employing visualization techniques for simultaneous inspection of different properties it becomes possible to follow system state transitions in the process of log-files analysis.

5.2 Visualization Techniques

Specific approaches are identified for different application areas, namely network security visualization, intrusion detection visualization, visualization for verification purposes. The variety of visualization techniques includes scatter plots, color maps, histograms, glyphs, tree-view diagrams and others. After generalizing characteristics of different visualization approaches we can conclude that color coding is the main technique used for human-computer interaction. Additionally, possibility to interact with graphical representations is another common characteristic of widely used systems.

The suggestion to use visualization techniques for the purposes of log-files analysis was not only a recommendation. Instead, this research had the intention to build a special tool for log-analysis that will provide valuable visualizations for employees at the studied company. Therefore, it became necessary to decide what types of visualization we should use in order to maximize the applicability of MBLogVis to the highest number of problems. For this purpose five semi-structured interviews were conducted involving participation of employees from different divisions. These interviews were planned to take around 60 minutes each and ended with open discussions about applicability of different visualization techniques in specific situations. In addition to interviews, the study of related works in the area of log files visualization was also performed in order to find best practices for further use in the implemented visualizations.

Discussion of the visualization techniques with different divisions at the MBM department resulted in a number of visualization suggestions and ideas. The main characteristics of those suggestions together with their originations are briefly described below:

- The most common proposal from both the DD and COEM division was generating state diagrams, Figure 5, showing transitions among the states in sequences of events of MBM devices in operation. This way is potentially helpful to identify wrong transitions and to locate the area in the log where the problem manifested. However, in addition to just generating a state diagram it was suggested to increase its interactivity by making states and transitions clickable and showing auxiliary information about problematic parts, thus enabling better and easier way of finding exact portions of information needed for analysis.

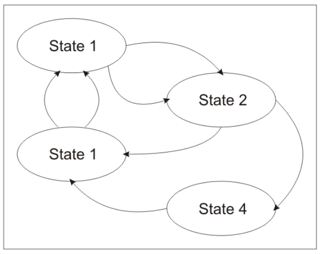

Figure 5. State diagram visualization technique.

- Another suggestion from the DD was generating sequence diagrams, Figure 6. However, due to the fact that sequence diagrams were already available in one of the available tools and generally not used, because of their complexity and time costs to analyze them, this suggestion became redundant.

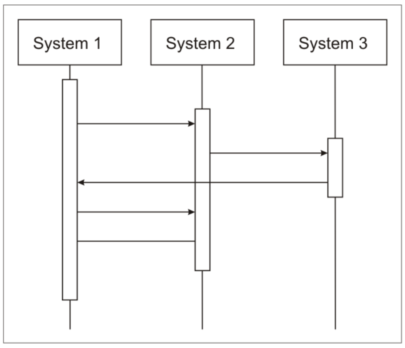

Figure 6. Sequence diagram visualization technique.

- The third suggestion emerged during interviews with the COEM division employees and its idea was showing the states of various registers for specified time stamps, Figure 7. This approach enables understanding the essence of transitions during operation of MBM devices at a particular moment of time without having to search for every register's value in a log file and potentially solves the problem of locating faulty changes of states for specific system registers.

Text line	Specific line from log
Register 1	State of register 1
Register 2	State of register 2
Register 3	State of register 3

Figure 7. Registers states visualization technique.

- In addition to Ericsson's suggestions, we also proposed colorizing logged sequences of events, i.e. generating log-derived documents, using the library of patterns for potential errors and suspicious conditions, where problematic areas in the log file are highlighted using color coding, Figure 8. This approach enables faster identification of problematic areas and simultaneous analysis of different types of information. However, sometimes it could be difficult to define precise patterns and this approach could not be applicable for all OEMs, since relatively large part of OEMs is currently not authorized to see logs.

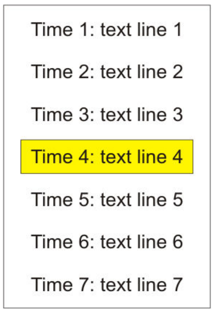

Figure 8. Colorizing events visualization technique.

- Another technique suggested by us was generating overall information about MBM devices in operation, Figure 9. Similarly to log colorizing, it uses search for patterns but without any limitations on being used on the OEM's side since the amount of information shown is much smaller and the nature of information is more generic. Various aspects of diagnostics are represented as a set of traffic lights.

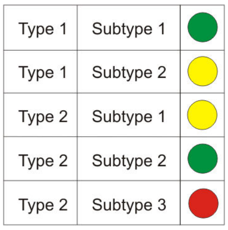

Figure 9. Overall information visualization technique.

One of the most important requirements for the system was possibility to adjust visualizations using configuration files. Some researchers defined different approaches in the area of language definition for the visualization purposes such as data-parameterized temporal specification language [56], regular expressions-based language [59] and attribute-based language [21]. It is important for engineers to have possibility of writing specifications using the patterns language since it increases re-usability of visualization configurations. Two crucial requirements for the pattern language are to be so simple that the engineers should be able to learn it fast and to be useful for expressing required properties for visualization. Therefore, it was chosen to use attribute-based language with the semantics and syntax of widely used XML and Perl languages.

5.3 MS Microsoft Sidebar Gadget

It is necessary to mention the extended potential for applicability of overall information visualization. Since this method does not introduce any information disclosure and visualizations can be generated automatically without any manual operation, in addition to becoming just one of the functionalities of the tool being developed, it can also work as a separate Windows Gadget application and be deployed on a much wider range of machines, including OEM's ones. The described gadget is a part of the conducted research and was developed as a separate application for log analysis purposes.

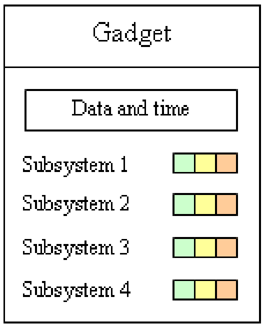

Figure 10. Gadget visualization technique.

The main idea of the application is that it uses the main functionality of the tool described above. It will automatically search for error patterns in a log-file at a remote resource and show up-to-date information to the user. Additionally, its usage assumes that scripts with definitions of fault patterns will be produced by system's experts with sufficient knowledge in log analysis.

Use of this application will potentially improve the quality of log-files obtained from OEMs. Due to the fact that gadget cannot disclose any sensitive information, it will be possible to use it on the OEM's side during log generation being ready to immediately notify about caught failures. Additionally, it will decrease requirements on employees' experience since definition of fault patterns will be written by experts. Also, use of the gadget for testing purposes will introduce the possibility to decrease the number of faults detected after release by accelerating the testing process with immediate notification about potential faults in the system.

This approach creates another perspective about resolving the visualization problem by turning the visualization tool into the diagnostics one. Being run automatically it will eliminate upfront the risk of forgetting to run the application and will immediately notify user if there is any problem.

6. ACTION TAKING

6.1 System Architecture

Attempts to collect and analyze knowledge about the process of data visualization have been taken recently. Ben Fry in [84] indicates 7 steps in the process of data visualization: *Acquire*, *Parse*, *Filter*, *Mine*, *Represent*, *Refine*, *Interact*. Acquisition involves obtaining data from some source, which could be either file on a disk or a source over a network. Second step, Parsing, provides structure for the obtained data and potentially structures the data into categories. Filtering extracts the data of interest from the initial amount of information. Mining used further for patterns recognition and rearranging structures in mathematical context. Representing leads to a choice of basic visual model and Refinement improves the essential visualization to make it clearer. Finally, Interaction appends visualization with methods for data manipulation. Nevertheless, Fry also states [84] that it is not necessarily to follow all the steps in real environment. For the purposes of the research the scheme proposed by Fry was simplified and it was reflected in the architecture of the developed visualization application. This reflection will be further described in this chapter.

