Visualizing Network Element O&M Data

Petri Tilli

University of Tampere
Faculty of Information Sciences
Department of Computer Science
Interactive Technology
Master's Thesis
Supervisor: Harri Siirtola
May 2010

University of Tampere
Department of Computer Science
Interactive Technology
Author: Petri Tilli
Master's Thesis, 59 pages, 3 appendix pages
May 2010

Monitoring a telecommunication network's performance means managing its operation and maintenance data. That data consists of fault management and performance management fragments.

The aim of this study was to provide new designs for performance and fault management by using information visualization techniques. In earlier network management systems, visualizations could be seen as inadequate or even non-existent. Designs were made based on some well-known information visualization principles and two interactive prototypes were constructed based on couple of designs.

This study covers the designs and evaluation of the designs. Research questions were formulated by Nokia Siemens Networks system architects and the success of this study was measured on how well these designs would answer to the questions asked. For evaluation, a user test was organized with 10 participants.

Based on the user test, the new performance management visualization concept answered the research questions well. The fault management concept had some problems and answered the questions partially.

Both prototypes were accepted as first designs, but further development must be done to make these concepts more useful and practical. Based on the user test, there is a need for information visualization in the network management systems.

Keywords: information visualization, network management, alarm, performance indicator, fault management, performance management, topology

Table of Contents

1. Introduction 1
2. Network management operations and maintenance 3
    - 2.1. Fault and performance management 3
    - 2.2. Use cases of network element O&M 5
    - 2.3. O&M data models and data content 6
        - 2.3.1. Network element's managed object topology 6
        - 2.3.2. Fault management 7
        - 2.3.3. Performance management 8
    - 2.4. Research questions 9
3. Information visualization 11
    - 3.1. Introduction to information visualization 11
    - 3.2. Evaluating visualizations 13
    - 3.3. Visualizing O&M data 15
    - 3.4. Existing O&M visualizations 16
        - 3.4.1. Monitoring versus reporting-like behaviour 16
        - 3.4.2. Fault management 16
        - 3.4.3. Performance management 19
        - 3.4.4. Problems with the existing visualizations 20
    - 3.5. Related work in information visualization 21
4. New O&M data visualization designs 24
    - 4.1. Common design practices 24
        - 4.1.1. Rationale for chosen designs 24
        - 4.1.2. Design proposal for common view and filtering UI 27
        - 4.1.3. Interaction design 28
        - 4.1.4. Prototype implementation 30
    - 4.2. Design proposals for FM 31
        - 4.2.1. Large scale FM view 31
        - 4.2.2. Alarm event view 33
        - 4.2.3. Alarm details view 34
        - 4.2.4. Thoughts about FM concept designs 35
    - 4.3. Design proposals for PM 35
        - 4.3.1. Colour-coding performance indicator instances 35
        - 4.3.2. PM view A: Performance indicator matrix 38
        - 4.3.3. PM view B: Reordered performance indicator matrix 40
        - 4.3.4. Performance indicator details view 41
        - 4.3.5. Thoughts about PM concept designs 42
    - 4.4. Evaluation of the designs 43
        - 4.4.1. User test setup 43
        - 4.4.2. Tester profiles 44

4.4.3. Test execution 45
5. Discussion 47
5.1. Test result analyzation 47
5.2. FM prototype comments and observations 51
5.3. PM prototype comments and observations 53
5.4. Top usability problems based on the test 54
5.5. Results and possible future work items 54
6. Conclusion 58

1. Introduction

Modern telecommunication networks consist not only of telecommunication channels and radio towers but also of many network elements. Network elements are pieces of hardware and software, which all have their specific purposes and roles in the network. The telecommunication channels connect the network elements to each other and to a centralized management system which handles all those elements and their data. A *network management system* (later called as NMS) is this system. It is mainly used by network operators to control and monitor the network behaviour.

Operations and maintenance (later called O&M) is needed for network management to make sure that every component in this live network is working without errors and within certain acceptable parameters.

Some human effort is still required to operate and maintain a telecommunication network. Network elements transfer data to NMS either real-time or by periodical pull. This data has to be interpreted by humans and decisions have to be made according to the data. Visualization of O&M data is something that has been largely ignored at least in some large network management systems, although effective visualization techniques could make O&M-related problems much more easily spottable and therefore the O&M related activities more effective.

The interest for this work was triggered by the realization that there is a lack of information visualization practices especially in performance management side of O&M.

Based on the discussion that I had with Nokia Siemens Networks system architects in winter 2009, some problems with the current O&M data visualization were identified and research questions were formed for this thesis work. Based on the research questions, some visualization techniques were examined and a few visualization designs were created. In addition, based on the designs, two example interactable visualization prototypes were implemented and tested with telecommunication specialists. Finally, based on the prototype user tests and literature examination, conclusions on the success of those designs can be made.

This work is structured as follows: In section two, the network management O&M is presented in more detail for the reader to get a picture of the data and

of the whole domain in question. At the end of the section, the research questions are introduced.

In section three, the information visualization domain is presented. The typical O&M visualizations and related work are also presented.

In section four, new PM and FM visualization designs are introduced. The interaction design, user interface design and the user test are covered as well.

Section five covers the test result analyzation and discussion based on the results.

Section six concludes the work and summarizes the findings.

2. Network management operations and maintenance

One has to make the distinction between *operations and maintenance* (later O&M) data and the actual network element data; O&M data is purely data related to monitoring and handling the network elements state and health and by doing so it makes sure that network element's data is safe and the element itself is functioning properly.

Usually the O&M interfaces are modeled after *FCAPS* which is an ISO Telecommunications Management Network model for a network management [ISO]. FCAPS is an acronym for Fault, Configuration, Accounting, Performance, and Security which are the management categories into which the ISO model splits the network management tasks [FCAPS].

The term *fragment* is used, when speaking about the specific parts of FCAPS model. Fault management is a fragment and performance management is another fragment inside operability. There are other fragments available but these are the two that are of interest here. Section 2.1 presents these fragments based on concepts defined in FCAPS.

2.1. Fault and performance management

Fault management's (later FM) purpose is to detect faults inside network elements and to notify the management system of these faults by generating an *alarm event* and sending it to the management system. Examples of a situation for raising an alarm event could be a hardware failure inside a network element, or a communication problem between network elements. Alarms are always created and raised by a network element. This means that a network element has some monitoring software inside that detects a fault and creates the alarm.

There are four possible operations available for alarms. Firstly, alarms can be *triggered* when a fault is created. Secondly, they can be *cleared* when a fault has been solved. Thirdly, the severity of the alarm can change, which means that an *alarm changed* event is sent. Fourthly, alarms can be *acknowledged* by the operator to indicate that the problem has been noticed and it is currenty under an investigation.

One network element can have hundreds of different monitorable entities that can trigger alarm events. Network management must be real-time aware of these alarms and their severity codes when they are triggered or cleared. Also

in telecommunications network, with tens or hundreds of network elements, a network management system has to be able to hold data of hundreds of thousands of alarms simultaneously. This is partly because a single fault can be detected by an offending component, its parent component or by some other components. This can lead to a net effect of a large number of alarms related to one single cause [Myers et al, 2002]. There can be a set of alarm correlation rules [Burns et al, 2001] as a part of fault diagnosis software which can identify causes for certain alarms. One example of this kind of behaviour model called Symptom-Fault-Action model is covered in a paper written by Yongning Tang et al. [2008]. If there are hundreds of alarms in NMS, identifying the root cause for some alarms is important.

In FM, the concept of *Alarm table* is important. An alarm table is always the representation of the currently active alarms in a NMS. Active alarms are alarms which have been raised or acknowedged. When alarm is cleared, it disappears from the alarm table. By monitoring the alarm table, an operator sees the current situation and can react to certain alarms as soon as they appear.

Performance management (later PM) is used for collecting and analyzing performance data from the network elements. This data is used to evaluate and report the behaviour of the whole network or certain subnetworks. This makes it possible to correct and optimize the behaviour of the network and the elements. The collecting of PM data is called making *measurements*. These measurements can be used to make reports showing the state of the network.

Measurement data is a collection of measurable entities, *performance indicators*, which have numerical values. Each performance indicator (later also referred to as PI) can represent a different type of numerical data. One indicator can be for instance a percentage, the second the amount of memory used in kilobytes and the third the amount of milliseconds since the epoch. Each of these examples has a totally different kind of meaning and numerical space so they are not comparable to each other as such.

Both FM and PM have also the information of what is the entity from where they are collected, a measurable entity. In network management terminology it is called a *managed object*.

A managed object is an abstract representation of the network resource that is managed by the network management system. It can be an actual physical entity, a service, or just an abstraction made for network management. A

managed object can represent anything from a network element to a single piece of hardware, for instance a single memory bank. Managed objects have types, so called managed object classes. The hierarchy of these managed object classes is called the *network topology*. A topology is usually a tree-like hierarchy with several root nodes and their children. It can also be a network-like structure. An example topology is presented in figure 1. The amount of children and levels in this hierarchy are basically unlimited. It is a task for network element O&M engineers in the collaboration of network management system engineers to design the managed objects and the topology in such way that it is meaningful and understandable in the management system.

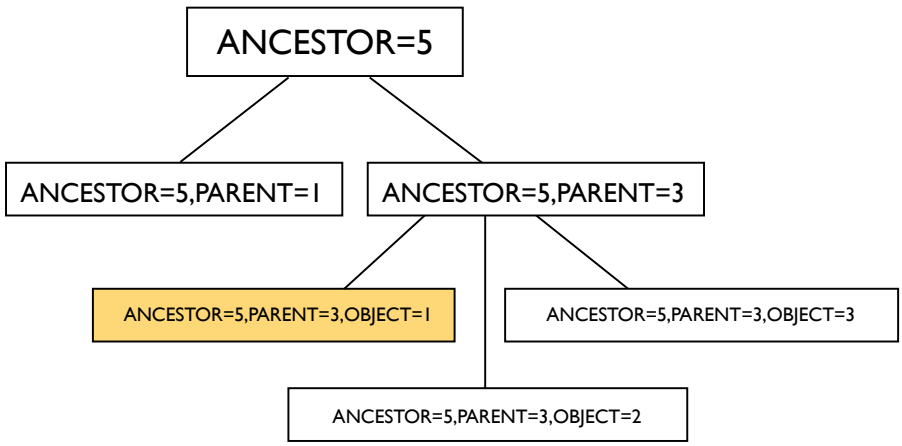

Figure 1. Example of managed object topology

In FCAPS, there are management fragments that this work does not cover, such as accounting, configuration and security. Instead, the focus is kept on fault and performance management and the visualization dilemmas they pose.

2.2. Use cases of network element O&M

PM is mainly used for checking that the hardware and software running in a network element is performing within certain acceptable parameters.

For example, there could be a performance indicator describing the amount of processor usage. By viewing this PI's value, the user can notice if the load is getting too high. This might mean that there are too many processes to handle, which could indicate some malfunction or a design flaw. On the contrary, if the load is too low, it means that there could be resources available and those could be utilized by that machine. A PM's nature is such that it is not traditionally real-time behaviour, but instead is more about measurements which are

collected during specific intervals and collection periods ordered by the network management. Real-time behaviour is currently still missing from PM.

FM covers activities related to overseeing the alarms. Based on a certain alarm, the user or the management system itself has to decide what corresponding actions to take.

One simple example of alarm functionality could be that a hard drive is failing inside a network element. O&M software is monitoring the health of the hard drive and is connected to the network management system. When the failure is detected, the software sends an alarm to the management system via an *O&M interface*. An O&M interface can mean a network protocol, a software API or even shared disk storage. The user of the management system notices the alarm in the alarm table and acknowledges the alarm by sending an acknowledgement message through the O&M interface. Then she creates a hard drive replacement task to the corresponding maintenance crew. When the hard drive is replaced, the alarm is cleared, either by user of the management system or by the O&M software, which notices that there is no longer a fault present.

Certain PM performance indicators can also be related to certain alarms, so when a performance indicator value exceeds a threshold value, there could be an alarm sent to the management system. For a user, an alarm can be a much more visual and clearer indication that something is wrong in the network element. Of course this is a bit irrelevant information for this work, because at the network management level, it cannot be known if the actual alarm was triggered by some performance indicator value or not. This is functionality of the actual O&M software.

2.3. O&M data models and data content

2.3.1. Network element's managed object topology

As mentioned in section 2.1, the unifying glue between fault and performance management fragments is the managed object topology. All alarms are assigned to a managed object and all performance indicators belong to a measurement which has a managed object.

In this work, the only content that managed objects have is *distinguished names*. The object's distinguished name is unique, one cannot find two managed objects with the same name in the management system. The form of network management object's distinguished name is specified in the RFC 1485 [RFC

1485]. The name consists of two parts; the managed object class name and instance identifier (separated with equals '=' symbol). Managed objects can be used to form a hierarchical structure, which can describe the actual network structure. Hierarchy can be represented with names because we can include the information of a specific managed object's parent and ancestors preceeding the actual specific managed object name (separated with the comma ',' symbol).

For example, in the figure 1 presented earlier, we have an object whose name is OBJECT=1. OBJECT=1 is under PARENT=3, which is under ANCESTOR=5. With this kind of hierarchy our object's full distinguished name is ANCESTOR=5,PARENT=3,OBJECT=1. When we refer to objects with this name, we know exactly where the object is in the object hierarchy. A single network element can contain multiple managed objects (An element can contain many machines, machines can have many disks, processors or processes) and a network management system can have a large amount of network elements, so displaying the managed object hierarchy is also an issue for this work.

2.3.2. Fault management

An alarm is an entity which has several attributes which describe it to the management system. Here is an attribute set that is used in this work:

- Distinguished name
- Specific problem code
- Severity
- Time

In real life systems, there could be much more attributes but this is the set which is the most essential in identifying certain alarms.

Following is a description of each attribute.

1. Distinguished name is already covered in the previous section 2.2.1, it is the name of the managed object this alarm refers to.
2. Specific problem code is the identifier of the alarm, it is a number which is reserved for this specific alarm by the network management system.
3. Severity says how important the alarm is, or how severe it is. There are 4 values possible here. From most severe to least, they are: 1. Critical, 2. Major, 3. Minor and 4. Warning.

4. Time is the date and time when the alarm has been raised.

In this work, the acknowledging of alarms is not covered, so there is no attribute telling whether an alarm has been acknowledged or not. Therefore, only alarm raising and clearing are in the scope of this work.

2.3.3. Performance management

A single performance indicator instance has a name, a value, a managed object and measuring time. This is true for visualization purposes. From a metadata perspective, a PI belongs to a managed object class. This means that all managed objects that belong to the same managed object class have the same PIs, but their values are different.

As an example, let us examine the two managed objects, first one being ANCESTOR=5,PARENT=3,OBJECT=1 and the other one being ANCESTOR=5,PARENT=3,OBJECT=2. In this case their managed object class would be ANCESTOR,PARENT,OBJECT and that object class has two performance indicators called memoryUsage and hddSpaceFree. In the example network management system, these PIs are queried once every 15 minutes from a network element and the next query returns the result seen in figure 2.

Distinguished name	Performance Indicator name	Performance Indicator value
ANCESTOR=5,PARENT=3,OBJECT=1	memoryUsage	76
ANCESTOR=5,PARENT=3,OBJECT=1	hddSpaceFree	12764
ANCESTOR=5,PARENT=3,OBJECT=2	memoryUsage	34
ANCESTOR=5,PARENT=3,OBJECT=2	hddSpaceFree	456764

Figure 2. Example measurement, made 12:30 PM

As a result, we have now made a measurement which includes 4 performance indicators in total. These PIs represent a specific point in time, in this example 12:30PM. Measurements happen periodically so when the next measurement is made, the performance indicator values might have changed, as presented in figure 3.

Distinguished name	Performance Indicator name	Performance Indicator value
ANCESTOR=5,PARENT=3,OBJECT=1	memoryUsage	34
ANCESTOR=5,PARENT=3,OBJECT=1	hddSpaceFree	12340
ANCESTOR=5,PARENT=3,OBJECT=2	memoryUsage	57
ANCESTOR=5,PARENT=3,OBJECT=2	hddSpaceFree	456764

Figure 3. Example measurement, made 12:45PM

Based on these measurements, historical data of some PI values can be formed. It is shown that during this period of time, some PI values have increased, some have decreased and one has stayed the same.

For visualization purposes this information is important to understand, because this work has to tackle problems presented in the next section.

2.4. Research questions

The purpose of this work is to firstly examine how proper visualization could make a user's work easier, or whether actually it is of any use to visualize things better. Secondly, this work aims to present a new, maybe different kind of perspective to FM and PM data.

In the winter and spring of 2009, I had a discussion with Nokia Siemens Networks system architects [NSN, 2009] and based on those discussions the following research questions were formed for this thesis.

Fault management:

1. How to capture the number of alarms over time?
2. Are there any patterns for specific alarms?
3. Are some alarms happening more frequently than others?
4. Can visualization find explanations for some alarms?

Performance management:

1. How to see the change of PI values over time?
2. Are there some trends in PI values?
3. Are there anomalies in PI values?
4. How to compare PI values in a meaningful way?

The purpose for this thesis is to find some visualization techniques that would cover all of these problems and help the network management O&M responsible person to pinpoint problems and bottlenecks in the networks she is responsible for.

One problem is the huge amount of O&M data in network management systems. One additional question which this work also needs to answer is that is it worthwhile or even possible to try to visualize everything using the data sets available.

3. Information visualization

3.1. Introduction to information visualization

Information visualization is a method which utilizes some available information set, collects some information from that data set and represents that information in such a form that humans can interpret it quickly and meaningfully. Representation can be a poster, a picture, a 3-D model or an interactional software user interface.

Following quote from Herbert A. Simon [Simon, 1996] describes the purpose of information visualization pretty well:

> "Solving a problem simply means representing it so as to make the solution transparent."

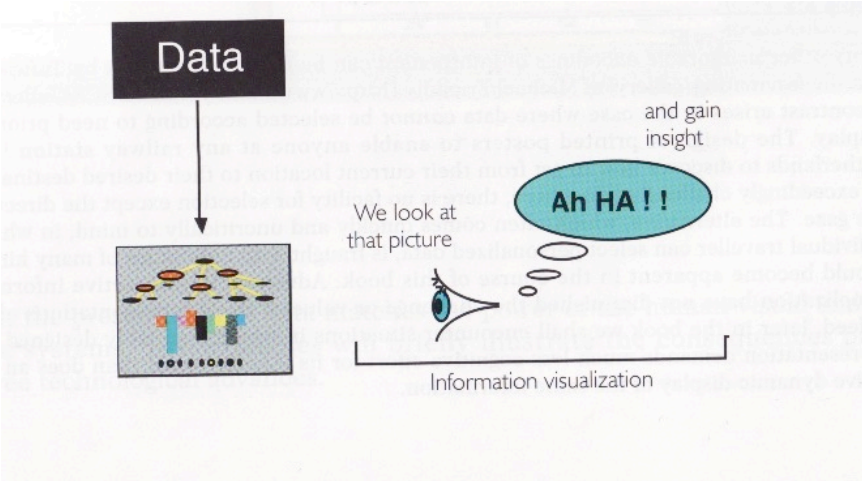

Figure 4. Visualization in the work [Spence, 2007, p. 5]

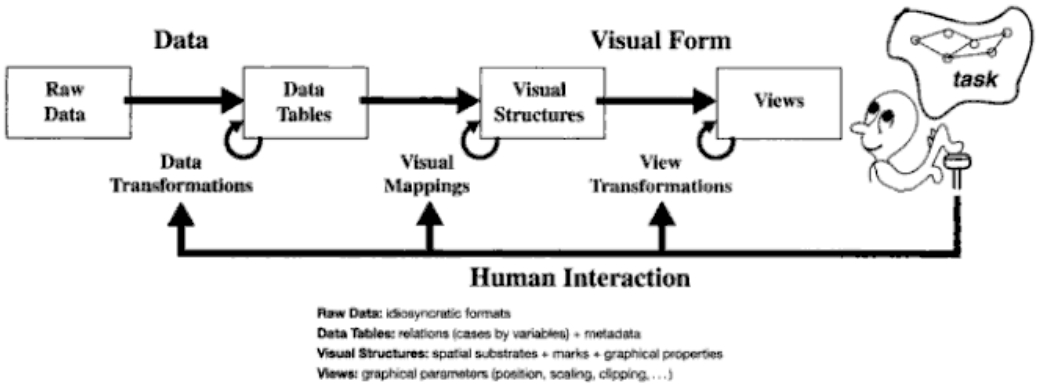

Figure 5. Information visualization reference model [Card *et al*, 1999, p. 17-33]

Different interpretations of the information visualization process can be seen in figures 4 and 5. Robert Spence has a simple yet very effective view on the subject in his book Information Visualization: Design for Interaction [Spence, 2007, p. 5]; In his view, data is transformed into pictures and a human being gains knowledge or makes a useful discovery by watching the pictures. In Spence's view, data and the visualization are separated, the task for information visualization is to allow information to be derived from data.

Card, Mackinlay and Shneiderman [Card *et al*, 1999] have constructed a reference model about information visualization, shown in figure 5. This reference model describes more precisely what actual phases the creation process of information visualization consists of. The model contains 7 consecutive parts.

Firstly, there is the *raw data*, which means data in its purest and most native form. In the second phase there is *data transformation* from raw data into a *data table*. A data table contains data and its metadata. Metadata means descriptions about data and the relationships of the data entities. The next phase is *visual mapping* which is a transformation process to transform contents of a data table into *visual structures*. Visual structures combine spatial substrates, marks and graphical properties. They also have to be perceived well by the human user and they have to preserve the data. To generate a human-viewable entity of *view* from visual structure, there is the *view transformation* process. View transformation can be an interactive modification and augmentation. It can be divided into three categories: Location probes, viewpoint controls and distortions:

1. Location probes: Location is used to form a specific view of the structure
2. Viewpoint controls: Zooming, panning or clipping
3. Distortions: Hiding or displaying certain things

During the phases of data transformation, visual mapping and view transformation there is human interaction present to transform the earlier data structure into the next one. Human interaction does not necessarily mean interaction per se, it can also mean programming the needed program code which makes the transformation.

A reference model is a useful tool for describing the information visualization process and for explaining some sub-processes. It also allows, according to Card, Mackinlay and Shneiderman, [Card *et al*, 1999] to simplify the discussion

about information visualization systems and to compare and contrast them. This kind of process explanation is also handy for software industrial purposes, where one has to measure, quantify and estimate work.

In my opinion, identifying the purpose and the need for visualizations is essential. One has to understand what is the actual use case of the visualization that is created. If the use case is not clear, the designer cannot design a suitable visualization. Understanding the working environment is important and therefore the designer needs to make studies related to the environment. Basically the same principles apply which apply to the user centered design practices; the designer needs to know the user's needs for whom she is designing for. One cannot fill the motorway traffic signs with 100 lines of text or display subway maps with station names alphabetically ordered with no linkage to the actual geographical locations. The designer needs to understand the use case.

Visualization usually involves showing several variables of the data set and the relationships of these variables. I see that information visualization sometimes needs thorough data mining practices and careful analyzing of data. Those are essential steps in making a successful visualization experience. The science of statistical analyzation, knowledge of human behaviour and the art of visual design walk hand in hand.

Although information visualization can be considered more an art than a science, one has to utilize the same principles as in user centered design; the user has to be involved in the making. This means not only the requirements gathering and environmental understanding, but also thorough testing with the actual user, good communication and iterative development of the visualization.

3.2. Evaluating visualizations

What makes information visualization more a science is the fact that it can be evaluated using some principles. The most important evaluation principle being the user testing and user feedback in the multiple phases of the information visualization design. Although the designer can have artistic and really subjective views on the design, in the end the final evaluation is in the hands of the actual user of the visualization to decide whether it is useful or not.

Jacques Bertin defines a measurable criterion for evaluating graphic designs in his work Semiology of Graphics [Bertin, 1967]. He calls the criterion as *efficiency*. [Bertin, 1967, p. 139] Efficiency in his work means basically that the shorter time it takes for the viewer to observe a construct, the more efficient it is. This sounds a pretty simple approach, but in the end that is what visualization is all about; to find a solution to the problem as fast as possible.

The most typical testing activity of user interface designs is that there are multiple testing tasks that the subject has to perform, the time for performing the task is measured and the task is either marked as a success or a failure. Tests might be piloted earlier with just 1-2 test users so that the most critical errors in the tests would be corrected for the actual user test [Nielsen, 1993, p. 174].

Catherine Plaisant has studied the information visualization evaluation field [Plaisant, 2004] and she has identified three evaluation challenges.

Firstly, the tools have to be matched to the users, tasks and real problems. This basically means that the designed visualization should be suitable for the purpose it is made for.

Secondly, user testing has to be improved. It is the backbone of the evaluation, even though test tasks are often very easy, because of the usual time limitations. According to Plaisant, there should be tests done along a longer time period for the same dataset. Also, test users should be able to perform visualization evaluation on their own, outside of the typical user testing tasks, even if this requires them to be highly motivated to explore the visualization at hand. Subjective satisfaction and learnability aspects should also be included in the user testing. Evaluation of user testing should reach beyond just checking error rates or the timing of tasks, as a user might find some totally new phenomena or behaviour by looking at the visualization and this improvement is hard to quantify.

Thirdly, universal usability needs to be addressed. Visualization has to be accessible and usable by a very broad audience, and a designer has to take different kinds of disabilities and special user groups into account.

The point that Plaisant made about the visualization answering the real problems and the long-term evaluation has been also noted by others, Elaine R.A. Valiati et al. made also the following comment [Valiati, 2008]:

> "In the information visualization literature, most of published works related to user involvement focused on controlled experiments in-laboratory. However, sometimes this experimental procedure may be inadequate, mainly during a research laboratory stage when goals and tasks - among other variables - may be not yet defined. Thus, for evaluating visualizations, longitudinal studies involving actual users participation are strongly recommended."

Unfortunately within the scope of this work there is no possibility to perform any suggested longitudinal evaluations of the designs but a moderated user test has been arranged and testers are more of skilled in the network management field.

3.3. Visualizing O&M data

There are several classes of data according to Spence [Spence, 2007, p. 30-70]. Data can be classified based on a visualizable number of attributes:

1. Univariate, has one attribute
2. Bivariate, has two attributes
3. Trivariate, has three attributes
4. Hypervariate, has more than three attributes

When thinking about the best ways to visualize different fragments of O&M data it is good to evaluate with the principles given by Spence [2007] into which data category those fragments would fall when regarding the number of attributes.

Network object topology is univariate data as such, managed objects only have one attribute: a distinguished name. On the other hand, a managed object also has information of its parents, and therefore could be seen as bivariate data. Performance management data can be seen as trivariate; a single performance indicator has a reference to its value, its managed object and also the time when it has been measured. Fault management data is hypervariate. A specific problem and distinguished name make an alarm unique. For the user, on the other hand, the alarm's severity (because that can tell how fast the alarm situation should be fixed) and alarm's event time (because of the same issue, time is important) are also relevant, so we get a total number of 4 attributes.

In addition to visualization attributes, one has to realize the amount of data in a typical network management system. Some examples of data amounts were provided by Nokia Siemens Networks architects [NSN, 2009] and the results were that visualization should handle hundreds or even thousands of alarm events per second at the highest level of the network management system. On the performance management side, a raw example estimate would be 250,000 performance indicators polled every 15 minutes from a single specific network element. This is only one example, the variance between different network elements is huge and this example would represent something from the larger end of the data set.

It is needless to say that these figures pose a big challenge for visualizations if one wants to visualize all of the data.

3.4. Existing O&M visualizations

3.4.1. Monitoring versus reporting-like behaviour

Existing visualizations presented in this section are based on the assumption that fault management traditionally is more a real-time monitoring activity and performance management is more a reporting activity which happens at specific intervals, although it could also be modelled as real-time behaviour in some cases. This means that alarms are monitored constantly because operators need to react to some of them almost instantly. PIs are gathered at steady intervals which can be very long, even days, and therefore they are used in reports which can be generated at any time from the system.

In the following section, some existing O&M data visualizations are examined to get a base from where new ones can be built. In this work, the boundaries between real-time and reporting systems are not so clear, and visualizations are made to work for both possibilities. So even FM can be built to be reporting-like and PM can be updated real-time.

3.4.2. Fault management

First let's look at fault management visualizations in an existing network management system.

In figure 6 there is an alarm event table. It is an alarm table implementation with all the alarm events as rows and alarms attributes in columns. One could argue that is this a visualization at all, nevertheless, it shows the information a

user wants to see. Alarm attributes are pretty much similar to those described in section 2.3.2. At the top of the screen there is a summary of the alarm amounts according to different severities. Additionally, the alarm severity is colour-coded for visualization purposes in the first column of the table; red depicting a critical alarm, orange a major alarm, yellow a minor and blue one a warning.

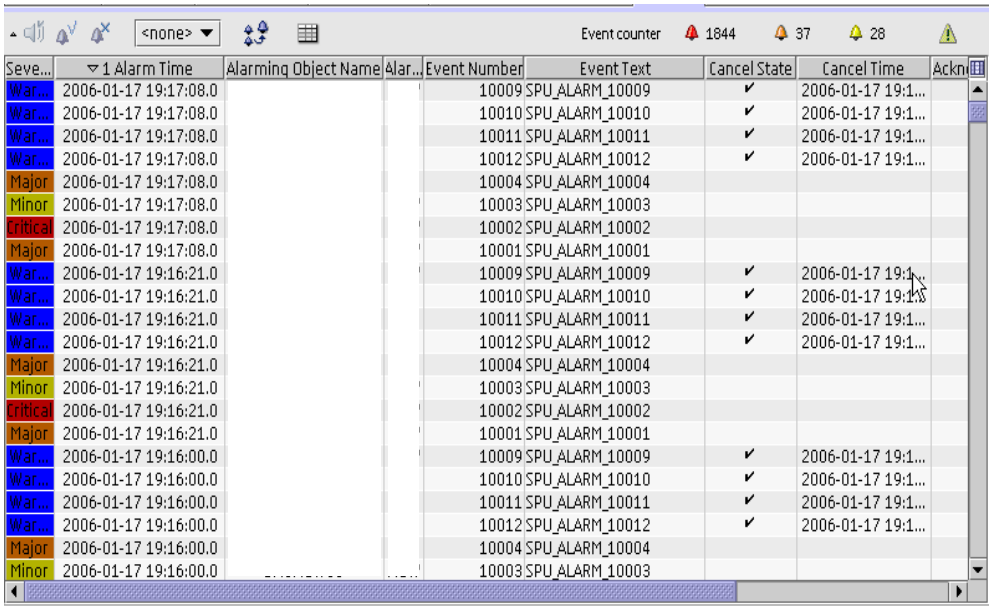

Figure 6. Alarm event table

There is also the possibility to filter the alarms in this table based on some criteria and to sort them by clicking the headers of the columns.

Based on this visualization, it can be seen that severity of the alarm is a very important data in the alarm event table. Emphasizing the most severe alarms is something that visualization needs to answer, which is also one of the visualization issues already presented in our problem domain section 2.4.

In figure 7, there is a details view which opens when an alarm has been selected from the event table. It shows more specific information about the situation and can be used to show data which would not otherwise fit in the alarm event table.