The list of non-functional requirements of the system was defined together with industrial partners that considerably influenced the selection of architectural pattern. The most important characteristics of the system are the following:

- Flexibility: characterizes how easily a system or component can be modified in order to be used in applications or environments that were not specifically targeted by its design [85].
- Maintainability: characterizes a software system's possibility for further performance improvements and ability to adapt to environmental changes.
- Extensibility: characterizes the degree to which future growth of a system is taken into consideration.
- Usability: characterizes the ease with which a user can learn to operate, prepare inputs for, and interpret outputs of a system or component [85].
- Portability: characterizes the ease of porting the software to different operating environments namely other host machines and/or operating systems [86].

Therefore, multi-tier architectural pattern was selected for implementation of the application considering the non-functional requirements described above. Since work with database was not among the system's functional requirements, only two tiers were implemented in the system namely presentation and logic tier. Separation of tiers was realized by implementing a library that performs actual data processing and provides its results to be further displayed in the GUI, Figure 11. The diagram summarizes the whole architectural view of the tool. Dark grey boxes represent executable units while white boxes represent information. Boxes with dashed borders represent dynamic entities generated by MBLogVis while boxes with solid borders represent static entities, which are always present and not changeable by the tool itself.

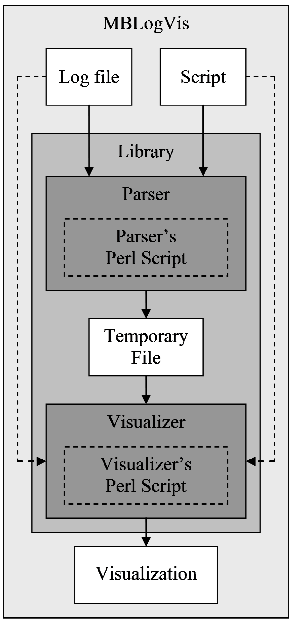

Figure 11. System's architecture.

In the Figure 11 "Script" is a special text file that provides possibility for users to configure parsing and visualization giving additional flexibility in use of MBLogVis.

This architecture takes into account the selected non-functional requirements in the following way:

- Flexibility: separation of presentation and logic tiers provides the possibility to modify each tier separately in order to use them in different environments.
- Maintainability: building two layers of the system allows in most cases to limit code churn caused by improvements and environmental changes.
- Extensibility: use of two tiers in the selected architectural pattern enables further evolution of data storage by adding a database that will be managed by an additional data tier. Besides, use of object oriented programming paradigm supports future growth of the number of supported visualizations by adding new classes and objects.
- Usability: ease of use and learn of the application under development is provided by utilization of widely used programming languages in the Script and Visualization files.
- Portability: ease of porting the application to different operating environments is performed by using the Perl language, which is supported for scripting in the wide range of operation systems. Additionally, generalized abstraction between the application logic and system interfaces defined in the selected architecture pattern is considered by the researchers [87] as a prerequisite for the portability.

The scheme of data visualization process proposed by Fry [84] is reflected in the selected architecture in the following way:

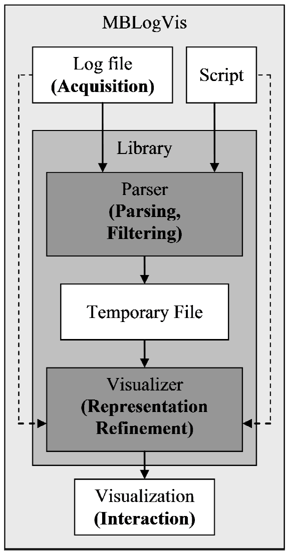

Figure 12. Fry's scheme in the system's architecture.

Since the context of the visualized log files is too simple to be restructured, the *Mining* step of Fry's approach became not applicable for the current visualization process and therefore was not reflected in the application design.

Therefore, the selected architecture pattern together with its detailed design reflects both the non-functional requirements defined in collaboration with Ericsson and the scientific approach to the visualization construction. More detailed information about implementation of the system and its parts is described below.

6.2 System Implementation

From the architectural perspective the implementation of MBLogVis was divided into two main parts: the library of visualizations and the main application. MBLogVis provides user-friendly GUI for working with visualizations, enabling easy access to configuring, selecting, running and saving the required visualizations while the library contains all the visualization techniques, Appendix 2. Dividing the project into these two parts opens the possibility of building several tools targeted for different types of users based on the same library and also enables easier division of responsibilities among the developers.

Visualization mechanisms for all techniques were implemented in two steps: parsing and visualizing. During the parsing stage lines of log are checked for matching one of patterns specified in a script, which are defined using regular expressions. If a match is found, the line will be saved in the temporary file of found patterns for further visualization. During the next stage information from the temporary file is converted into one or several visualizations, sometimes using the contents of the original log-file as well.

For quick parsing using regular expressions the library generates and runs Perl-scripts, which are specific for each visualization technique. This allows using a powerful mechanism of regular expressions in Perl and minimizes the potential amount of code being rewritten if need to port the tool to another platform arises while keeping the performance on the acceptable level.

It is necessary to mention that temporary file and Script files use XML syntax. Furthermore, the HTML markup language is used for the generated visualization file. The choice of the mentioned languages was made due to their wide usage that could possibly help to avoid or decrease additional need for trainings for employees.

In order to better understand the applicability of the implemented system to more specific situations, an example of its usage is provided in the chapter 6.3.

6.3 Example of Usage

In order to generate the desired visualization from the log file user needs to go through the following steps:

1. Select the configuration file that defines the tree of scripts.

   Since it is possible to have several scripts for various patterns and their visualizations, all the patterns are organized in the hierarchical structure consisting of three layers: pattern groups, pattern families and patterns themselves. Further in this paper we call this structure "tree", because of this hierarchy. Therefore, each pattern in this structure is connected to others as a member of the same pattern family or group.

2. Select or write a visualization script.

   The script used for setting up visualization attributes uses semantics and syntax of the widely used XML and Perl languages: the structure of the script is based on the XML and the regular expressions in attribute "name" in "RE" item should be defined using Perl language. The attributes that configure visualization are described in the system documentation. For example, a script for generation state-transition diagram may look as follows:

```
<Script>
 <Items>
  <Item>
   <REs>
    <RE name="\QGTB: *** Sub-State = \E"
       begin="Sub-State = "
       end="***"
       offset="1">
    </RE>
   </REs>
  </Item>
 </Items>
</Script>
```

   The attribute "name" describes regular expression that will be used for searching lines that contain information about changes of states; "begin", "end" and "offset" define place in lines with names of states.

3. Select from the list of visualizations the ones user wants to generate.

   It is possible to generate several visualizations from the same pattern. For example, state diagram is especially useful together with log colorizing for better understanding and tracing the behavior of the MBM device in the log. The user can choose the visualizations to generate from the list of currently implemented ones: state transition diagram, time stamps, log colorizing or overall system information. For the above mentioned script example we will choose "state transition diagram" type of visualization.

4. Select a log-file to work with.

   Since the same script may be applied to different log files, it becomes necessary to define the log-file in the application and not in the script. For this example's purposes we choose a log-file, which contains the following lines:

```
3151687   GTB: *** Sub-State = STARTED ***
3152637   GTB: *** Sub-State = PAUSED ***
3155323   GTB: *** Sub-State = RESUMED ***
3156339   GTB: *** Sub-State = PAUSED ***
3157800   GTB: *** Sub-State = STOPPED ***
3160034   GTB: *** Sub-State = RESTARTED ***
3160564   GTB: *** Sub-State = STARTED ***
```

5. Run generation of visualizations.

   During this step visualizations of a log-file are generated and shown to the user. For of the script described above the following diagram was generated:

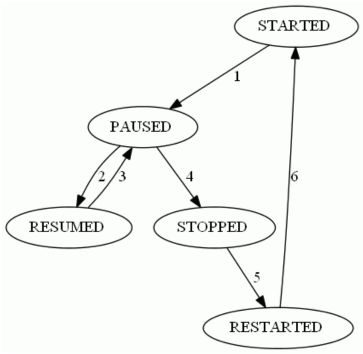

Figure 13. Example of generated state diagram.