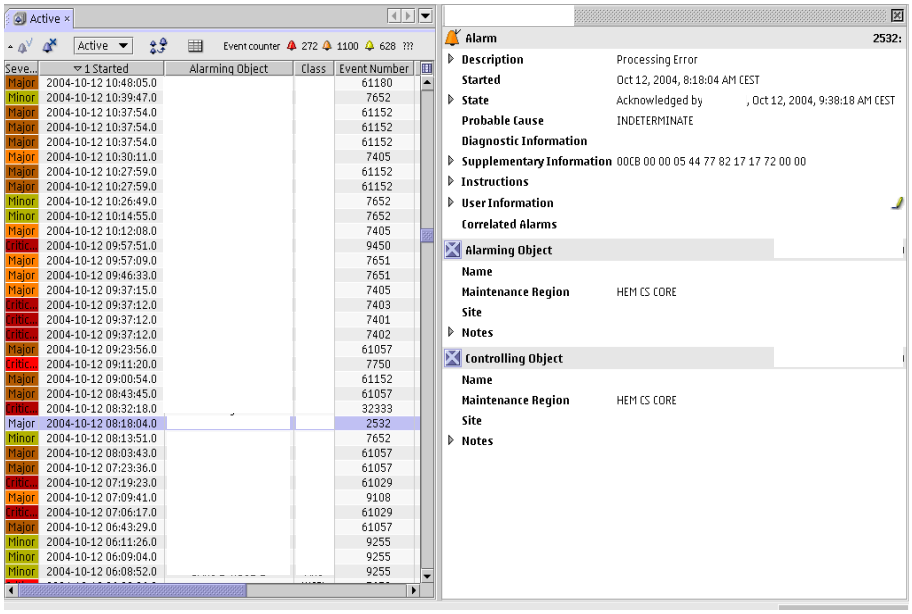

Figure 7. Alarm details view

Finally, in figure 8, a part of the network topology view is presented.

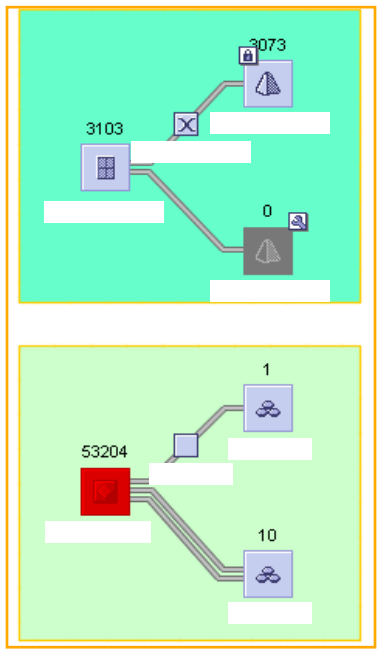

Figure 8. Two examples of the network topology view

It is a very top-level view where a user can see the managed objects and their connections as a graph. The state of the managed object is colour-coded. It could be used for instance so that critical alarms turn the managed object red. There could also be some other symbols with some other meanings next to the managed objects describing the amount of alarms or various other states of the managed object.

3.4.3. Performance management

On the performance management side, there are two examples of displaying the data. The first one, in figure 9, is probably the most simple way of displaying it; displaying measured performance indicators in a table.

No.	Period start time	name	name				
1	01.17.2007	BRBOBC0	LA745B	338.75	56.25		0.00
2	01.17.2007	BRBOBC0	00791A	1,951.50	727.88		0.00
3	01.17.2007	BRBOBC0	00791B	11,726.88	2,279.13		0.00
4	01.17.2007	BRBOBC0	00794A	5,651.00	3,414.50		0.01
5	01.17.2007	BRBOBC0	00794B	720.25	1,027.13		0.00
6	01.17.2007	BRBOBC0	00794C	709.00	322.13		0.00
7	01.17.2007	BRBOBC0	01673A	343.50	104.75		0.00
8	01.17.2007	BRBOBC0	01673B	284.25	93.38		0.00

Figure 9. Measurement result table

This table consists of measurements made for performance indicators on specific objects. The second column from the left tells the time when a measurement has been done, the third column displays the parent managed object's name and the fourth column shows the name of that parent's child object. Following those are the actual performance indicator values.

Like in the alarm event table, here also rows can be sorted by clicking a single column. Rows are then ordered according to the values in that column.

The second and third examples of displaying the performance data are presented in figures 10 and 11. There are two types of graphical presentations of PI data. Both figures are presenting one managed object's PI values changing over time. In the first one, a simple line chart is used for a single performance indicator and in the second one a combination of line chart and a histogram is used for three PIs.

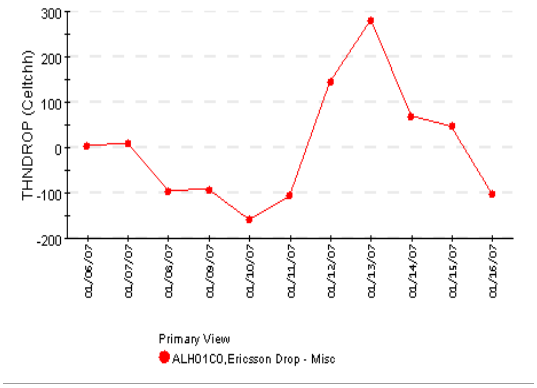

Figure 10. Graphical view of one performance indicator value over time

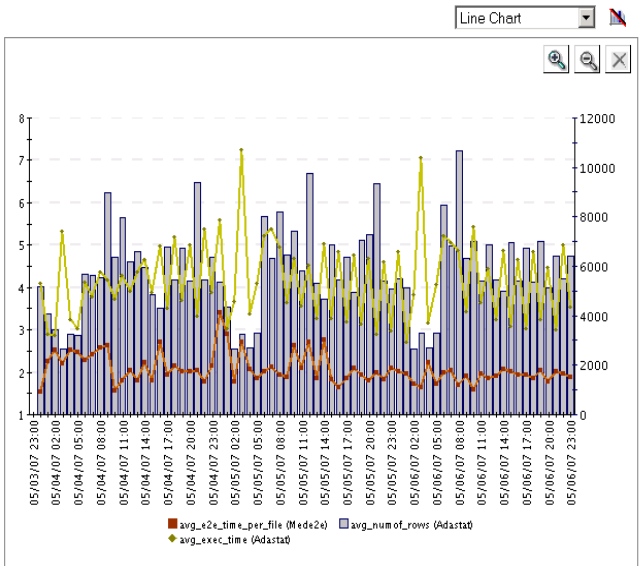

Figure 11. Graphical view of three performance indicator values over time

Figure 11 shows the problem of displaying multiple types of PIs in the same chart. Different types mean different numerical spaces. There is one PI type whose value varies in between 1 and 8. Another PI type's value ranges from 0 to 12,000. Those values have to be presented on separate axes. In a two-dimensional chart there can only be two types of values presented. If there would be a third PI type, a separate chart would be needed for that.

3.4.4. Problems with the existing visualizations

When analyzing the visualizations just presented based on the problem domain that was created in section 2.4, the following conclusions can be made.

On the fault management side it can be safely said that these visualizations do not solve any of the presented problems. The visualizations do not show alarm

amounts over time, although a user can sort the alarm tables based on the time of the event. Still, the amount of alarms over time can not be seen, at least not very clearly. Same goes for the pattern; a user can sort the table according to time, but it is not really clear which alarms are triggered more often than others.

The performance management problems are answered fairly well by the example visualizations. A simple linechart presentation seems to answer almost every problem except the last one; comparing performance indicator values to each other in a meaningful way. Like stated in the previous section, it is easy to compare performance indicator values in a chart if they share the same numeric space, but if the performance indicators that are to be compared have a different numerical space, their comparison is hard to present beyond two types of numeric values.

3.5. Related work in information visualization

The reorderable matrix concept developed by Jacques Bertin [Bertin, 1967, p. 256-257] is interesting and useful for displaying information in a really usable manner. The data in a matrix is organized by reordering columns and/or rows in such manner that cells with the most similar values are located closest to each other.

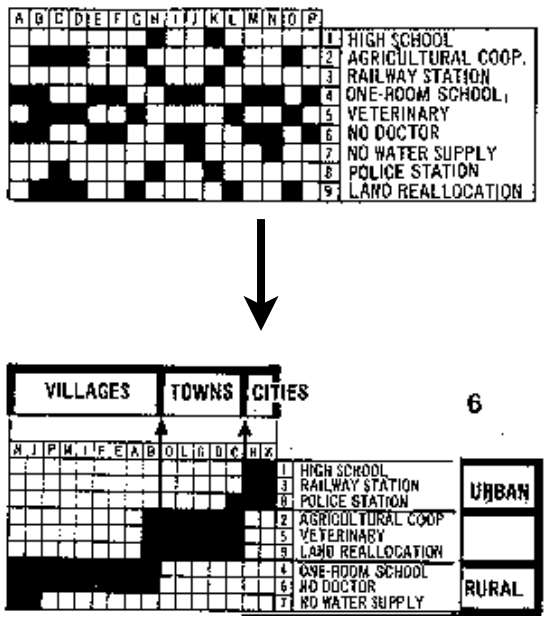

Figure 12. The reorderable matrix. The upper picture presents the raw data matrix before reordering and the lower picture the matrix after reordering the rows and columns. [Bertin, 1967]

The idea is to form a new kind of understanding of the data with the reordering. The value is visualized by changing the size or colour of the cell according to the value. This way the information is more visible and understandable for the viewer.

For PM purposes, it needs to be studied how the reorderable matrix approach would work. The trends or anomalies in data might be more spottable with this approach.

Harri Siirtola et al. have studied the reorderable matrices and parallel coordinates [Siirtola, 1999, 2003; Qeli, 2004]. Reorderable matrices are described in that work as varying-sized blobs inside the table cells. Bigger blobs represent data which has a bigger value compared to others in the matrix.

Parallel coordinates [Inselberg and Dimsdale, 1990] is a technique suitable for representing data which has many attributes. The idea is to display multiple attributes with as many parallel axes and the data values are simply lines connecting those axes. So n-dimensional data can be displayed with just two dimensions. The positions of the lines are representing the relationships of values with each other. One can display as many dimensions as there is room in the display. Based on Siirtola, parallel coordinates have been proven to work even with over 100 variables [Siirtola, 2003]. So, parallel coordinates is a technique which is quite suitable for presenting data with a large number of variables, but in this work the problem is maybe not so related to the amount of attributes but more to the size of the data.

Siirtola uses the matrix to represent parallel coordinates in such manner that columns and rows of the table can be reordered based on the values [Siirtola, 2003]. He uses it together with the traditional parallel coordinates view - the parallel axes view - to form a combined view where one can select data from the matrix and display it in the parallel axes view. In the evaluation of the reorderable matrix combined with the parallel coordinates, it can be seen that there are benefits from combining the different views. This is a finding that is examined also in this work so that different, even conceptually different, views are utilized in visualization for users to form a better understanding of the data.

Reorderable matrices are also used by Ermir Qeli et al. [Qeli, 2004] to form a matrix of coloured data for representing value changes over time. This is also relevant research for this work where the displaying of value changes over time is an issue. The procedure for making the visualization is such that first the data

matrix cells are coloured based on the value change over time, then the matrix is reordered based on those colours. Then it is easier for a user to see where the actual changes happened.

Daniel A. Keim has made some notable studies and work in the field of pixel-based visualization [Keim *et al*, 2002, 2006; Ziegler *et al*]. The idea is that one visualized object can be as small as a single pixel on the screen. To visualize multiple attributes in one pixel, the methods to use are the location and colour of the pixel. This way, though, the amount of attributes is limited to three (X-axis, Y-axis and colour) and in case there are more attributes present, one has to decide what attributes to include in the visualization. By using the relative location data and by generating pixel bar charts presented in [Keim *et al*, 2002] it is possible to present even more attributes.

Because of the huge load of information that exists in the O&M systems, pixel-based information could be useful for also this work's visualization designs, as screens should be as full of data as possible.

4. New O&M data visualization designs

4.1. Common design practices

4.1.1. Rationale for chosen designs

As it was already stated in section 2.4, the main problem for O&M data visualization is the huge amount of data that needs to be displayed. For large data set cases, data has to be packed somehow to fit in the available display. According to Becker *et al*. [1995], the three methods usually used for reducing the network data size are:

1. Aggregation, for large number of links or nodes
2. Averaging, for large numbers of time periods
3. Thresholds and exception reporting, for detecting changes

Aggregation and averaging are useful tools for data packing in this work. Thresholds and exception handling are also useful visualization tools and thresholds are somewhat used for providing filtering capabilities for the visualization. Exceptions are not considered in the designs, but they have to be considered in future development based on this work.

As it was stated in section 3.3, the FM data is hypervariate and PM is trivariate. It was also noted that PM should be able to handle in some cases roughly 250,000 performance indicators on a single point in time. If those PIs would be presented as single pixels without *data packing*, those could be fitted roughly on a display with the resolution of 500x500 pixels. In the case of PM, trivariate data for 250,000 PIs could be displayed with pixel-based visualization but the visualization would be limited to one time-instance. If a display with a resolution of 800x600 pixels would be used, larger amounts would be difficult to show, and if the time-scale could not be shown, the problem 1 for PM, presented in section 2.4, "showing performance indicator values over time", could not be solved. If, on the other hand, time would be put as one variable the amount of PIs would be seriously limited.

The technique presented by Keim et al. in Pixel bar chart [2002] shows a good example of a bar chart which has pixel-information in the bars. The same kind of behaviour could be utilized, generating bars of PIs in our visualization of PM. Each bar would represent a single measurement time slot, a single point in time, when those performance indicators were measured. Bars could be 500 pixels high and 10 pixels wide per time slot, so that way there could be 50 time slots with 5000 performance indicators each in the same resolution of 500x500

pixels. One alternative would also be 10 time slots with 25,000 performance indicators each. It is evident that some kind of compromise needs to be reached; all 250,000 performance indicator instances just can not be shown as single instances if they need to be compared on a common time scale.

On the other hand, with modern display technologies and chips, even a regular consumer can afford a WQXGA display of 2560x1600 pixels. This resolution means that roughly 4 million pixels can be displayed. If the entire screen would be filled with PI data, 16 time instances of those 250,000 PIs could be presented. It is needless to say that in this project, the whole used display resolution has to be used as efficiently as possible for the performance management visualization. Keeping in mind that the customers and end users of these visualizations are corporations; operators and their personnel, I think that the displays are much better than those presented here. Operators will make the investments if their income depends on it.

On the fault management-side, there is also a huge load of data to deal with. According to Nokia Siemens Networks system architects, [NSN, 2009] a system can receive hundreds of alarms per second from the network. Similar pixel bar charts could be used in that case also. If a similar resolution would be available, and a maximum of 1,000 alarms were put per time slot, they could be displayed in 2x500 bars and in 250 time slots.

Cases presented above are optimum cases when one wants to present one performance indicator or alarm as one pixel. If more information needs to be presented, the size of the pixel has to be extended to a square with some information, e.g. text, inside it. If the sizes of our squares are extended, the amount of data that can be presented is greatly reduced. One possible solution to this dilemma is the distortion+focus technique, also known as the “fisheye view”, first described in the Bifocal display system [Spence & Apperley, 1982]. With this technique, the specific part of information is highlighted and its properties are extended to be more detailed while still maintaining the other surrounding information in the view. In these visualization cases, the latest information could be extended by default, but the visualization could be interactive, so a user might highlight and extend whatever part she wants to highlight.

The fisheye view would be the solution for some hundreds of alarms. But when you are talking about thousands of alarms in the same view, you would need to have just simple pixel-based visualization with all the alarms shown just by one

pixel. If requirement is to address the problem described in section 2.4 “Visualizing amount of alarms over time” and all the alarms over time need to be seen, even one pixel is not small enough for a single alarm instance.

Because of these different scales, all the problems can not be solved really easily with just one visualization view and so different views for different scales of data amounts are needed. Michelle Q. Wang Baldonado et al. have researched the multiple view usage in the field of information visualization [Wang Baldonado *et al*, 2000] and they have gathered four rules regarding the use of multiple views.

The rules are *diversity*, *complementarity*, *decomposition* and *parsimony*. The rule of diversity is telling the designer that multiple views should be used when there are many attributes, models, abstraction levels or genres present. The rule of complementarity means that different views reveal correlations between data structures. The rule of decomposition means that multiple views are used when data needs to be decomposed into more manageable chunks. Finally, the rule of parsimony tells that one should use multiple views as little as possible.[Wang Baldonado *et al,* 2000]

If these rules are followed, it is easy to divide the FM visualization into two views:

1. Large scale FM view
   - All alarms in one single view
2. Alarm event view
   - Alarm instances in more detail

These FM visualizations are utilizing the same concepts in the design, as can be seen later in the design presentations, so they might be fused together somehow with some clever linking principles. The designs of these views are presented in more detail in section 4.2.

Two different views are presented for PM also, although they are not so similar but there can be different kind of approaches to the same data:

1. PM view A: Performance indicator matrix
2. PM view B: Reordered performance indicator matrix

To be able to filter the data for the views, there is a need to design an interactable user interface for selecting the filtering criteria. The same type of

interface could be used for both FM and PM because filtering information is pretty similar.

The filtering criteria is selected based on what both FM and PM fragments have in common. That is the distinguished names of objects and event times. That criteria needs to be displayed in viewable, interactable and usable manner.

4.1.2. Design proposal for common view and filtering UI

First, the overall layout of the user interface for this prototype is presented. Figure 13 contains a layout of the UI design.

Figure 13. User interface layout of the concept design

The layout of the design is simple and traditional, it contains filtering criteria, the visualization and the details view. In their book About face 3, Cooper et al. [2007] define interaction design patterns. Based on those patterns, this design can be categorized as a member of structural interaction design [Cooper *et al*, 2007, p. 158-159]. The structural pattern simply means that the layout of the software design is such that specific parts of the layout have specific structural meanings. This design is optimal for full-screen purposes, the manipulation of view objects and the displaying of the detailed attributes of those objects. The pattern permits all this to be done on a single screen without the need for extra windows or screens. Specific parts of the layout are not overlapping each other and no dialogs or menus are displayed.

Some description of the components on screen:

1. Filtering criteria. At the top there are the managed objects in a tree-like hierarchical view. Whenever a user clicks any managed object, the visualization view changes so that only alarms which belong to that managed object are displayed. If a user selects a top-level node on the tree, alarms or performance indicators of that managed object and all managed objects under it are displayed. On the bottom there is the date-time filtering criteria, starting time and ending time. If a user selects a starting time, only alarms or performance indicators visible after that time are displayed and if a user selects the ending time, only alarms or performance indicators visible before that time are displayed.
2. Visualization view. This is the visualization display. A user can manipulate the view with a pointing device. There are buttons at the top of visualization view for changing between multiple views. There would be an option to make the visualization displayable using the full screen resolution, so that the whole screen size could be utilized for maximum performance.
3. Details view. This area can display some more specific information on any visualization object which a user selects with a pointing device.
4. Legend. A user can select the colours she wants for the visualization by clicking the appropriate types. This is a useful feature, for the colour-blinded people at least.

4.1.3. Interaction design

The human-computer interaction in general and the interaction in this work follows also Donald Norman's action cycle model [Norman, 1988, p. 45-53] presented in figure 14. In short, this means that to achieve a goal, a "change in the world" must be done. This means that user needs to execute some action which affects the software so that user feels that his will is done.

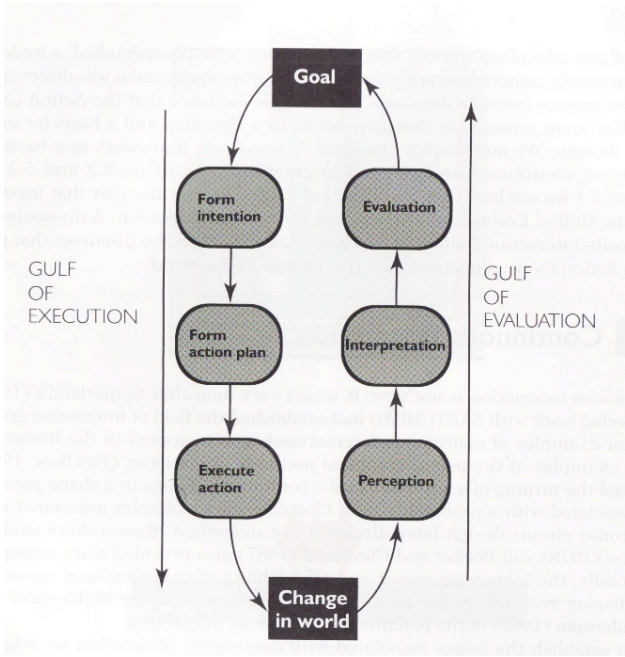

Figure 14. Norman's action cycle [Norman, 1988]

This kind of action mapping is particularly true in the case of information visualization. What visualization design can do, is to reduce the cognitive steps in the gulf of evaluation and in the gulf of execution. Visualization can be designed so that whatever the user does to the visualization, i.e. filters it or clicks viewable entities, he gets a response which he can see, understand and evaluate quickly. In the gulf of execution stream, the intention and action plan forming are also cognitive efforts and the visualization can help by providing the user familiar, recognizable and intuitive controls that he can associate with the actions he wants to achieve. [Norman, 1988, p. 45-53]

I have designed the visualizations so that a user has to use a pointing device to interact with the visualization. A user can move around the visualization, interact with the fisheye views either by hovering the cursor over the visualization or by clicking the pointing device on selected parts or entities. Hovering or clicking can be used to select the data entity in the view. Pointing device can be a mouse, but we do not exclude the use of touch screens with which the pointing device could be a stylus pen or a finger.

Overall interaction of the visualization prototypes is intended to work so that whatever the user selects from the filters part, described in 4.1.2, the visualization should respond instantly. All filtering criteria are clickable, and when user clicks or selects some criteria or changes the existing criteria, the visualization is instantly reconstructed correspondingly. Entities which have

been selected out from the view are deleted and new entities which have been selected in are created. To enhance the gulf of evaluation, this should happen as fast as possible. This requires speed and efficiency from the algorithms that make the queries and filtering on the actual database, but the responsiveness speed is crucial for the user experience and for the behaviour of the visualization.

In FM visualization proposal, a user sees every received alarm in the visualization, but there is also a filter available where a user can select the monitored managed objects and time-frame in which she wants to monitor the alarms. This way, she can also monitor historical data from any given time.

In the PM visualization proposals, the user would need to have more filters to select data from. There would be the managed objects and time filter like in FM case, but also the selection of actual monitored performance indicators could be displayed. PIs could have metadata of types, so that a user could select what type of performance indicators she would want in the visualization. Due to the nature of the PI types, there could be percentage numbers, gauges and incremental PIs. One would have to decide what to include in the visualization so that the visualization would be useful for the given purpose.

4.1.4. Prototype implementation

The actual implementation of these designs is a Flash application developed with Adobe Flex Builder 3 [Adobe Flex Builder]. The main benefits of this technology are platform-independency and ease of user interface development. The latter benefit was the main reason for selecting this technology for this work.

The programming language used in the work is Actionscript 3.0. Actionscript is a standards-based, object-oriented language [Kazoun & Lott, 2008]. It's syntax resembles very much Java's language syntax. Because Actionscript is an object-oriented language, it can be viewed as a collection of APIs generally in the form of classes.

Because of the popularity of the Flex development tools, there is a huge number of ready-made components for the Flex applications from the web. One particular such component was used in this work: Ely Greenfield's fisheye view component [Greenfield, 2009]. With that component the implementation of the

fisheye view for FM visualizations in this work was much easier and less time-consuming than implementing those from scratch.

This implementation is designed to work in a web browser with Adobe Flash Player 9 [Adobe Flash Player] or later installed. The Flex development tool could be used also to create the application as a desktop application with Adobe Air [Adobe AIR] technology, but a browser-based implementation was selected because of its better platform-independency and easier deployment.

In the scope of this work, there was not sufficient time nor resources to make every design presented here into a prototype application. One design from FM and one from PM were selected as the most promising designs. Those prototypes were user tested and the test results are presented in section 4.4. The selected designs were considered as the most influential and therefore the most important designs in this work.

4.2. Design proposals for FM

4.2.1. Large scale FM view

A typical fault management visualization, as presented in section 2.4.2, is a table of alarms. In this work, the fault management visualization is presented as an interactive visualization of stacks of alarm boxes on a time axis, representing the situation in the alarm table across time.

In the "Large scale FM view" the goal is to see every alarm ever sent. This is possible by packing data into such form that it can be displayed using any resolution. The view is two-dimensional; the entity amount is shown on the Y-axis, and time is shown on the X-axis. A simple solution to packing data is to divide the given data into displayable proportions.

If the data is packed vertically, it means that separate entities have to be combined into one. In the FM visualization, entities are alarms. In the PM visualization they are performance indicators. How this can be achieved is presented in the specific FM and PM visualization proposals.

If the data is packed horizontally it means that time-wise, a timeframe is taken at intervals, e.g. every 10 timeframes and the timeframes are put into one, again displaying the proportions during that timeframe. This means gathering all the alarms with the same severity into one entity.

Figure 15 shows the concept design. Vertically, there are 4 possible alarm severities used. The proportions of those severities are visualized with size, so that the user can understand which severity group has the most alarms. In the visualization, a bar is created which is as tall as the total amount of alarms. Next, this bar is divided into 4 sections, which represent the alarm severities. Finally, the portions will be set in such manner that their relationships are visible to the viewer. The number of alarms during a certain time period can also be seen at a glance in the visualization.

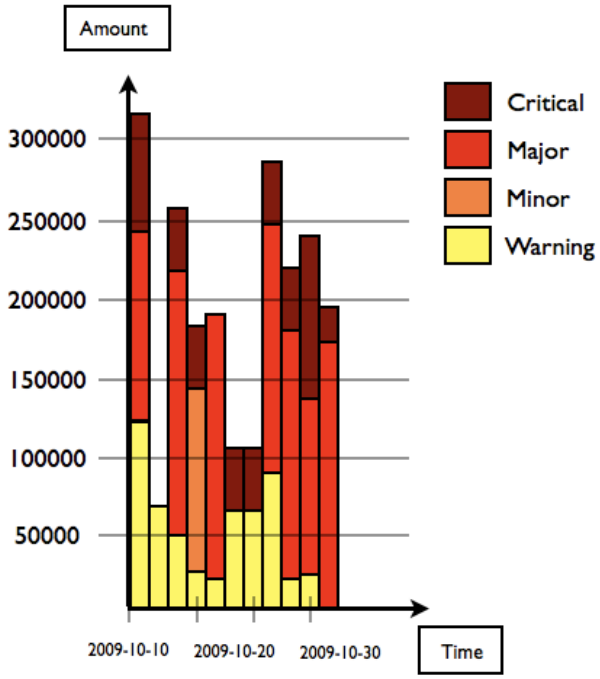

Figure 15. Large scale FM view concept design

This view reflects the situation in the alarm table. This means that the alarm boxes do not reflect the actual alarm events; they describe the alarm table state at any given time. Alarms can appear and can be cleared during the times, and the bars show which alarms have been present in the alarm table during that specific time period.

There could be some kind of linkage implemented between this view and the following ”Alarm event view”. Clicking some parts of the graph might select some alarms for the ”Alarm event view” but in this work that functionality is not covered.

4.2.2. Alarm event view

This visualization follows the same concepts as the ”Large scale FM view” in 4.2.1. The alarm entities are more visible and more distinguishable in this visualization. Also more of their details can be seen. Like the earlier visualization, this visualization also reflects the situation of the alarm table across time.

In figure 16, the main elements of the visualization can be seen. All four attributes described earlier in the FM presentation can be displayed; alarm's severity, alarm's triggering time, managed object's distinguished name and alarm's specific problem-number.

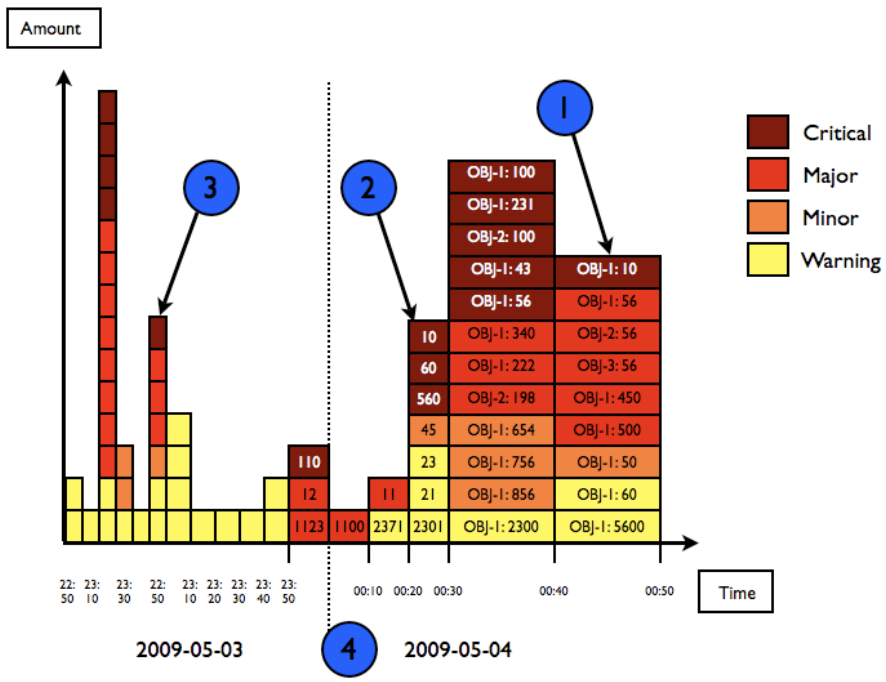

Figure 16: Alarm event view concept design

The fisheye concept presented earlier is utilized in this visualization. When the pointing device is moving in the visualization, it highlights and expands the parts over which it travels. Parts which are further away from the pointing device contract.

Description of specific elements in figure 16:

1. The latest alarms are on the right side of the chart and highlighted by default. Highlighting means that the alarm boxes are slightly bigger

than the rest, and they have the managed object's name and specific problem-information inside them. Colour depicts the severity of the alarm. Alarms are sorted vertically so that the most severe alarms are on top, and the less severe at the bottom of the stack.

2. Alarms next to the highlighted-ones are in smaller boxes. Boxes do not contain distinguished name-information, but they have specific problem-information.
3. Alarms further away from the highlighted-ones are even smaller, and they do not have other attributes than colour. These alarm boxes could be made as small as one pixel-size in width.
4. Time-axis shows the dates and times and in case there is a date change, it is displayed with a vertical line. Alternatively, vertical lines could also be used to display hour change.

4.2.3. Alarm details view

Alarms have valuable information inside, so the visualization needs to display that also. By clicking a certain alarm box with a pointing device, the alarm details view appears beneath or beside the visualization as shown in figure 17.