6. Save newly-created visualization.

   If there is a need to use the visualization for further reference, it is possible to save the visualization itself and all the files it references.

Therefore, the described example provides general idea of using the visualization tool for log analysis. As well as using the GUI, the tool could also be run as a console application. Additional parameters for console application include locations of a script, log and visualization files. The ability to run the application from console enables its use in the implementation of visualization gadget.

6.4 MS Windows Sidebar Gadget Implementation

The gadget employs running MBLogVis from console on a periodic basis to enable frequent update of the information shown to user in the sidebar. The script-file with the similar structure as described in the previous chapter is used for setting up the attributes that configure selection of information from a log-file. Another representation of the overall system information is used for generating HTML-document with basic statistics of the MBM device in operation. The example of this visualization may look as follows:

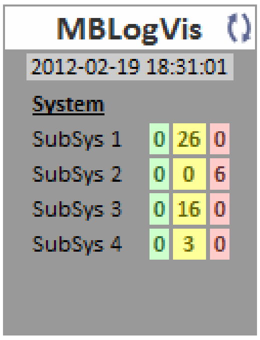

Figure 14. Example of gadget.

7. EVALUATING

7.1 Obtaining Feedback

In order to evaluate the results of the performed work and to define improvements to the business process, different techniques for feedback collection can be used. However, in the software engineering area the list of appropriate techniques is limited to interviews, observations, surveys, archival data, metrics and checklists [88, 89, 90, 91, 92, 93]. While they include both quantitative [89] and qualitative [91, 93] approaches, not every technique is appropriate for collecting feedback about the implemented application and for the assessing improvements made at the company. Interviews, observations and survey have been chosen for feedback collection in the current study.

7.2 Qualitative Assessment

The advantage of using qualitative methods is keeping the complexity of the problem, which is typically abstracted away by quantitative methods [91]. Therefore, the results of qualitative research can be richer and more informative. Interview is among the common techniques used for collecting qualitative data. We applied it to collect opinions and impressions about the developed tool and, more importantly for the goals of this study, to obtain information about the influence that use of the tool can make to business processes. We used semi-structured interviews [90], which included a mixture of closed-ended and open-ended questions.

While research interviews are usually conducted by one interviewer, the interviews in this research project were conducted by two researchers. The role of one researcher was to lead the interview, while the responsibility of another was taking notes and asking additional clarifying questions. This approach was selected due to the nature of such kind of interviews involving longer discussions where more follow-up questions could be asked [91]. Moreover, two researchers had better opportunity to discuss and compare their interpretations and understandings of the interview results, thereby getting better results for the conducted study.

We held 6 semi-structured interviews with average duration of around 50 minutes. Inviting experienced software architects, designers and developers from the DD as well as customer support engineers from the COEM and other divisions provided us with the opportunity to collect feedback about the system from different perspectives. The obtained information was recorded and subsequently analyzed in order to understand how MBLogVis influenced the processes at the department and in which areas it could be used to resolve the problems mentioned above. The most important feedback about MBLogVis was classified into a number of categories:

- While MBLogVis gives possibility to widen the scope of the analysis conducted at the department, it cannot completely replace other tools. Additional functionality is available in those tools and currently support other parts of the business process of maintenance. Nevertheless, interviewees agreed that use of MBLogVis as an additional tool can accelerate analysis of log files.
- Interviewees expressed the need for an additional functionality for capturing log-files in real-time from MBM devices in operation. At the moment the developed tool cannot be used for logs generated at real-time since it requires presence of log files before execution.
- The library of patterns will evolve as new errors emerge and might not be exhaustive and therefore absence of failures in a report does not guarantee error-free system.
- Since the library of patterns is voluminous, filling it requires collaboration of a relatively large group of experts thus introducing need for more extensive collaboration among the experts.

- Filling the library of patterns by experts removes the need for high qualification from its users. This opens an opportunity to use the knowledge of experts throughout a global company even in places where highly-qualified specialists are of limited supply.
- Since visualizations can be easily configured using scripts, adjusting visualizations for particular needs becomes a straightforward task.
- Visualizations that show only generic information to users can be employed for communication with OEM's without introducing the problem of compromising network security by disclosing sensitive information.
- Interviewees emphasized that the ability to identify transitions in log files and showing them in state diagrams are considerable advantages for log analysis. Besides, referencing parts of log-files that are responsible for those transitions opens a possibility to extract only the relevant parts from log-files. Therefore, it resolves the problem of using other tools for finding error patterns in large files obtained from OEMs.

The most important feedback about the gadget is:

- Interviewees appreciated the gadget's potential applicability to work with logs generated at real-time and its ability to immediately notify about potential problems in the system. Therefore, subsequent log generation can be aborted when failures occur early during operation of MBM devices since the already obtained information is sufficient for analysis.
- Interviewees expressed their satisfaction with the ability to deploy the gadget on different machines to show the state of the same MBM device in operation. Therefore, it enables simultaneous monitoring by several engineers, thus decreasing turnaround time.
- Deployment of the gadget on test machines with MBM devices enables immediate error notifications. Therefore, testing process can be accelerated since information about an error will be received as soon as error occurs. Additionally, it enables engineers to concentrate testing efforts on machines where the gadget displayed errors.

7.3 Quantitative Assessment

In order to broaden the subject base we complemented qualitative assessment, which can be interpreted in different ways and analysis of results is generally more labor-intensive [91], with a quantitative assessment. The results of using this method supplement the results of qualitative research, which could be considered "unclear" or "ambiguous" in the software engineering industry since engineers and practitioners require accurate justification of research results summarized in a short list of statements or measurements.

The survey method was chosen as the most suitable for the purposes of quantitative assessment of this research project's results. We have considered pre- and post-case study measurements but due to the time constraints imposed on the current research by Ericsson, testing the developed application for real work at the department became outside of the scope thus making measurements of characteristics of the application unfeasible.

In the process of survey creation we used the guidelines provided by Mark Kasunic [93]. His work, published by the Software Engineering Institute, contains the list of best practices for researchers who need to conduct surveys in the software engineering area. Nevertheless, while Kasunic [93] describes survey as a separate research method, we used survey as a component of our action research. The survey helped us to collect information about the extent to which the implemented application can be helpful in resolving the problems mentioned above.

The survey had an objective to obtain quantitative interpretations of qualitative characteristics that help researchers in validation of the MBLogVis application in accordance with stated requirements. Additionally, open-ended questions also help understanding the prospects for further usage of MBLogVis.

Since the number of potential respondents was relatively small – 13 people – the goal was to cover most of them. The reason is the fact that the entire group of respondents needed to be included in the sample to achieve a desirable level of precision in situations when the number of potential respondents is less then 200 [93]. The target group for the survey was the same as for the interviews and their responsibilities vary from system design and development to customer support. However, the common characteristic among them is the need to analyze system log-file in daily work meaning knowledge and experience in log analysis.

Objectives and characteristics of the target group became the basis for the questionnaire. Mostly, survey questions ask about recipient's attitude or beliefs [93] and provide an opportunity to assess the MBLogVis application, which was not in experimental use yet, by collecting perceptions about it. We deliberately used close-ended questions with provided answer choices since answers to this type of questions can be easily assessed by assigning numerical values. The survey consisted of 20 questions, among them 19 were close-ended and one was open-ended ones. The paper-based method of surveying was selected for the current study, since it enables receiving immediate answers from respondents and is least time demanding. We gave the survey to every respondent in person and were available for questions during surveying to minimize the risk of misunderstanding. The pilot test questionnaire was filled by representatives of the DD and COEM divisions and the results were used for improvement of the survey quality. The possible answers to closed-ended questions are stated in the Table 1.

Response	Scale
Strongly disagree	1
Disagree	2
Undecided	3
Agree	4
Strongly agree	5

Table 1. Possible answers to close-ended questions.

Presenting answers as opinions enabled avoiding their possible ambiguity and misinterpretation from respondents' side.

1	You often need to analyze log-files.
2	Files that you need to analyze are often large (over 20 MB).
3	You experience significant problems with analysis of log-files.