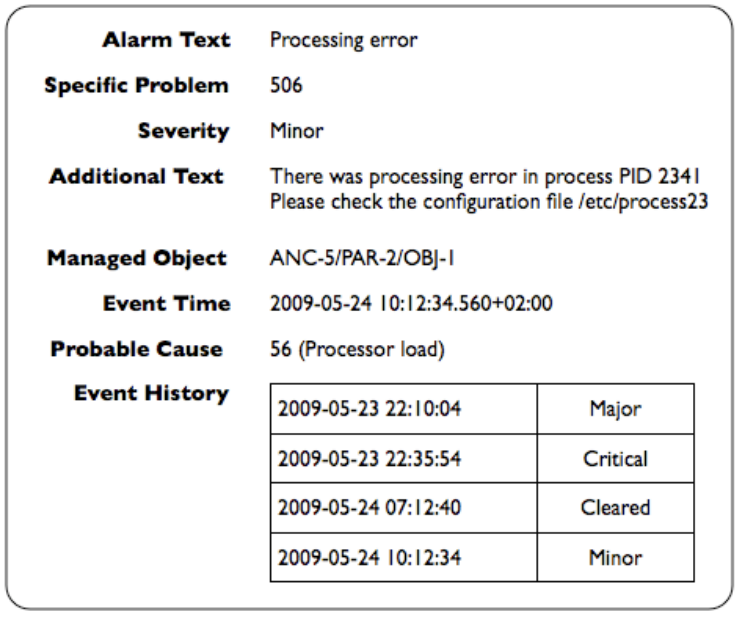

Figure 17. Alarm details view

The alarm details view presents more specific data about an alarm that would not fit the visualization any other way. This data might still be important to the

user to see so that she can handle the alarm, and notify other personnel to fix the problem. This is just an example what the view could hold, and actually some additional fields besides these could be added. Some statistical data might be also important. Maybe some indicator telling for how long the alarm has been active without any change in its state.

4.2.4. Thoughts about FM concept designs

When the proposal designs are analyzed in the light of the problem domain, presented in section 2.4, it can be said that they definitely solve the problem of showing the amount of alarms over time. For the other given problems, they do not answer so well. For the problem of alarm patterns, a user can maybe get some kind of vague idea by viewing the chart and by viewing the height of the bars. For the causes of the alarms, it is difficult to say how well the visualization would answer. Probably not well.

4.3. Design proposals for PM

4.3.1. Colour-coding performance indicator instances

A typical performance management visualization is either a line chart or a bar chart. In this work, the performance management visualization is coloured boxes on a time axis. In this section, the basic concept of both PM visualizations - colour-coding of performance indicator instances - is presented.

To keep PI instances visible in these visualizations, performance indicator data is not packed. This means that there will be an upper limit for performance indicators, and also that the data set has to be filtered before the visualization can be presented.

The most crucial problem of the PM visualization was presented earlier; how would the visualization be able to compare PIs which would share a totally different numerical space. The solution is to use PI values that are colour-coded based on how they measure up to the overall scale, across time. This could be done by taking the median value of the given data set and showing the value's difference compared to the median as a specific colour. The median value could be selected because it is immune to extreme values, it's values are affected only by the values that are the centermost of the data set [Holopainen and Pulkkinen, 2006, p. 76-80].

How to measure the difference from the median is the next question. One solution would be to divide the data set in to quartiles, 4 segments, where each segment would represent 25% of the values of the data set. An even better way to divide the data might be to classify it by deciles; A decile is 10% of the data set values [Holopainen and Pulkkinen, 2006, p. 81] and when summing up 10 deciles, one gets a picture of the complete data set. Deciles could be shown with different colours for better understanding of the value's measure. Deciles would be better than quartiles because there would be more segments to show, and the value jumps between segments would not be so big. A simple description of deciles would be that 1. decile is the lowest 10% of the data set, the 2. decile is the next 10% of values of the data set and so on.

The procedure is demonstrated by using the example data presented in figure 18. In that data, there are two performance indicators measured every 15 minutes. PI "memoryUsage" is a percentage-value and the other PI, "hddSpaceFree" is a number representing kilobytes. There are 10 values of both PIs in the measurement history.

	memoryUsage	hddSpaceFree
10:15 AM	34	56078
10:30 AM	54	55090
10:45 AM	23	45076
11:00 AM	87	12980
11:15 AM	88	8785
11:30 AM	95	7600
11:45 AM	76	7650
12:00 PM	66	8754
12:15 PM	50	9780
12:30 PM	32	12348

Figure 18. Example measurement of two performance indicators

The construction of the visualization for these performance indicators begins by dividing the data set into deciles. Next, a scale for the performance indicator values for marking the specific values has to be created. If deciles are used, the scale has 10 steps, which each can have configurable colours. In this example, however, the values are like in figure 19.

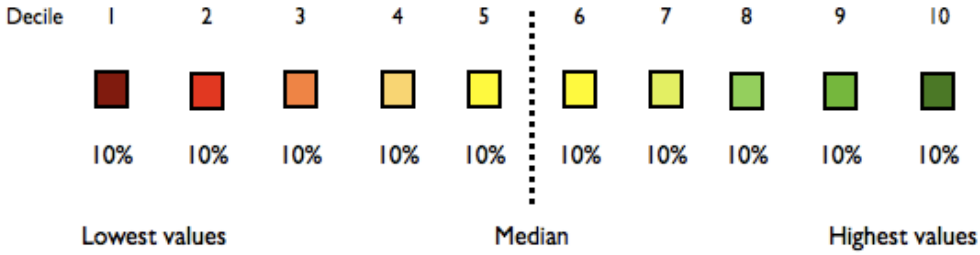

Figure 19. Example decile scale for PI value presentation, bigger values on the right, smaller values on the left

Scale is constructed so that the colour yellow is in the middle, describing the median value and all the values that are within 10% of it, either higher or lower. Higher values are in shades of green, and lower values are in shades of red. The next colour from yellow, either way, represents values that are 10% higher or lower of the median value and so on. Decile 1 contains the lowest values and decile 10 contains the highest values. The next step is to apply the scale to our performance indicator values. When this scale is used and performance indicator values are coloured, the result will be the table shown in figure 20. The decile is also calculated now and it is shown next to the performance indicator value.

	memoryUsage	hddSpaceFree
10:15 AM	34 (3)	56078 (10)
10:30 AM	54 (5)	55090 (9)
10:45 AM	23 (1)	45076 (8)
11:00 AM	87 (8)	12980 (7)
11:15 AM	88 (9)	8785 (4)
11:30 AM	95 (10)	7600 (1)
11:45 AM	76 (7)	7650 (2)
12:00 PM	66 (6)	8754 (3)
12:15 PM	50 (4)	9780 (5)
12:30 PM	32 (2)	12348 (6)

Figure 20. Colour-coded performance indicator values based on the deciles

Based on the deciles, performance indicator values can be ranked and compared to each other, even if the performance indicator values are of a different type.

A drawback of decile-based colouring is the fact that anomalies in the data are not so easily spotted. This is because every performance indicator will have all the colours across the timeslots always. Even if a performance indicator has the same value constantly, the values will be colour-coded, and the same performance indicator value could be both green and red. A solution to this would be to make the colour-coding only based on the median value and difference from the median. Extreme colours could be used only for maximum and minimum values of the performance indicator, and the rest of the colours would be based on the difference between the performance indicator's value and the median value.

In the prototype implementation, both styles are utilized and the user test tells us which style would be the most useful. This colour-coding is then used in both PM visualization proposals, A and B.

4.3.2. PM view A: Performance indicator matrix

This proposal has PIs on the vertical axis and time on the horizontal. The PIs are colour-coded by the deciles, or the difference to median value, using methods presented in section 4.3.1. PIs are fixed on specific rows in a matrix so following a single performance indicator's value changes is easy. Columns in the matrix represent time slots and time slots represent the measurement instances. In the figure 21, the measurement interval is 30 minutes, i.e. PIs are measured every 30 minutes.

This design is very similar to the design of the FM visualization of "Alarm event view" presented in section 4.2.2. This design also utilizes the fisheye concept and text inside the performance indicator instance boxes. Text represents the performance indicator value in this case.

On the left-side of the chart the PI names are displayed and on the bottom the measurement time.

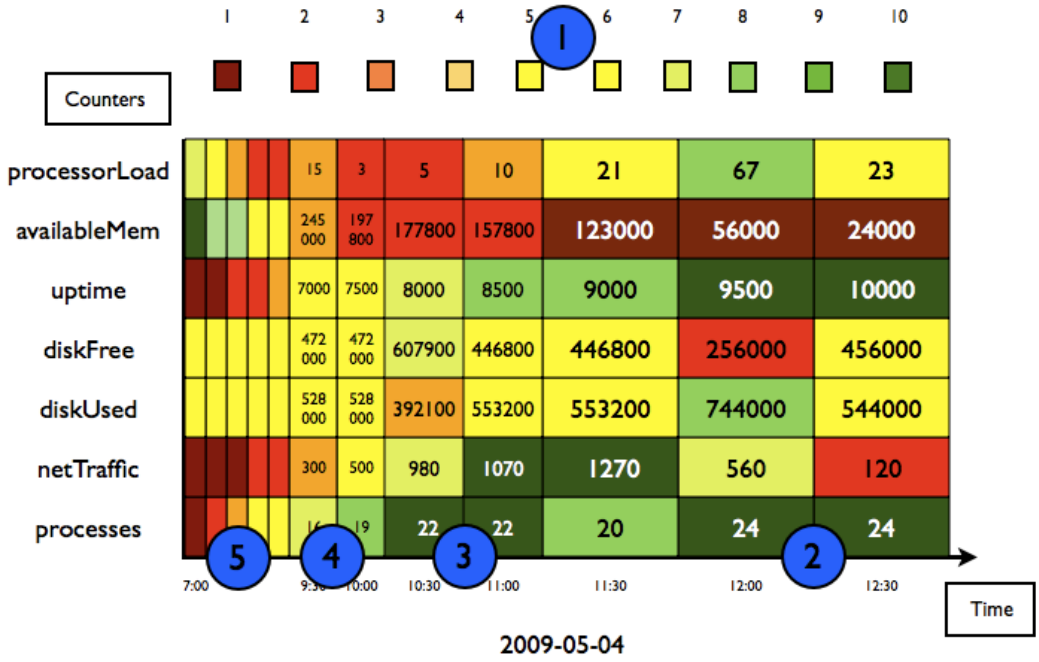

Figure 21. Concept proposal for "Performance indicator matrix" view for PM

Description of elements in figure 21:

1. The legend for PI colour codes. This is the same colour coding that was already presented in section 4.3.1. These colours should be configurable, for the same reasons that were also valid for the FM case.
2. Like in the FM visualization "Alarm event view", there is the fish-eye perspective. The data on the right side of the chart is the latest, and is highlighted by default. The highlighted section has the PI values shown in the PI boxes as well as the colours telling the viewer how a specific value compares to the median.
3. Slightly smaller boxes. Boxes which are next to the highlighted ones. The distance should also be configurable so that user could decide how the fisheye would react.
4. Even smaller boxes. These boxes are further away from the highlighted ones, and text uses a smaller font. Again, this distance should be configurable.
5. Smallest boxes, no PI values, just colours. These can be also made as small as one pixel in width, like in the FM case.

Displaying PI names and values limits the amount of displayable performance indicators. This can still be a handy view for some cases and when a look is

taken at the problems presented in section 2.4, actually all the problems presented there have been solved with this visualization.

By clicking a specific PI-instance, we should also be able to get a detailed-view of the PI value. Proposal of this view is presented in section 4.3.4.

4.3.3. PM view B: Reordered performance indicator matrix

In this visualization proposal, the reordered matrix concept developed by Jacques Bertin [1967, 1977] is being utilized. Reordering the PI matrix means only reordering the rows, columns are left untouched so that visualization can maintain the measurement positions on a horizontal axis so that spotting performance indicator trends is still possible.

Rows are reordered based on the decile so that the lowest decile performance indicators are located at the bottom of the axis and the highest deciles at the top. This way it is possible to see instances which are above the median value at the top of the visualization graph, and the instances which are below the median value at the bottom of the graph.

In the figures 22 and 23 the concept design can be seen.

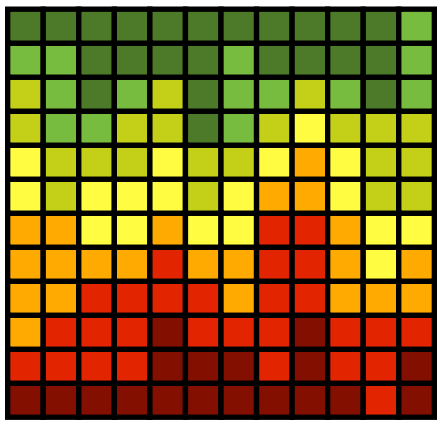

Figure 22. Concept of the proposal for “See entities” view for PM fragment

The decile is one ordering factor, but in the case of this visualization, it is not the only one, also an index value from median is calculated. So to help in ordering the values which are in the same decile, because there will be many, an index is calculated so that the base number is the median value. The index value for the median value is 100. It can then be calculated what is the percentual difference between PI values.

Let's say that there are two values with the same decile value. The first one is 35% greater than its median value, the other one being 40% greater than its median value. Therefore the index values of 135 and 140 assigned to those performance indicators. This means that the second PI with index value of 140 is displayed on top of the PI with the index value of 135.

When a user selects some instance, she should be able to see the other instances in the other time slots. In figure 23, there is one solution to this; by highlighting all the instances, a user should be able to tell if the overall value of the performance indicator has gone up or down or stayed the same over time.

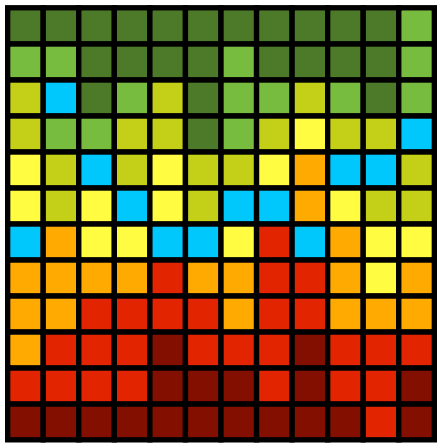

Figure 23. Situation when some performance indicator has been selected from the view

Some additional information of the specific PI can also be displayed by clicking the PI instance, as in the earlier visualization in section 4.3.2. This functionality is described in section 4.3.4.

4.3.4. Performance indicator details view

Figure 24 presents an example of what the details view could hold inside. All kinds of statistical information of the PI is important, but here the focus is on the current, median, maximum and minimum values of the performance indicator. Also, some kind of line chart can be drawn to describe more about the history of the PI's behaviour.

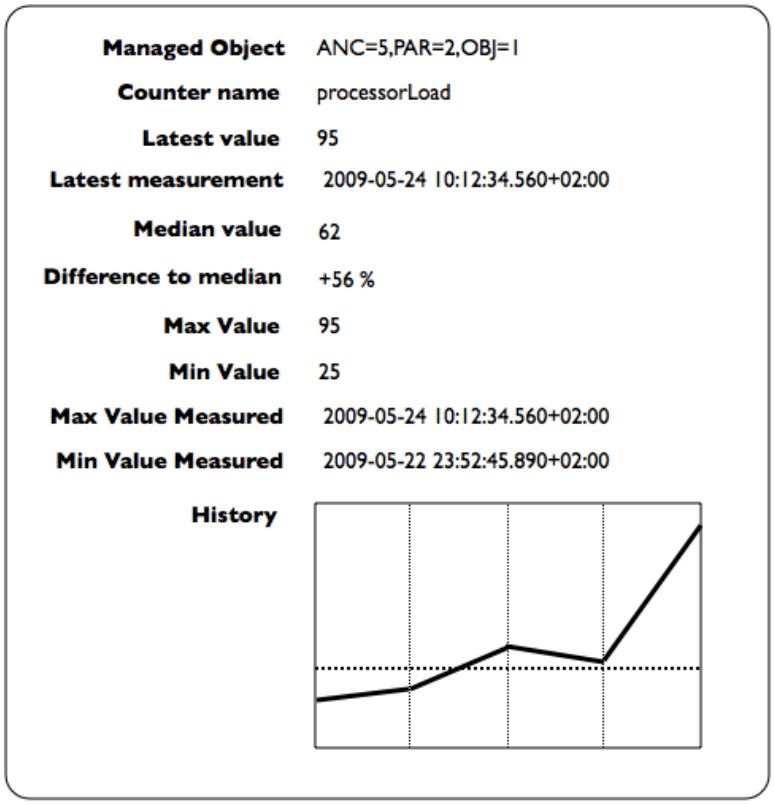

Figure 24. Performance indicator details view

The performance indicator details view can be opened from either one of the PM visualization proposals by clicking a specific performance indicator instance.

4.3.5. Thoughts about PM concept designs

With the visualization PM view B presented in section 4.3.3, the problem of comparing PI values is solved, but there is a drawback. Seeing trends in specific performance indicator instance becomes difficult, even though it depends on the figure 23's trend view implementation. Anomalies can be spotted if the levels for the median-difference scale are wisely defined. It should give a better overall picture of the managed entities and because the visualization on this level does not have any text or other detailed data inside, it is possible to fill the screen with pixel-based data and a lot of PIs from a huge data set. Data can be organized into a pixel bar chart-type [Keim et al, 2002] structure where at least tens of thousands of PIs can be displayed across time-slots.

The visualization PM view A in section 4.3.2 provides a more detailed view which would be suitable for closer examination of some specific performance indicator set. The filtering capabilities should be such that the user should be

able to pick interesting performance indicators into this view. This would make their comparison easy.

The question remains whether the decile-based view or the value difference-based view should be used for this view. There are two possibilities for the displayed data as shown in figure 25.

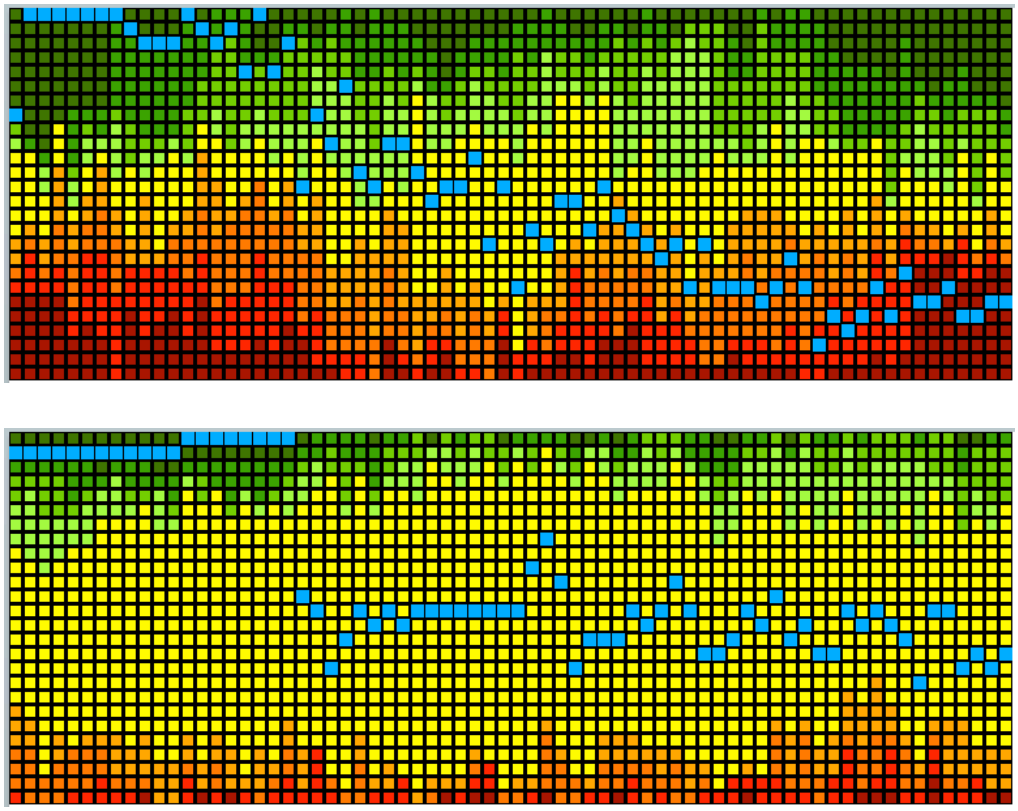

Figure 25. The same performance indicator presented in a decile-based visualization (upper graph) and in a median value-based visualization (lower graph)

In this work, the PM view B prototype is constructed and is used in the user test and the test feedback will indicate the success of the design.

4.4. Evaluation of the designs

4.4.1. User test setup

The user test was first planned to be held as a kind of feedback-form-driven test, which every participant would be able to do in their own time, and they would also try out the visualizations on their own. This approach was based on Catherine Plaisant's list of improving user tests [Plaisant, 2004] presented in section 3.2. This plan was quickly scrapped though, because it became clear that this would not provide useful data because testers would not be able to ask

questions or, most importantly, the test could not be moderated and the actual user reactions would go unnoticed. So the plan was changed to make a moderated, laboratory-like test event.

The test plan was made with separate PM and FM tests. A piloting test was conducted with two testers before the actual user test began. Based on the pilot test, two test cases were changed and the time for completing the test cases measured for future test plans. The final PM and FM tests had four actual test cases each. The test cases were mainly about spotting data in the visualization by using filtering capabilities of the prototype software. On the PM side, also a selection between two different alternatives, decile view and value view was asked to be made. The test situations also included time for gathering comments and giving feedback on the prototype and the test itself. Great importance was given to subjective opinions of the visualizations.

A meeting room was reserved for executing the tests. The test setup included an IBM Thinkpad T43 laptop which had a mouse as a pointing device and a data projector for the moderator for viewing the actual test progress.

The test tasks were printed on paper and given to the test participant to read as well as a form for collecting background data of the participant before the tests and a feedback form at the end of the test. The test tasks and feedback form can be found in appendix 1.

4.4.2. Tester profiles

A group of 10 people, consisting mostly of telecommunications specialists was invited to this moderated user test. All test subjects except two were experts in the network management field, so the user test was expected to give valuable feedback for the visualization designs. It has been stated by Jakob Nielsen in his book Usability Engineering [Nielsen, 1993, p. 166-169] that a group of 13 expert-level test subjects in a usability test is enough for 90% confidence of the usability test getting to +-15% of the standard mean value. In this test, one could say that there were 2 totally novice users and 8 expert users according to Nielsen. According to Nielsen, 8 expert users would give 90% confidence level with +- 20% of the mean value for this user test. And when taking the two novice users into account, the result is actually even better.

The test participants were asked just four background questions; background knowledge of PM and FM and years of experience with PM and FM.

Background knowledge was categorized into four categories (from least to most): None, some, basic and expert.

Gender	Title	FM knowledge level	FM experience (years)	PM knowledge level	PM experience (years)
Male	Telecom specialist	Expert	9.5	Expert	6
Male	IT specialist	None	0	None	0
Male	Computer science student	Basic	2	Some	2
Male	Telecom specialist	Expert	10	Expert	10
Male	Telecom specialist	Some	6	Some	0
Female	Usability expert	Basic	3	Some	0.5
Male	Telecom specialist	Expert	4.5	Basic	4.5
Female	Social science student	None	0	None	0
Male	Telecom specialist	Expert	5	Basic	2
Male	Telecom manager	Basic	6	Some	6

Figure 26. Test participant profiles

4.4.3. Test execution

User tests were executed within a timeframe of two weeks. Each test event took approximately one hour to complete, with shortest being about 40 minutes and the longest being 2 hours.

The testing situation always followed the following script:

1. Introduction to the FM prototype application
2. FM test tasks
3. Discussion about tasks and the FM prototype application
4. Introduction to the PM prototype application

5. PM test tasks
6. Discussion about tasks and the PM prototype application

Overall, the FM tests took less time than the PM tests. The PM prototype introduction took more time than FM and especially explaining the decile view in PM prototype raised some questions and was time-consuming. Discussion after the tests took more time than the actual test tasks, but was very fruitful and helpful as can be seen in the result analysis. The time used for each test task was measured during the tests using the clock on a mobile phone.

5. Discussion

5.1. Test result analyzation

Both the PM and FM tests included four test tasks. Tasks were rated as a success or failure by sometimes pretty subjective judgement. The test tasks and complete test results are in appendices 1-3.

Task 3 in FM tests had actually three expected findings and all were found, although one finding was seen only by one participant. The finding was that a specific alarm triggered other alarms. The task was marked as a success even if the participant did not see this. If both of the two other findings were found, the task was marked as a success, and if one of them was found, it was marked as a failure. The final failure rate of 50% for that task indicates the high complexity of the task.

Task 4 in PM tests could have never been a failure, because it was just a decision call, so its failure percentage is 0. All other tests could have failed and it is noteworthy that task 1 of PM tests was never a failure. The reason for this might well be that the task was fairly easy and PM tests were always performed after the FM tests, so test participants had already "warmed up".

	FM Test Task Failure Rate	Average execution time	Minimum execution time	Maximum execution time
Task 1	20.00	4.00	1.00	15.00
Task 2	30.00	2.07	0.20	5.00
Task 3	50.00	3.05	1.00	6.00
Task 4	20.00	1.48	0.33	3.00

	PM Test Task Failure Rate	Average execution time	Minimum execution time	Maximum execution time
Task 1	0.00	3.05	1.50	6.50
Task 2	20.00	1.80	0.20	5.00
Task 3	10.00	4.10	2.00	7.00
Task 4	0.00	5.80	2.50	9.00

Figure 27. Test task failure rates, average and minimum & maximum execution times (in minutes), FM test tasks in the upper table, PM tasks in the lower table

The average execution time for each test task is visible in figure 27. It can be seen that overall, PM tasks took longer to complete than FM tasks. Task 4 caused a lot of thinking and discussion and did not have a straight numerical answer, and may explain the really long execution time. FM task 4 and PM task 2 were fast and easy to execute, and this is shown in the statistics. In both

visualizations, the first tasks took longer, which might be explained by the learning curve; it took some time for participants to get familiar with the visualization.

There were big differences between the test participants' behaviour; others wanted to start a conversation during the test cases which naturally made execution of the task take longer. Others just quickly performed the tasks and their execution time was significantly lower. Because of those differences, comparing the execution times between participants is maybe not so beneficial.

FM test tasks were seen overall as being easier than PM test tasks, as can be seen in figure 28. However, an interesting finding is that although the FM tasks were seen as easier, the failure rates for the tests were greater for FM tasks, which can be seen in figure 27. This might be due to the fact that the PM prototype application was seen overall as more complex and more difficult to understand for test participants. This was repeated in many participants' comments.

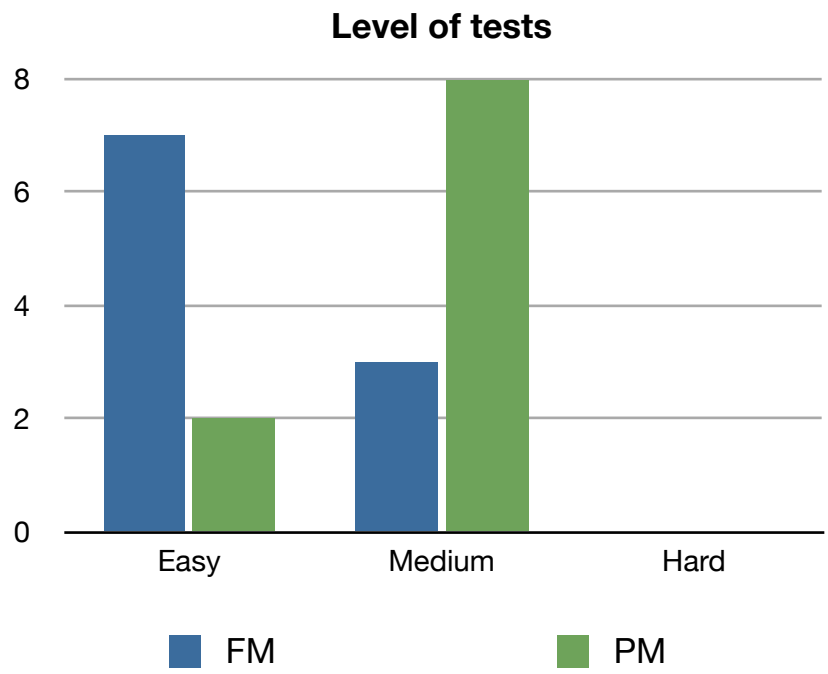

Figure 28. Participants' feedback on test task difficulty

It is also noteworthy that none of the test participants saw any test task as hard. This might mean that the test tasks were too easy, but on the other hand, test participants did not feel frustrated or angry at the prototype application, which might explain the pretty good overall score got in the user satisfaction rating.

PM tests had the task of comparing the decile and median-based value views. The result can be seen in the figure 29; 30% of the test participants (3/10 people) saw the decile view as better than the median-based value view. Most people had difficulties in understanding the whole decile concept, and others who understood it, had a hard time in picturing real use cases for it. The test result is quite clear; the default view should be based on a median-based value comparison.

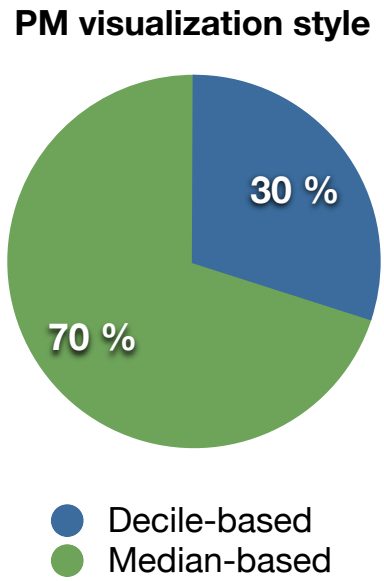

Figure 29. PM test task 4: Which one do you prefer, the decile-based view or the median-based view?

In the feedback form, each participant also gave an overall usefulness rating (0-10). Usefulness was explained as meaning the usefulness of the whole visualization concept, not so much the prototype itself. Because the expectation that the usefulness rating would vary between more experienced professionals and novices, the graphs in figures 30 and 31 have also ratings for the expert users separately.

Expert users in these graphs mean participants who filled their background knowledge being as either basic or expert level. If the user filled her background knowledge as being none or some, she did not qualify as an expert user.

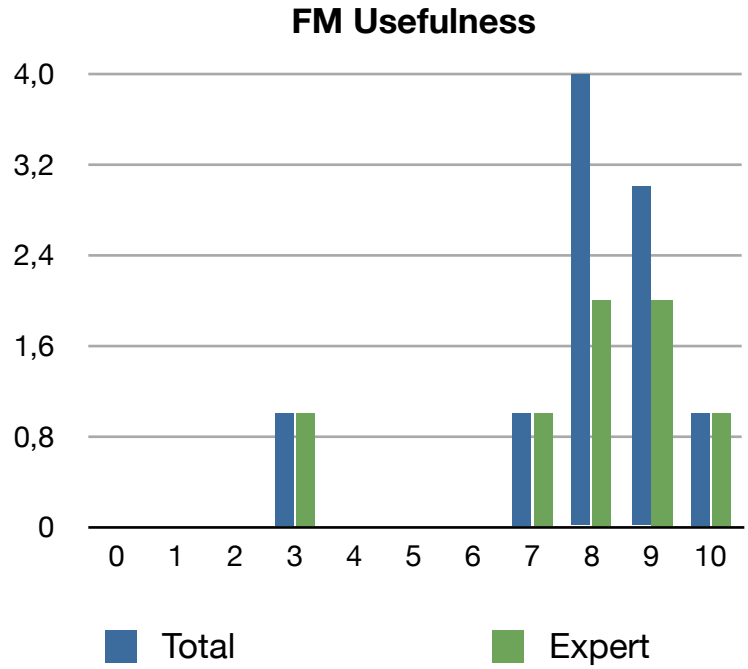

Figure 30. FM visualization concept usefulness rating from all participants

The FM visualization got excellent feedback from all participants and especially from expert users. One test participant was not satisfied with the visualization at all and rated it as 3, but others were practically excited about the new approach that the fisheye-based visualization offered for the FM data. It became clear that FM visualization has already at this phase a big potential for productization.