4	You believe that visualization of log files could help you to resolve log analysis problems.
5	You often use special tools for log analysis.
6	You often experience problems with log analysis tools.
7	You often experience problems searching for error causes using log analysis tools.
8	You often experience problems searching for text patterns in log-files using log analysis tools.
9	You often experience problems working with large files using log analysis tools.
10	You often use visualizations generated by log analysis tools.
11	You experience significant problems generating visualizations using log analysis tools.
12	You believe that the visualization types in MBLogVis are appropriate for log analysis.
13	You believe that adjusting visualizations in MBLogVis by editing scripts is convenient and practical for log analysis.
14	You believe that pattern-based text search in MBLogVis could be helpful for you in analysis of log-files.
15	You believe that the visualization types available in MBLogVis could be helpful for you especially in analysis of large files.
16	Visualizations help you to focus on the content rather than on the "syntax" of log-files.
17	How much time do you believe could be saved by using MBLogVis in addition to the tools that you use today (in %)?
18	Which features are missing in MBLogVis but would be important for your work?
19	You believe that the gadget will be helpful for real-time testing.
20	How often, from your point of view, can the gadget be used in your daily work assignments?

Table 2. Survey.

The results received are presented in the Tables 3, 4 and 5.

#	Strongly Disagree	Disagree	Undecided	Agree	Strongly Agree
1	0%	0%	0%	31%	69%
2	0%	0%	24%	38%	38%
3	0%	15%	23%	54%	8%
4	0%	0%	0%	62%	38%
5	7%	31%	0%	31%	31%
6	0%	31%	23%	31%	15%
7	0%	0%	46%	46%	8%
8	8%	23%	15%	54%	0%
9	8%	0%	8%	69%	15%
10	23%	38%	23%	8%	8%
11	0%	8%	46%	15%	31%
12	0%	0%	8%	75%	17%
13	0%	17%	25%	50%	8%
14	0%	0%	8%	75%	17%
15	0%	0%	0%	75%	25%
16	0%	0%	25%	50%	25%
19	0%	0%	33%	50%	17%

Table 3. Survey results.

#	<10	10-19	20-29	30-39	40-49	50-59	60-69	70-79	80-90	>90
17	0%	8%	25%	17%	8%	25%	8%	0%	8%	0%

Table 4. Results for the question #17.

#	Never	Rarely	Sometimes	Very Often	Always
20	8%	26%	50%	8%	8%

Table 5. Results for the question #20.

The details of survey analysis and conclusion are described in the section 8 of this paper.

7.4 Turnaround Time of Testing

From the development perspective introduction of MBLogVis decreases turnaround time and therefore improves the quality of the testing process – which was the feedback obtained from the interviews described in Section 7.1. For example, the tools that are currently in use can only test systems on the network level and have the turnaround time around several days while the typical turnaround time during earlier development stages is minutes or hours.

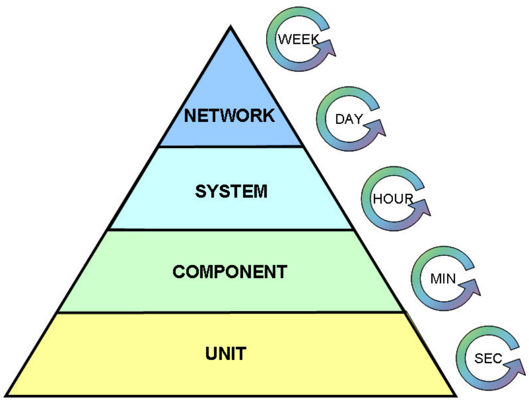

Figure 15. Turnaround time.

Therefore, the MBLogVis application enables much shorter feedback loops thus making faster turnarounds enabling higher quality of features under testing. Additionally, it was mentioned by employees who have verification of MBM modules as the main responsibility that earlier and more focused testing creates new opportunities for decreasing time to market and increasing quality of products.

8. SPECIFYING LEARNING

At the Section 4 we described the problems that were found at the studied department. As it was said, the development of MBLogVis had an intention to resolve those problems by applying visualization techniques to the process of log-files analysis. In this Section we will describe how different problems were resolved using the MBLogVis application. We will present the analysis of qualitative and quantitative feedback received from the employees with respect to the initial problems.

8.1 Analysis of Interview Results

In order to assess achievement of the goals stated for this research, we analyzed the initial problems together with the impact of the MBLogVis on them and received the following results:

- It was mentioned in the interviews that the ability to fill the library of patterns by experts removes the need for high qualification from users of the MBLogVis application. This means that presence of experts during log analysis is no longer required and can be replaced by use of the developed tool with the already predefined library of patterns. Moreover, in our case MBLogVis supports sharing trees of patterns, which helps to quickly fill the patterns library.
- Interviewees emphasized the ability to reference events that are responsible for transitions as a considerable advantage for log analysis. Therefore, the tool enables referencing events, which resolves the issue of finding exact areas in large files. Additionally, the interviewees expressed their contentment in the tool's functionality of identifying transitions in log-files and showing them in state diagrams. This is an evidence of applicability of the tool for locating faulty state changes.
- Interviewees were unanimous in their opinion that the problem of high turnaround time caused by sending large files as well as by receiving files without information about failures can be addressed by applying the gadget for work with logs generated at real-time on OEM's side. Additionally, it would minimize the amount of time needed for log generation on OEM's side because subsequent log generation can be aborted when failures occur early during operation since the already obtained information is sufficient for analysis.
- Interview results state that visualization types showing only generic information can be employed for communication with OEMs without any concern about disclosing sensitive information. Additionally, the gadget and generic visualization types can also be used for performing initial analysis of log-files on OEM's side.

The problem of time-consuming search in the industry standards was not addressed by the developed tool since it was not among the goals for this circle of the action research

8.2 Analysis of Survey Results

The survey was performed among 13 representatives of the DD and COEM divisions, who have analysis of log-files among their work assignments and therefore are potential users of the developed tool. Since [93] describes Likert response scale as one of the most appropriate ways for determining the intensity of a respondent's belief and explicitly prohibits combining and manipulating the numbers assigned to scales to form averages, the response information was reported only as frequencies within each category based on percentage of respondents who hold the same opinion.

The survey confirmed strong need to analyze log-files among all the employees of the studied department. Furthermore, over 75% of respondents stated the need to analyze large files. However, the survey revealed the difference between the divisions in the way their employees analyze log files. Figures 16 and 17 show that employees in the DD generally do not use special tools for log analysis while the COEM division uses them to higher extent. This observation also correlates with the reported frequency of problems with log analysis tools, which is also shown in Figures 16 and 17. Therefore, COEM employees, who often use special tools, experience problems relatively often, while the frequency of these problems is insignificant for the DD employees, who use these tools less often.

Therefore, it was reasonable to hypothesize that the problems at the studied department were caused by the tools that are currently in use. In order to confirm this hypothesis the reported frequency of problems with analysis was compared to the reported frequency of problems with the tools: 62% of respondents agreed or strongly agreed that they experienced problems with analysis of log-files and 46% of respondents experienced problems with log analysis tools.

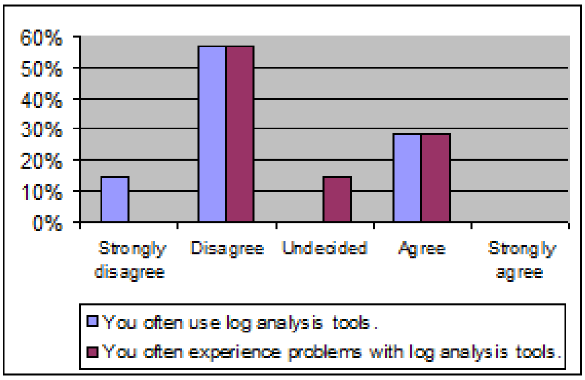

Figure 16. Usage of the special tools in the DD.

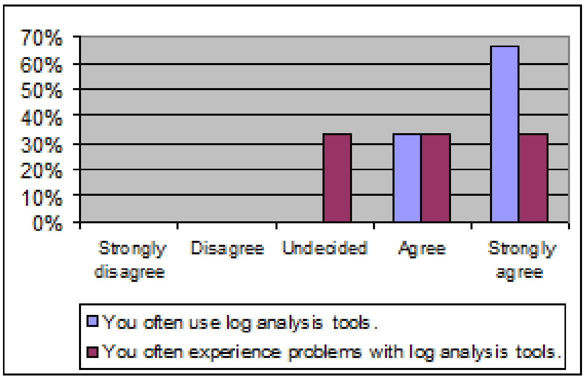

Figure 17. Usage of the special tools in the COEM.

Another important conclusion was drawn from the evaluation of the respondents' belief that visualization can help in resolving log analysis issues. The results show that all the respondents agree with this statement and around 40% have strong agreement. Additionally, survey has indicated that the majority of respondents often do not use visualizations generated by log analysis tools and over 50% experience problems generating visualizations using those tools. This fact shows limited use of the tools that are currently available at the studied department caused by low level of satisfaction. Therefore, it becomes clear that the need for the visualization tools that are applicable to log analysis was still not met by real tools at the time the research started.

Analysis of the respondents' vision of the potential applicability of the developed tool revealed the strong belief from their side that the visualizations available in the tool will be useful for log analysis. As it can be seen from the survey results, the respondents agreed upon the appropriateness of the implemented visualization types, convenience of use pattern-based text search and applicability of MBLogVis for analysis of large files. Figures 18 and 19 show that the majority of respondents experienced difficulties with these aspects of log analysis but also agreed that MBLogVis has high potential for addressing them. Therefore, the problems in work with pattern-based text search and analysis of large files, which are currently present in other tools, can be resolved by the MBLogVis application.

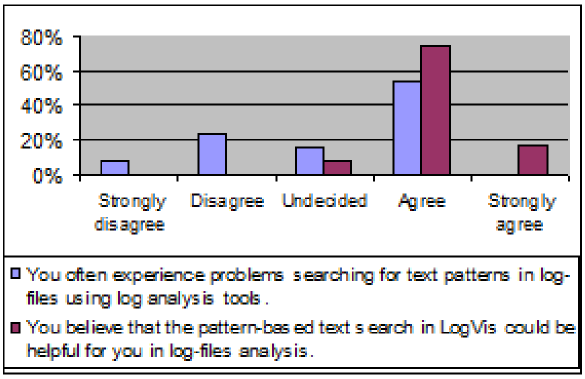

Figure 18. The potential of using MBLogVis for pattern-based text search.

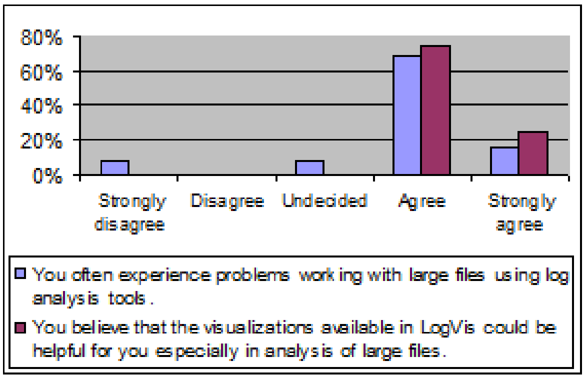

Figure 19. The potential of using MBLogVis for work with large files.

The functionality of visualizations' adjustment in the developed tool by editing scripts received appreciation from the respondents. The assumption that it would be useful and convenient was not supported by 17% of respondents. However, their opinion was clarified further among the improvement suggestions, where they proposed the features that would make scripting in the developed tool more convenient. For example, a special form was suggested for entering key values of the script parameters or a wizard for generating scripts from log-files. These suggestions were well-received by employees of the COEM division, who are generally not programmers and prefer building script-files rather than writing them in text fields.

Summarizing opinions of the employees' at the studied department, it becomes clear that 75% believe that visualizations built by the developed tool enable focusing on the content of log-files rather than on the syntax. This is especially important for work with large files when distinguishing problematic parts becomes impossible without abstracting the irrelevant information away from the essential data. With regards to the increase in effectiveness of log-files analysis, over 90% of respondents agreed that they could save minimum 20% of the time they generally spend on this task, which means that the approach proposed by us potentially enables more efficient processes contributing to higher company profits. One of target areas for improvements can be the *Problem and Modification Analysis* activity of the maintenance process.

The respondents' view of the gadget's applicability for log analysis process revealed strong interest in its usage for real-time testing. Improvement of in-house real-time testing and acceptance testing on the OEM side will potentially lead to acceleration of the *Maintenance Review/Acceptance* activity of the maintenance process.

Since the employees of the DD and COEM divisions, who were the respondents for the survey, generally do not have testing responsibilities, they would "sometimes" use the gadget in daily work assignments. Therefore, the degree to which the gadget can be useful and can possibly decrease an effort for *Maintenance Review/Acceptance* activity will become clear only after applying it to the current real-time testing processes, which was outside the scope of the current study.

Therefore, the overall impact of the tool's and gadget's use in the process of software maintenance potentially leads to reduction of efforts in two of the three core activities, namely *Problem and Modification Analysis* and *Maintenance Review/Acceptance*, which is shown in Figure 20.

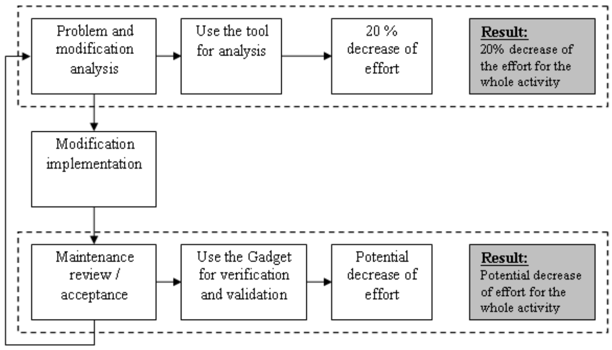

Figure 20. Impact of MBLogVis and the gadget on the maintenance process.

Taking into account future growth of software complexity, which will lead to increased need of maintenance and higher rate of post-release defects, improvement of the iterative core part of the maintenance process will become even more crucial.

9. CONCLUSIONS

The goal of the research presented in this thesis was to improve the processes of integration and maintenance of software for embedded broadband modules. Based on the analysis of identified problems, implementation and evaluation of the MBLogVis application and the gadget we can conclude the following:

- visualizations from MBLogVis combined with the gadget constitute potentially the most efficient improvement for communication between company and its customers;
- "hiding" sensitive information in visualizations is possible in such a way that it enables communication without introducing security issues;
- applicability of the visualization tool and gadget for different stages of software development process enables further improvements.

Evaluation of the influence caused by the improvement in the company business processes requires reviewing the initial problem from the maintenance perspective. Since *Process Implementation*, *Migration* and *Retirement* occur only once during long-term maintenance [95], optimizing them would not provide significant improvement for the overall maintenance process. However, reduction of time spent on log-files analysis, which 90% of respondents estimated as minimum 20%, accelerates the activity of *Problem and Modification Analysis* and leads to multiple improvements for the whole maintenance process proportionally to the number of maintenance iterations.

The other 80% of time spent during the *Problem and Modification Analysis* activity cover communication, reproducing failure in the testing environment, detailed analysis of events represented by error lines in a log-file and preparing a description of a failure for developers. To accomplish most of these steps employees need fixed amounts of time, which cannot be improved by use of additional tools or methods. For example, time spent on detailed analysis of events depends on the competence level of employees, which cannot be addressed by tools either. Therefore, the rest 80% of time spent on the *Problem and Modification Analysis* activity cannot be improved or automated by use of additional tools.

10. FUTURE WORK

The second circle of the conducted action research will concentrate on the improvement of the maintenance review/acceptance activity of the maintenance process. Functionalities of the MBLogVis application and the gadget will be extended and the impact of these improvements will be measured by calculating the decrease in time spent on problem, decrease in number of faults discovered after release and modification analysis and maintenance review/acceptance activities.