In the FM usefulness rating, two participants actually filed a score of 8.5. In this report, however, only non-decimal values are used, and therefore those ratings were considered as 8. Without this change the mean scores would be even higher than those presented.

The mean values for FM usefulness score were:

- Mean value for all participants: 8.0
- Expert user mean value: 7.71

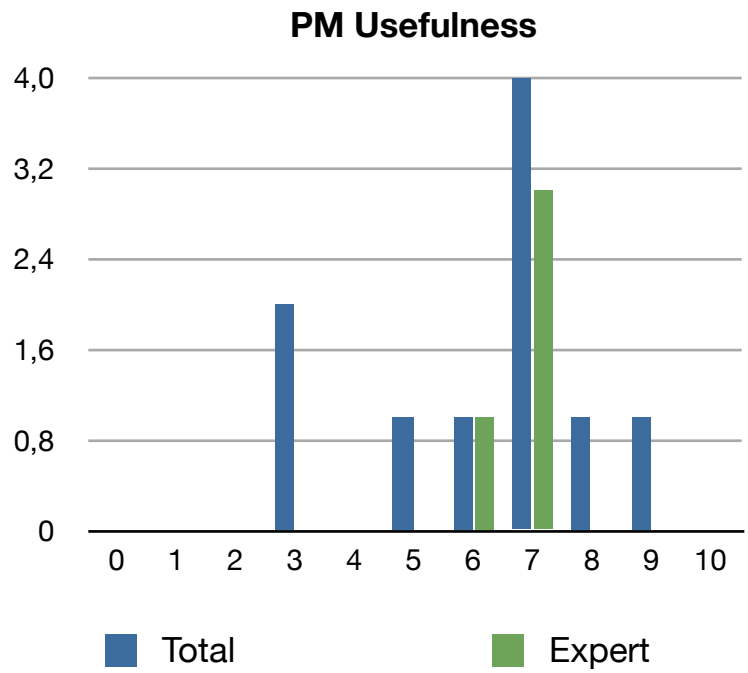

Figure 31. PM visualization concept usefulness rating from all participants

The PM visualization feedback was a bit more modest but still fairly good. The level of expertise in the group of participants for PM was lower than FM, only four test users qualified as expert users. It is noteworthy that two users gave the PM visualization a rating of 3 and one participant who gave 9 said that if the visualization would have only the decile view, he would give 8. Expert users still gave pretty good score, 3 users gave 7 and one gave 6.

The mean values for PM usefulness score were:

- Mean value for all participants: 6.2
- Expert user mean value: 6.75

Test task execution and especially the discussion revealed a big set of future improvement items. Meanwhile, the discussion and the whole testing event revealed its importance for prototype development work. This was the first iteration of development and it became clear that both prototypes could be easily improved a lot for the next one. In the next iteration, applications would be much more suitable for a bigger audience and would gain a better rating when developers would focus on the issues raised by the test participants.

5.2. FM prototype comments and observations

Because of the big amount of telecommunications specialists among the test participants, there were a lot of comments and suggestions for future releases

recorded during testing. Some comments repeated pretty often and here is a look on the most repeated and most interesting observations.

Besides the comments, during the testing it was noted that test participants had some difficulties during the test execution which they did not mention verbally but which were pretty clear when test task execution was observed. Also those observations are recorded here.

On the FM side, the most repeated comments were (in order of popularity):

1. An alarm history for a single alarm should be more clear and/or highlighting of alarms should be visualized better (6 participants)
2. How to handle many (thousands of) alarms in this kind of visualization (5 participants)
3. Fisheye is too fast or too sensitive, should be adjusted to be less aggressive (5 participants)
4. Alarm amount or a summary of alarms should be presented (3 participants)
5. Current alarm table situation should be visualized better (3 participants)
6. Specific problem code (and alarm's severity) should be also filterable (3 participants)

Other comments that were not so often repeated but which were interesting:

1. "Everything has to be clicked to see the alarm information"
2. "Font should be configurable and default colours should be changed to more clear ones"
3. "This is not useful to any operator"
4. "For managing and reporting purposes, this is good"
5. "Interesting approach, but what would an operator think?"
6. "There is potential"

Observations included the following:

1. Almost all participants had big problems when trying to count the amount of alarms in FM task 4. Problems were mainly because of the fisheye sensitivity and the rapid mouse movements. The fisheye sensitivity also caused other problems all the time during testing
2. Participants did not usually realize that they have to scroll the whole visualization through in tasks 1 and 3, they usually stopped when

they found the first instance of the alarm which was required to be found.

5.3. PM prototype comments and observations

On the PM side, the comments were overall more scattered than on the FM side, but still there was some consistency. The most repeated comments were (in order of popularity):

1. The colours in the visualization should be based on absolute performance indicator values which could have configurable thresholds rather than the difference from the median or decile (4 participants)
2. User should be able to highlight the performance indicator value in the graph by clicking the line diagram and vice versa (4 participants)
3. Comparing different kinds of data is not so useful or important (3 participants)
4. The decile view is difficult to comprehend (3 participants)

Here is a set of single comments that were interesting and noteworthy:

1. "I doubt that is this useful at all"
2. "If situation is always good or always bad, it does not show in the visualization"
3. "Visualization is better when there is more data, not just single lines"
4. "In some cases colours should be reversed, a bigger performance indicator value does not mean usually a good thing"
5. "The absolute values are what operators are interested in"
6. "The median-based value view is understandable, but the decile view is not"
7. "The decile view corresponds more clearly to the line diagram"

The observations that were made during the tests:

1. Participants needed to usually click the performance indicators separately in the graph to find the one that they were searching for. This took a long time and tests would have been executed much faster with some clearer indication of a specific performance indicator

2. A single performance indicator-view turned out to be a failure at least in the median-based value view. Test participants were using it when they found it but seeing performance indicator value trends with that was almost impossible in some cases
3. A lot of participants had a hard time understanding that performance indicators are jumping from one row to another, when for instance two performance indicators have been selected for the graph. This happened usually in task 3 of the PM tests; participants selected both cpu0 and cpu1 performance indicators for the graph and did not understand the view at first.

5.4. Top usability problems based on the test

Based on the user test, the most critical usability problems can now be identified for the prototypes.

Top usability problems of the FM prototype:

1. Fisheye sensitivity is too great, mouse moves too quickly from one bar to the next, it is difficult to focus on a certain alarm
2. Searching for a single alarm (and its states) is difficult
3. Alarm amount numbers are difficult to see
4. The overall and current alarm situation is difficult to see

Top usability problems of the PM prototype:

1. Performance indicators are difficult to spot from the graph; the user has to click the boxes to know which performance indicator it is
2. The single performance indicator view is pointless
3. Colour-coding based on median value or decile is not useful if the performance indicators do not present same kind of data
4. There is no interaction with the line diagram and graph

5.5. Results and possible future work items

The test was a good and helpful experience for the prototype development and during the testing it was easy to understand Nielsen's [Nielsen, 1993] observation that even 5 expert-level test participants could be enough for some projects. After 4-5 tests it became already clear what the issues might be for the

visualization. During prototype testing, the big amount of test participants might be unnecessary.

In overall, based on the feedback, the FM prototype seemed to be much closer to a productized tool than the PM prototype. One could see from the results that the FM prototype as such was already pretty much approved, and by focusing on the top usability issues for the next version, it could be already very close to a final visualization product which would be useful. The stacked boxes of alarms was an accepted approach.

The PM prototype on the other hand, was a failure as such, and the median-value based visualization and especially the decile-view based did not get so good feedback, although it was said that they have some potential. It seems that at least the decile-based view should be dropped completely, and the next version should focus on a performance indicator-by-performance indicator threshold-configurable value approach. The median-based approach could still be an alternative in the next version. The work should also focus on how to make performance indicators more visible in the graph straight away without the need for the user to click them to realize what they are. It seemed that the concept of having performance indicators as coloured boxes shown across time was an accepted approach, and so the next version could still utilize the same reordered matrix approach but with the tweaks presented earlier.

In section 3.2, Bertin's efficiency term was introduced as being the information visualization evaluation principle. This user test did not take the efficiency into consideration because it would require comparison between two or more optional visualizations. Now the only comparison that was made was between the decile and median-based views in the PM test. To be able to say whether the visualizations were effective or not, one should also have the older visualizations and perform the same user tests for both visualizations. Now there is no definite answer how these visualizations perform, although there are some subjective opinions about it by the test participants.

How do these presented designs then address the problems stated in section 2.4? The design ideas that were not covered with the prototypes were at least the large scale data and pixel-based visualizations. Prototypes based on those ideas might be constructed in the future.

The following lists the FM related research questions and discussion on how they were solved in this work:

1. Capturing the number of alarms over time
   - Based on user tests, this was solved well. At least it was easy to spot the time when the most alarms were received. There were problems in calculating the amount of alarms, and this should be improved in the future.
2. Are there any patterns for specific alarms?
   - The user tests proved that this is badly implemented; it is not easy to spot any patterns with the current FM design. This could be improved with some highlighting technology.
3. Are some alarms happening more frequently than others?
   - Again, with some highlighting of similar alarms, or gathering a summary, this could be solved. With the current design, it is not solved.
4. Can visualization find explanations for some alarms?
   - No, visualization did not offer any solution to this.

So on the FM side problems, only one real problem was solved and even that only partially. That was a bad result for the initial purpose. Still the overall usefulness score given by test participants is so good that even with this result the FM design is easy to accept for future development.

Next the PM research questions and the discussion:

1. Change of performance indicator values over time
   - This seemed to be a case of problem solved. The user tests proved that test participants were able to spot performance indicator value changes and highs and lows from the performance indicator matrix, although improvement is needed for specific performance indicator spotting.
2. Are there some trends in performance indicator values?
   - This was not directly tested in user tests but line diagrams were used and trends were spotted easily in at least the median-based view.
3. Are there anomalies in performance indicator values?
   - With the current design and implementation of median-based value view, it is not possible to spot anomalies, at least not so easily. But with configurable thresholds for performance indicators, it would be possible.
4. How to compare performance indicator values in a meaningful way?

- In the user test it was soon noticed that the performance indicator comparison was not seen as very important. At least comparing two performance indicators which were completely different was not seen as useful at all. In the user test it was still brought up that it would be convenient to be able to compare similar performance indicators from two or more network elements, for instance. For that purpose, visualization was seen as pretty useful.

Although the PM visualization did get a worse usefulness score than the FM, it still addresses better the initial problems that were issued for it. The PM visualization test did suffer from the fact that test participants had more knowledge of FM than PM. In addition, the PM visualization is not so self-explanatory from the start as the FM is.

6. Conclusion

The goal of this thesis was to construct new designs for visualizing network element FM and PM data which would solve at least some of the problems with the current O&M visualizations.

It can be seen in the visualization examples presented in section 3 that the earlier FM and PM visualizations are simple and modest. Visualization in earlier times of telecommunication network management systems has just meant bar charts and line diagrams. With the information amounts that the management systems gather from network elements, there are many possibilities for new types of visualizations.

This work includes some design ideas based on relatively old technologies. It is worth mentioning that the Jacques Bertin's reorderable matrix approach presented in the PM visualization design dates back to 60's. The fisheye view or distortion + focus approach used in both the PM and FM designs dates back to 1982. Basic concepts are old but decent applications for these old technologies have been still missing, at least in the telecommunication field. In the field of computer applications in general, these technologies have been used already, and although display technologies, interaction methods and raw computing power are all progressing, these both tehcnologies will still see much improvement in the future. Perhaps the improvement in computing power will create applications based on old ideas which the original inventors could not even imagine.

The network management system usage model is such that it requires professional personnel to operate it. This may be one of the reasons why information visualization and user experience in general has been more or less overlooked in network management applications, as people already know what to do and how to do it even with complex systems, no matter how clumsy or hard it is. Information visualization applications might suffer from good looks; expert-level personnel do not generally care how the management system looks, the only thing that matters is whether it is getting the job done or not and how fast. Some users might have an initial attitude problem against anything visual which they see as either childish or too flashy.

This means that the design of the visualization is very important. A new visualization design has to be generally accepted. Interaction has to be fluent and intuitive. Visualization marks have to be intuitive and clear. Also, most importantly, visualization needs to improve the performance so that there is the

justification for the user to start using the new information visualization application.

With great and efficient O&M data visualizations, a network management system can also distinguish itself from the competition and provide value to the software developing company in the form of income. Because everything visual is easily noticed and remembered, they can be very powerful marketing aids.

The contribution of this thesis work is to provide a starting point for developing PM and FM visualizations. The user test proved that the designs presented here do have potential. The validity of the user test may be questioned because in this case the test moderator and visualization designer were the same person. This fact might have caused the fairly high usefulness score. Also it was not clear were the comments totally honest, and was everything said out loud in the test event. Validity could be increased by having a separate test moderator, because test participants' comments would then go un-influenced.

All in all, the positive attitude of test participants and their comments about the prototypes proved that there really is a need for new visualizations, and the prototypes already are a good first step. Further research should focus on improving the anomaly detection in performance management and on conducting user tests with actual operator personnel to get more correct feedback.

References

[Adobe AIR] *Adobe AIR*, http://www.adobe.com/products/air/

[Adobe Flash Player] *Adobe Flash Player*, http://www.adobe.com/products/flashplayer/

[Adobe Flex Builder] *Adobe Flex Builder 3*, http://www.adobe.com/support/documentation/en/flex/

[Becker *et al*, 1995] Richard A. Becker, Stephen G. Eick and Allan R. Wilks, *Visualizing Network Data*, IEEE Transactions on visualization and computer graphics, vol 1, no 1. 1995.

[Bertin, 1967] Jacques Bertin, *Semiology of Graphics: Diagrams, Networks, Maps*, Editions Gauthier-Villars, Paris, 1967.

[Bertin, 1977] Jacques Bertin, *Graphics and Graphic Information Processing*, Flammarion, Paris, 1977

[Burns *et al*, 2001] L. Burns, J. L. Hellerstein, S. Ma, C. S. Perng and D. A. Rabenhorst, *Towards discovery of event correlation rules*. IBM T. J. Watson research center, Hawthorne, New York, USA, 2001.

[Card *et al*, 1999] Stuart K. Card, Jock D. MacKinley and Ben Shneiderman, *Readings in Information Visualization: Using Vision to Think*, Academic Press, 1999

[Cooper *et al*, 2007] Alan Cooper, Robert Reimann and David Cronin, *About Face 3, The Essentials of Interaction Design.* Wiley Publishing, Inc., Indianapolis, Indiana, 2007.

[FCAPS] *IEC Online education: Element Management Systems.* http://www.iec.org/online/tutorials/ems/topic03.asp

[Greenfield, 2009] Ely Greenfield's implementation of *Fisheye component, version 0.3*. http://www.quietlyscheming.com/blog/components/fisheye-component/

[Holopainen and Pulkkinen, 2006] Martti Holopainen and Pekka Pulkkinen, *Tilastolliset Menetelmät*, WSOY, 2006

[Inselberg and Dimsdale, 1990] Alfred Inselberg and Bernard Dimsdale, *Parallel Coordinates: A tool for visualizing multi-dimensional geometry,* Proceedings of IEEE Conference on Visualization '90, IEEE 1990

[ISO] International Organization for Standardization. http://www.iso.org/iso/home.htm

[Kazoun & Lott, 2008] Chafic Kazoun & Joey Lott, *Programming Flex 3,* O'Reilly Media, Adobe Developer Library, 2008.

[Keim *et al*, 2002] Daniel A. Keim, Ming C. Hao, Umesh Dayal and Meichun Hsu, *Pixel bar charts: a visualization technique for very large multi-attribute data sets.* IEEE, 2001.