Based on the knowledge and feedback obtained at the first circle of the research, further development and enhancement will include:

- Work with real-time logs, which was the most important requirement mentioned by the respondents for further use of the MBLogVis application and gadget in testing.
- Collecting log files from devices, which will remove the need to use additional tools and therefore simplify use of MBLogVis.
- Wizard for script generation with possibility to use logged events, which will enable saving time on training employees before using the tool.
- Use of multiple files in the gadget and accumulation of statistics, which will facilitate use of the gadget in situations of more complex testing.

ACKNOWLEDGEMENTS

First and foremost, we would like to thank Ericsson's MBM department for organizing this research, especially Henric Bergenwall, Erika Asp, Anders Josefsson and Martin Edofsson. Without them being ready to discuss our findings, to help us contact different persons at Ericsson, to tirelessly read our reports and to provide us with prompt and extensive feedback this research would not be possible. Besides, we would like to thank many others from Ericsson for providing us constructive ideas and feedback and for spending their time explaining technical details to us.

Moreover, we wish to use this opportunity to express our profound gratitude to Dr. Miroslaw Staron, who patiently, thoroughly and wisely guided us through the entire research.

REFERENCES

[1] *Telecoms software: worldwide forecast 2010–2014.* Analysys Mason Limited, 2010.

[2] Schipper, I. and De Haan, E.2005. *CSR Issues in the ICT Hardware Manufacturing Sector*. SOMO

[3] Valdman, J. 2003. *Log File Analysis*, Phd Report, Plzen, Czech Republic.

[4] Jansen, B.J. 2006. Search log analysis: What it is, what's been done, how to do it. *Library & Information Science Research*, 28(3), 407–432.

[5] Ozmutlu, H.C., Ozmutlu, S. and Spink, A. 2002. Analysis of large data logs: an application of Poisson sampling on excite web queries. *Information Processing and Management*, 38(4), 473-490.

[6] Zhao, Q. and Cao, T. 2009. Collecting Sensitive Information from Windows Physical Memory. *Journal of Computers*, 4(1), January 2009.

[7] Zawawy, H., Kontogiannis, K. and Mylopoulos J. 2010. Log filtering and interpretation for root cause analysis. *IEEE International Conference on Software Maintenance*, ISSN 1063-6773, 2010, 1–5.

[8] Frie, A. and Rennhard, M. 2008. Histogram Matrix: Log File Visualization for Anomaly Detection. *Proceedings of the Third International Conference on Availability, Security and Reliability*, 2008, 610-617.

[9] Tory, M. and Möller, T. 2004. Human Factors in Visualization Research. *IEEE Transactions on Visualization and Computer Graphics*, 10(1), 72-84, January/February 2004.

[10] Rabkin, A., Xu, W., Wildani, A., Fox, A., Patterson, D. and Katz, R. 2010. A graphical representation for identifier structure in logs. *SLAML 2010*, Vancouver, Canada, October 2010.

[11] Doyle, L. 1961. Semantic roadmaps for literature searchers. *Journal of the Association for Computing Machinery*, 8(4), 367–391.

[12] Risch, J., Kao, A., Poteet, A. and Wu, Y.J. 2008. Text Visualization for Visual Text Analytics. *Visual Data Mining*, 2008, 154-171

[13] Wise, J.A., Thomas, J.J., Pennock, K., Lantrip, D., Pottier, M., Schur, A. and Crow, V. 1995. Visualizing the non-visual: Spatial analysis and interaction with information from text documents. *Proceedings of the IEEE Information Visualization Symposium InfoVis 1995*, 51-58.

[14] Barringer, H., Groce, A., Havelund, K. and Smith, M. 2010. Formal Analysis of Log Files. *Journal of Aerospace Computing, Information, and Communication*, 7(11), 365-390.

[15] Eick, S.G., Nelson, M.C. and Schmidt, J.D. 1994. Graphical Analysis of Computer Log Files. *Communications of the ACM*, 37(12), 50-56, December 1994.

[16] Eick, S.G., Steffen, J.L. and Sumner, E.E.Jr. 1992. Seesoft™—A Tool For Visualizing Line Oriented Software Statistics. *IEEE Transactions On Software Engineering*, 18(11), 957-968, November 1992.

[17] Ball, R., Fink, G. and North, C. 2004. Home-Centric Visualization of Network Traffic for Security Administration. *ACM Conference on Computer and Communications Security, Workshop on Visualization and Data Mining for Computer Security 2004*, 55-64.

[18] Hochheiser, H. and Shneiderman, B. 2001. Using Interactive Visualizations of WWW Log Data to Characterize Access Patterns and Inform Site Design. *Journal of the American Society for Information Science and Technology*, 52(4), 331–343, 2001.

[19] Pitkow, J. and Bharat, K. 1994. Webviz: A tool for world wide web access log analysis. *Proceedings of First International Conference on the World Wide Web*.

[20] Takada, T. and Koike, H. 2002. MieLog: A Highly Interactive Visual Log Browser Using Information Visualization and Statistical Analysis. *Proceedings of the 16th Systems Administration Conference*, 2002, 133-144.

[21] Makanju, A., Brooks, S., Zincir-Heywood, A.N. and Milios, E.E. 2008. LogView: Visualizing Event Log Cluster. *Sixth Annual Conference on Privacy, Security and Trust*, 2008, 99-108.

[22] Kioke, H. and Ohno, K. 2004. SnortView: Visualization System of Snort Logs. *CCS Workshop on Visualization and Data Mining for Computer Security*, ACM Press, October 2004, 143 – 147.

[23] Lam, H., Russell, D., Tang, D. and Munzner, T. 2007. Session viewer: Visual exploratory analysis of web session logs. *IEEE Symposium on Visual Analytics Science and Technology*, IEEE Press, November 2007, 147 – 154.

[24] Ma, S. and Hellerstein, J.L. 1999. Eventbrowser: A flexible tool for scalable analysis of event data. *Proceedings of the 10th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management*, Springer-Verlag, 1999, 285– 296.

[25] Eick, S.G. 2000. Visual Discovery and Analysis. *IEEE Transactions on Computer Graphics and Visualization*, January-March 2000, 6(1), 44-59.

[26] Erbacher, R.F., Walker, K.L. and Frincke D. 2002. Intrusion and Misuse Detection in Large-Scale Systems. *IEEE Computer Graphics & Applications*, 22 (1), 38-48.

[27] Erbacher, R.F. and Garber M. 2004. Fusion and Summarization of Behavior for Intrusion Detection Visualization. *Proceedings of the IASTED International Conference on Visualization, Imaging, and Image Processing*, September 2004, 159-165.

[28] Erbacher R.F. 2003. Intrusion Behavior Detection Through Visualization. *Proceedings of the IEEE Systems, Man and Cybernetics Conference*, October, 2003, 2507-2513.

[29] Erbacher, R.F., Teng, Z. and Pandit, S. 2002. Multi-Node Monitoring and Intrusion Detection. *Proceedings of the IASTED International Conference on Visualization, Imaging, and Image Processing*, September 2002, 720-725.

[30] Girardin, L. and Brodbeck D. 1998. A Visual Approach for Monitoring Logs. *Proceedings of the 12th Systems Administration Conferences*, December 1998, 299-308.

[31] Goldstein, J., Roth, S.F., Kolojejchick, J. and Mattis. J. 1994. A framework for knowledge-based, interactive data exploration. *Journal of visual languages and computing*, 5, 339–363.

[32] Kasemsri, R.R. 2006. *A Survey, Taxonomy, and Analysis of Network Security Visualization Techniques*. Computer Science Theses, Paper 17, Georgia State University, USA.

[33] Wong, D.H.-T. and Ramadass, S. 2010. An Automated Algorithm In Data Visualization For Large Network Data: A Review And An Initial Study. *IJSSST*, 11(2), April 2010.

[34] Erbacher, R. F. 2001. Visual Behavior Characterization for Intrusion Detection in Large Scale Systems. *Proceedings of the IASTED International Conference on Visualization, Imaging, and Image Processing*, September 2001, 54-59.

[35] Axelsson, S. 2004. Visualising Intrusions: Watching the Webserver. *Proceedings of the 19th IFIP International Information Security Conference*, 259-274.

[36] Goldring, T. 2004. Scatter (and other) Plots for Visualizing User Profiling Data and Network Traffic. *Proceedings of the 2004 ACM Workshop on Visualization and Data Mining for Computer Security*, ACM Press, 119-123.

[37] Koike, H. and Ohno, K. 2004. SnortView: Visualization System of Snort Logs. *Proceedings of the 2004 ACM Workshop on Visualization and Data Mining for Computer Security*, ACM Press, 143-147.

[38] Muelder, C., Ma, K. and Bartoletti T. 2004. A Visualization Methodology for Characterization of Network Scans. *IEEE Workshop on Visualization for Computer Security*, VizSEC 2005, 29-38.

[39] Colombe, J. and Stephens, G.2004. Statistical Profiling and Visualization for Detection of Malicious Insider Attacks on Computer Networks. *Proceedings of the 2004 ACM Workshop on Visualization and Data Mining for Computer Security*, ACM Press, 138-142.

[40] Axelsson, S. 2004. Combining a Bayesian Classifier with Visualisation: Understanding the IDS. *Proceedings of the ACM CCS Workshop on Visualization and Data Mining for Computer Security*, ACM Press, 99-108.

[41] Axelsson, S. 2004. *Visualising the Inner Workings of a Self Learning Classifier: Improving the Usability of Intrusion Detection Systems*. Technical Report, Department of Computing Science, Chalmers University of Technology, Gothenburg, Sweden.

[42] McPherson, J., Ma, K., Krystosk, P., Bartoletti, T., and Christensen, M. 2004. PortVis: A Tool for Port-Based Detection of Security Events. *Proceedings of the 2004 ACM Workshop on Visualization and Data Mining for Computer Security*, ACM Press, 73-81.

[43] Teoh, S.T., Ma, K.L., Wu, S.F. and Zhao, X. 2002. A Visual Technique for Internet Anomaly Detection. *Proceedings of IASTED International Conference on Computer Graphics and Imaging*.

[44] Atkison T., Pensy, K., Nocholas, C., Ebert, D., Atkison, R. and Morris, C. 2001. Case Study: Visualization and Information Retrieval Techniques for Network Intrusion Detection. *Proceedings of the Third IEEE Eurographics Visualization Symposium*, Springer, May 2001, 113-120.

[45] Scott, C., Nyarko, K. Capers,T. and Ladeji-Osias, J. 2003. Network Intrusion Visualization with NIVA, an Intrusion Detection Visual Analyzer with Haptic Integration. *Journal Information Visualization*, June 2003, 2(2), 82-94.

[46] Erbacher, R.F. and Grinstein, G.G. 1995. Issues in the Development of 3D Icons. *Visualization in Scientific Computing*, Springer-Verlag, 109- 123.

[47] Axelsson, S. 2003. Visualisation for Intrusion Detection: Hooking the Worm. *Proceedings of the 8th European Symposium on Research in Computer Security*. Springer-Verlag, 2003, 309-325.

[48] Yin, X., Yurcik, W., Treaster, M., Li, Y. and Lakkaraju, K. 2004. VisFlowConnect: NetFlow Visualizations of Link Relationships for Security Situational Awareness. *Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security*, 26-34.

[49] Conti, G. and Abdullah, K. 2004. Passive Visual Fingerprinting of Network Attack Tools. *Proceedings of the 2004 ACM Workshop on Visualization and Data Mining for Computer Security*, ACM Press, October 2004.

[50] Fortier, S.C. and Shombert, L.A. 2000. Network Profiling and Data Visualization. *Proceedings of IEEE Systems Man and Cybernetics Society Information Assurance and Security Workshop*, 136-142.

[51] Jirapummin,C., Wattanapongsakorn, N. and Kanthamanon, P. 2002. Hybrid Neural Networks for Intrusion Detection System. *Proceedings of the 2002 International Technical Conference on Circuits/Systems, Computers and Communications*.

[52] Labib, K. and Vemuri, R. 2002. NSOM: A Real-Time Network-Based Intrusion Detection System Using Self-Organizing Maps, *Networks and Security*, 2002.

[53] Yoo, I. 2004. Visualizing Windows Executable Viruses Using Self-Organizing Maps. *Proceedings of the 2004 ACM Workshop on Visualization and Data Mining for Computer Security*, ACM Press, 82-89.

[54] Xin, J., Dickerson, J.E. and Dickerson, J.A. 2003. Fuzzy Feature Extraction and Visualization for Intrusion Detection. *The 12th IEEE International Conference on Fuzzy Systems*, 2, 1249-1254.

[55] Smith, M. and Havelund, K. 2008. Requirements Capture with RCAT. *Proceedings of the 16th IEEE International Requirements Engineering Conference*, IEEE Computer Society, 183–192.

[56] Havelund, K. 2008. Runtime Verification of C Programs. *Proceedings of the 1st TestCom/FATES Conference*, Springer, 7–22.

[57] Mansmann, F., Keim, D., North, S., Rexroad, B. and Sheleheda, D. 2007. Visual Analysis of Network Traffic for Resource Planning, Interactive Monitoring and Interpretation of Security Threats. *IEEE Transactions on Visualization and Computer Graphics*, December 2007, 13(6), 1105–1112.

[58] Minarik, P., Dymacek, T., Goodall, J.R., Conti, G. and Ma, K.L. 2008. *NetFlow Data Visualization Based on Graphs, VizSec 2008*, Springer-Verlag, 144–151.

[59] Chang, F. and Ren, J. 2007. Validating System Properties Exhibited in Execution Traces. *Proceedings of the 22nd IEEE/ACM International Conference on Automated Software Engineering*, ACM Press, 517–520.

[60] Bearavolu, R., Lakkaraju, K., Yurcik, W. and Raje, H. 2003. A Visualization Tool for Situational Awareness of Tactical and Strategic Security Events on Large and Complex Computer Networks. *IEEE Military Communications Conference*, 2, 850-855.

[61] Teoh, S.T., Jankun-Kelly, T.J, Ma, K.-L. and Wu, S.F. Visual Data Analysis for Detecting Flaws and Intruders in Computer Network Systems. *IEEE Computer Graphics and Applications, special issue on Visual Analytics*, 2004.

[62] Toelle, J., Niggemann, O. 2000. Supporting Intrusion Detection by Graph Clustering and Graph Drawing. *Proceedings of Third International Workshop on Recent Advances in Intrusion Detection RAID 2000*, Springer-Verlag, October 2000.

[63] Kitchenham, B., Pfleeger, S.L., Pickard, L.M., Jones, P.W., Hoaglin, D.C., Emam, K.E. and Rosenberg, J. 2002. *Preliminary Guidelines for Empirical Research in Software Engineering. Journal of IEEE Transactions on Software Engineering*, 28(8), 721-734.

[64] Perry, D.E., Porter, A.A. and Votta, L. G. 2000. Empirical studies of software engineering: a roadmap. *The Future of Software Engineering, ACM Press*, May 2000, 345-355.

[65] Creswell, J.W. 1994. *Research Design, Qualitative and Quantitative Approaches*, Sage.

[66] Lau, F. 1999. Toward a framework for action research in information systems studies. *Information Technology & People*, Emerald, 12(2), 148-175.

[67] Susman, G. and Evered, R. 1978. An assessment of the scientific merits of action research. *Administrative Science Quarterly*, 23, 582-603.

[68] O'Brien, R. 1998. *An overview of the Methodological Approach of Action Research*. Faculty of Information Studies, University of Toronto, Toronto, Canada.

[69] Baskerville, R. and Pries-Heje, J. 1999. Grounded action research: A method for understanding it in practice.

Accounting, Management & Information Technologies, 9, 1-23.

[70] Avidson, D. 2002. Action Research: A Research Approach for Cooperative Work. *The 7th International Conference on Computer Supported Cooperative Work in Design*, 19-24.

[71] Avison, D., Lau, F., Myers, M.D. and Nielson, P.A. 1999. Action Research. *Communications of the ACM*, 42(1), January 1999, 94-97.

[72] Mathiassen, L. 2002. Collaborative practice research. *IT & People*, 15(4), 321-345.

[73] Gronholm, S. and Goldkuhl, G. Understanding the practices of action research. *ECRM 2003*.

[74] Brydon-Miller, M., Greenwood, D. and Maguire P. 2003. Why action research? *Action Research*, 1, 9-28.

[75] Baskerville, R. and Myers, M.D. 2004. Special issue on action research in information systems: Making is research relevant to practice - foreword. *MIS Quarterly*, 28(3), 329-335.

[76] Sandberg, A., Pareto, L. and Arts, T. 2011. Agile Collaborative Research: Action Principles for Industry-Academia Collaboration. *IEEE Software*, July/August 2011, 28(4), 74-83.

[77] Avison, D., Baskerville, R. and Myers, M. 2001. Controlling action research projects. *Information Technology & People*, 14(1), 28-45.

[78] Davison, R.M., Martinsons, M.G. and Kock, N. 2004. Principles of canonical action research. *Information Systems Journal*, 14, 65-86.

[79] Hendrifsson O., Lindgren R., Action Research in New Product Development. in Kock, N.F. ed. *Information Systems Action Research: An Applied View of Emerging Concepts And Methods.* Springer, Laredo, Texas, USA, 2007, 193-216.

[80] Henfridsson, O. and Lindgren, R. 2005. Multi-contextuality in ubiquitous computing: Investigating the car case through action research. *Information and Organization*, 15(2), 95-124.

[81] Medeiros dos Santos, P.S. and Travassos, G.H. 2009. Action Research Use in Software Engineering: an Initial Survey. *Proceedings of the Third International Symposium on Empirical Software Engineering and Measurement*, 414-417.

[82] *Fourth Quarter Report, Q4, 2011*. Retrieved March 6, 2012, from Ericsson: http://www.ericsson.com/news/1579912

[83] Rice, R.E. and Borgman, C.L. 1983. The use of computer-monitored data in information science. *Journal of the American Society for Information Science*, 44, 247−256.

[84] Fry, B. *Visualizing Data*. O'Reilly Media Inc., 2007

[85] *IEEE Std.* 610.12-1990

[86] *IEEE Std.* 830-1998

[87] Bass, L., Clements, P. and Kazman, R. *Software architecture in practice*. Addison-Wesley, Boston, 2003.

[88] Karahasanoviæ, A., Anda, B., Arisholm, E., Hove, S.-E., Jørgensen, M., Sjøberg, D.K. and Welland, R. Collecting feedback during software engineering experiments. *Empirical Software Engineering*, April 2005, 10(2), 113–147.

[89] Runeson, P. and Höst, M. 2009. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, April 2009, 14(2), 131–164.

[90] Hove, S.E. and Anda, B. 2005. Experiences from conducting semi-structured interviews in empirical software engineering research. *11th IEEE International Software Metrics Symposium*, 10–23.

[91] Seaman, C. 1999. Qualitative Methods in Empirical Studies of Software Engineering. *IEEE Transactions on Software Engineering*, 25, 557-570.

[92] Basili, R. and Weiss, D.M. 1984. A Methodology for Collecting Valid Software Engineering Data. *IEEE Transactions on Software Engineering*, 10(6), 728-738.

[93] Kasunic, M. 2005. *Designing an Effective Survey. Handbook*. CMU/SEI, HB-004, September 2005.

[94] *ISO/IEC Std.* 12207-2008.

[95] *ISO/IEC Std.* 14764-1999.

APPENDIX 1. MBLOGVIS DESIGN.

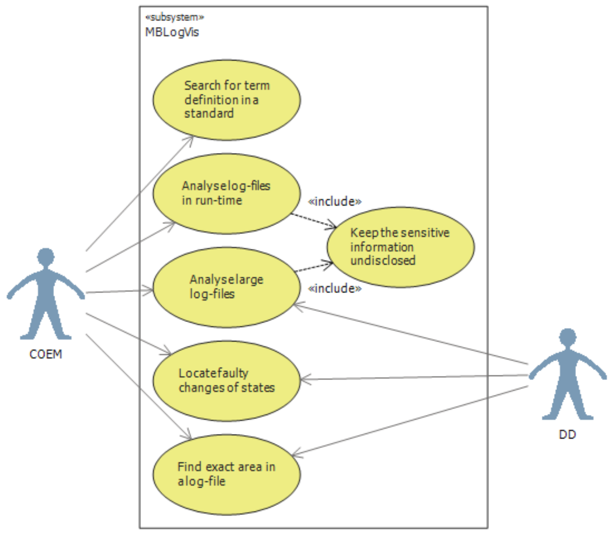

Figure 21. Interaction of users with MBLogVis.

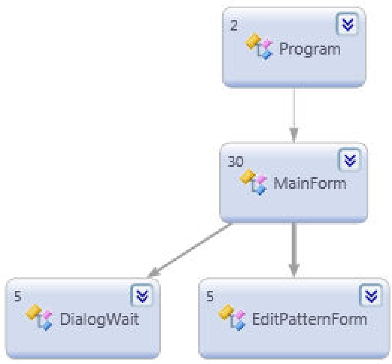

Figure 22. Class dependencies diagram of the MBLogVis tool.

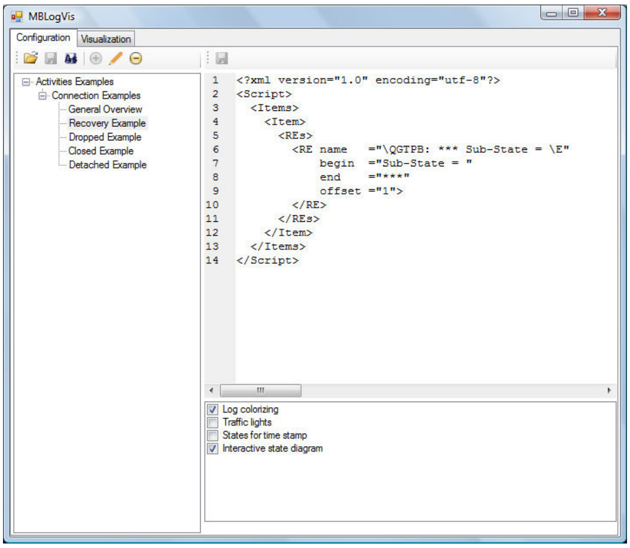

Figure 23. Configuration page of the MBLogVis application.

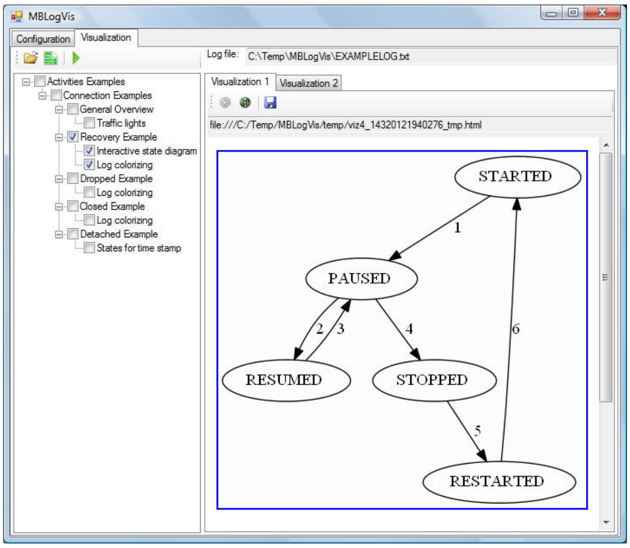

Figure 24. Visualization page of the MBLogVis application.