[Keim *et al*, 2006] Daniel A. Keim, Tilo Nietzschmann, Norman Schelwies, Jörn Schneidewind, Tobias Schreck and Hartmut Ziegler, *A Spectral Visualization System for Analyzing Financial Time Series Data,* Eurographics, IEEE-VGTC Symposium on Visualization, 2006.

[Myers *et al*, 2002] Michelle Myers, Roy Sterritt, Edwin P. Curran and Hongzhi Song, *Exploring visualization of complex telecommunications systems network data*. School of information and software engineering, Faculty of informatics, University of Ulster, IEEE 2002.

[Nielsen, 1993] Jakob Nielsen, *Usability Engineering,* Academic Press, 1993

[Norman, 1988] Donald A. Norman, *The Design of Everyday Things,* Doubleday, 1988

[NSN, 2009] Discussions with Nokia Siemens Networks Adaptation Technology architects, February-May 2009.

[Plaisant, 2004] Catherine Plaisant, *The Challenge of Information Visualization Evaluation,* Proceedings of AVI '04 - ACM Conference on Advanced Visual Interfaces, ACM Press, 2004

[Qeli, 2004] Ermir Qeli, Wolfgang Wiechert and Bernd Freisleben, *Visualizing Time-Varying Matrices Using Multidimensional Scaling and Reorderable*

Matrices, Proceedings of the Eighth International Conference on Information Visualisation (IV '04), IEEE 2004

[RFC 1485] S. Harcastle-Kille, *RFC 1485 - A String Representation of Distinguished Names (OSI-DS 23 (v5))*, Network working group, ISODE Consortium, July 1993

[Siirtola, 1999] Harri Siirtola, *Interaction with the Reorderable Matrix,* Proceedings of the International Conference on Information Visualization 1999, IEEE 1999

[Siirtola, 2003] Harri Siirtola, *Combining Parallel Coordinates with the Reorderable Matrix,* Proceedings of the Coordinated & Multiple Views in Exploratory Visualization, IEEE 2003

[Simon, 1996] Herbert A. Simon, *The Sciences of Artificial*, Cambridge, MA, MIT Press, 1996.

[Spence, 2007] Robert Spence, *Information Visualization, Design for Interaction, 2nd edition*. ACM Press, 2007.

[Spence & Apperley, 1982] Robert Spence and Mark Apperley, *Data base navigation: an office environment for the professional.* Behaviour and information technology, vol 1, no 1, p.43-54, 1982

[Tang *et al*, 2008] Yongning Tang, Ehab Al-Shaer and Raouf Boutaba, *Efficient fault diagnosis using incremental alarm correlation and active investigation for internet and overlay networks*. IEEE Transactions on network and service management, vol 5, no 1, march 2008.

[Valiati, 2008] Eliane R. A. Valiati, Carla M.D.S. Freitas and Marcelo S. Pimenta, *Using Multi-dimensional In-depth Long-term Case Studies for Information Visualization Evaluation,* BELIV'08, ACM, 2008

[Wang Baldonado *et al*, 2000] Michelle Q. Wang Baldonado, Allison Woodruff and Allan Kuchinsky, *Guidelines for Using Multiple Views in Information Visualization,* ACM Advanced Visual Interfaces '00, 2000

[Ziegler *et al*] Hartmut Ziegler, Tilo Nietzschmann, Daniel A. Keim, *Visual Analytics on the Financial Market: Pixel-based Analysis and Comparison of Long-Term Investments.*

FM User Test

1. Basic knowledge of FM (None/Some/Basics/Expert)
2. Years of experience in NMS
3. Task 1: When is the alarm with the specific problem 66 triggered on 'MACHINE=4'?
4. Task 2: When is that alarm cleared?
5. Task 3: Can you make any other observations regarding that alarm?
6. Task 4: At what point in time is there the most number of alarms and how many?
7. Were these questions (Easy/Medium/Hard)
8. Usefulness of the visualization IYO: 1-10
9. Comments, critics, suggestions

PM User Test

1. Basic knowledge of PM (None/Some/Basics/Expert)
2. Years of experience in NMS
3. Task 1: Examine the performance indicator values of performance indicator 'MACHINE=1,OBJ=performance indicator_5'. When does the performance indicator reach its maximum value?
4. Task 2: When does the performance indicator reach its minimum value?
5. Task 3: Examine performance indicators OBJ=cpu0 and OBJ=cpu1 on MACHINE=1, what observations can you make?
6. Task 4: Compare the performance indicators by value and performance indicators by decile views, which one do you prefer to use?
7. Were these questions (Easy/Medium/Hard)
8. Usefulness of the visualization IYO: 1-10
9. Comments, critics, suggestions

FM User Test Result Matrix

	A	B	C	D	E	F	G	H	I	J
Basic knowledge	Expert	None	Basics	Expert	Some	Basics	Expert	None	Expert	Basic
YoE	9,5	0	2	10	6	3	4,5	0	5	6
Task1	Mon 24.1., 2010	4.1.2010	10.01	Jan 31	Feb 1	Jan 11	Jan 12	Jan 13	Feb 12	27.1.
Task1 time	5	1	2	5	3	15	5	1	2	1
Task1 success	yes	yes	yes	no	no	yes	yes	yes	yes	yes
Task1 comments		Did not see the other instance (in task 3 did)	Found first others, noticed only later that clicking is possible	Got the last alarm in the viz	Did not see the previous alarms at all	Filters the MACHINE=4, had to explain the behaviour of the alarm table (that it contains history)	Filters MACHINE=4	Clicks MACHINE=4	Filters MACHINE=4, found the alarm from the last instance	Filters MACHINE=4 first,
Task2	Jan 7	25.1.2010	14.1.	cleared 6 times (after every alarm triggering")	Next day	Feb 1	Jan 16	Jan 17	Feb 15	30.1.
Task2 time	4	1	1	5	0,2	1	3,5	2	2,5	0,5
Task2 success	yes	yes	no	no	yes/no	yes	no	yes	yes	yes
Task2 comments		Did not see that the alarm was cleared earlier already	user did not realize that the alarm was cleared earlier	user thought that alarms are cleared after every instance in the viz	Did not see the previous alarms at all		Did select the following day, not the actual clearing day	Watches first only the first column, selected the next column where it was not anymore	"Where do I see the clears",	
Task3	State has changed from Major to critical and back (two times)	State has changed, also noticed that has happened two times with the same pattern	State has changed, noticed only the first one	If the alarm instance is same color it has been cleared between the instances	Major alarm	severity is changing	State changes from major to critical and back to major, and happens again	Alarm's state changed from major to critical, nothing was done for two days, was dropped back to major and cleared, did see the related alarms	Has been triggered two times, severity has changed from major to critical and back	Severity has changed, from major to critical and back
Task3 time	4	3	2	6	1	1	4	5	3	1,5
Task3 success	yes	yes	no	no	no	yes	yes	no	yes	no
Task3 comments	Did not find the related alarms	Did not find the related alarms	Did not find the related alarms or the other instances of alarm		No state changes or anything else	Did not see the related alarms	Did not find the related alarms	Did not see the following alarms	Did not see the relations, otherwise perfect	Did not see the related alarms or other instances in the end
Task4	Jan 24 and 25	23.1.-24.1.	11.12. 13 alarms	3 bars, 14 alarms	14, 2 bars, 12 and 13 Jan	12 Jan, 13 alarms	Jan 13, Jan 14, 6 alarms	Feb 1, 2 and 14 alarms	4 days and 11 alarms	28.1. and 14 alarms
Task4 time	1	1	1	2	0,33	1	2	3	2	1,5
Task4 success	yes	no	yes	yes	yes	yes	yes	yes	no	yes
Task4 comments		Had the MACHINE=4 filter on, did not see all the alarms, did complete the task after instructions						Has to stop the mouse movement and count the alarms by hand	Finds first only MACHINE=4 related, corrects when noted	
Level	Easy	Medium	Easy	Medium	Easy	Medium	Easy	Easy	Easy	Easy
Usefulness	8,5	8,5	9	3	7	10 (manager), 3 (operator)	9	8	8	9
Comments	Mouse is moving the fisheye, although it is not "in" the graph	Mouse cursor should be an arrow	Other same alarms should be also highlighted in the viz	Difficult to understand the changes in alarm states	Realizes quickly how to move in the viz	severities don't necessarily change	Could somehow show the wanted alarm	Difficult to say because of lack of knowledge	How to handle many alarms	Did not first realize that box contains specific problem code
	There is a confusing additional row in the visualization's date-field (visible in the starting state of the viz)	Too much sensitivity in fisheye	Currently active alarms should be more clearly visible	Fisheye is way too aggressive and too sensitive	Default critical color is bad	history data not necessarily interests the operator	Should highlight all the other alarms in next timeslots	Should be able to see single alarm more clearly	Interesting but what would operator think	Alarm amounts should be more visible (as numbers) in the side or somewhere else
	Should be able to see historical data	Instead of click, user should be able to select the alarm by just hovering over the alarm	Specific problem should be more clear, maybe bigger font	Difficult to see the text in dark red boxes, maybe default colors should be different	Critical values should be lower	specific problem code should be as filter	Day is too large set for single timeslot	Mouse is too fast	Fisheye scale could be resizable	Visualizing the changes happened to certain alarms could be better
	Specific problem code and severity should be also filters	There is potential	Specific problem filter should be present	Should be able to see alarm history more clearly	management controls should be present	Date filters should be visible in the view, and empty slots should be also displayed	Alarms should be sorted based on specific problems and object name	Keyboard usage would be useful	FM and PM data would be useful to unify	Maybe machines could be visualized with certain colors inside the boxes, could be more clear to spot
	Should not be only visualization, alarm table is still needed	Relationships between alarms are clear	Alarm situations for different machines should be better visualized, maybe by line diagrams under the viz	Font should be changed to more clear one	how it handles 40000 alarms? :)	More filters based on details (all details should be there)	How would the very big object name handled, if it does not fit to the box	Critical alarms should be lowest		"M" sign in the smallest boxes maybe a bit confusing
	Is there enough room for more alarms	There should be an option to go back to the starting state	Alarm amounts should be visible as numbers	Everything has to be clicked to see the alarms	more speed	could be more effective with for instance self-monitoring,	Should show the empty slots also			Details could be updated by hovering, not clicking
	Default colours should be different, white as warning and bright red as critical	Highlight should be more clear, there should be an indicator which alarm was selected		Lack of summary is bad, should be able to see alarm amounts		operator needs to react to alarms instantly	Useful for seeing the alarm history, or some specific alarm behavior			Maybe the selected alarm could be more clearly presented, and selection could be cleared with a separate button
				Date filter is not working, you don't know what to set if you want to see everything		maybe too much effort for real-time handling of the alarms	Makes more sense when filtering to some data based on interests			Maybe the current situation in the right side of the visualization, machine filter could be used there but date filter would not affect
						for history viewing and reporting purposes this is good	Date filters should also have time			Correlation between alarms should be visualized
						overall situation is the most interesting	Comparisons should be possible to do			Root alarms should be visible maybe causes not
						some regions are more important than others, priorities can change	Mouse sensitivity not so much an issue, maybe mouse settings should have been changed to slower before the tests?			Text color should be also reversed color
						Mouse is too sensitive	Maybe a line marker on where the week starts etc. Current small boxes on dates are bad to interpret			How does it scale to many more alarms
						How about thousands of alarms				

PM User Test Result Matrix

	A	B	C	D	E	F	G	H	I	J
Basic knowledge	Expert	None	Some	Expert	Some	Some	Basics+	None	Basics	Some
YoE	6	0	2	10	0	0,5	4,5	0	2	6
Task1	Jan 26	Nov 18	Feb 1	Dec 22	Nov 25	Feb 2	Nov 26, in the beginning	Dec 7	Dec 29	9.2.
Task1 time	3	3	2,5	1,5	3	2	6,5	4	2	3
Task1 success	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes
Task1 comments		User is clicking the counters separately for seeing the difference	Selected one counter to view, looking at the line diagram all the time, confused about the colors because everything yellow	User is checking the viz so that all counters of MACHINE=1 are in the graph	Comparing should be done in line diagram	User clicks the separate instances	Difficult to find the counter_5, if the MACHINE=1 is observed, uses the line giagram to spot the maximum value	Is bothered by the disappearing counter graph	Object names cause a bit of confusion, clicking all in order	Selects single counter into view
Task2	Dec 28	Dec 28	Jan 3	Jan 5	Feb 2	Nov 25	Jan 5	Dec 26, 28	Dec 16	27.12.
Task2 time	1	2	3	2	0,2	1	5	2	0,33	1,5
Task2 success	yes	yes	yes	yes	yes	yes	yes	no	no	yes
Task2 comments		User was using pen&paper to record the results, when going through the viz	Clicking the line diagram, commenting that hovering feature is easier than click	User is checking the viz so that all counters of MACHINE=1 are in the graph		"should I use the time filters?"	Clicking a lot	Did only watch the median, not the absolute value	Did watch the median, not the absolute value	
Task3	cpu1 value stays the same, Dec 24 cpu0 value goes over median	Overall load has increased (which is a correct observation but not exactly what was targeted)	cpu0 rising, cpu1 staying the same	cpu0 rising, cpu1 stable	cpu1 load has increased all the time, cpu0 has stayed the same	cpu0 14.12 smaller than median, bigger than median Jan 12, found the smallest and biggest values, cpu1 staying the same all the time	cpu0 rising, cpu1 staying the same	cpu0 first at low, then really high, cpu1 stays close to median, always the same	cpu0 value has increased all the time, cpu1 stays the same, matrix shows the value maybe incorrectly sometimes	Other one stays stable, other one is increasing
Task3 time	7	4	3	5	2	4	5	4,5	3	3,5
Task3 success	yes	no	yes	yes	yes	yes	yes	yes	yes	yes
Task3 comments	Decile view was used in this test, big difficulties in reading the visualization		Distracted about the counter instances moving to other line in the viz	User makes observation: Lowest box not necessary the lowest value in the counter array	Graph is totally useless		With two rows, maybe not so clear to see, with 10 counters maybe more clear			Selects two counters into view
Task4	value	value	decile	decile	value	value	value	value	value	decile
Task4 time	7	4	6	7	5	9	7	4	2,5	6,5
Task4 success	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes
Task4 comments	Decile view was pretty hard to understand	Value is more clear but it is not "telling the overall load"	"decile view corresponds more clearly to the line diagram"				"Decile view is difficult to comprehend"			
Level	Medium	Medium	Medium	Medium	Easy	Medium	Medium	Medium	Medium	Easy
Usefulness	7	8	7	6	3	5	7	9 (8 decile view)	7	3
Comments	You should be able to view counter instance in the line diagram by clicking it in the viz and vice versa	Shade should be taken out of the line diagram	Line diagram is more useful when checking just one counter	There is no sense in comparing counters with different types	Doubts that is it useful at all	Operator would be interested in seeing just counter value in specific time	Same counter from different managed objects, comparing would be much more useful	Cannot find the counters, selects the counter by hover, then starts to go through one by one	Measurement name and number and counter description are missing	Sees the value from the line diagram
	All selected counters should be also in the line diagram	Hovering the graph is maybe not so useful	Visualization is more useful when checking a bigger set of counters	Minimum and maximum values should be more visible	Line diagram is the only useful thing in the viz	if situation is always good or always bad, it doesn't reflect in the graph	Comparing different kinds of data not so useful	Colors don't tell you anything	Aggregating rules and equations should be visible	Diagram should be clickable
	Decile view is unnecessary	Selected day should be somehow better highlighted	User is clicking the visualization a lot	Time scale could be brushable from the line diagram	Overall confusing visualization	maybe some thresholds could be more useful than difference to median	Line diagram does not reflect the place in graph	Decile view maybe needed only for problematic counters	What about decibels	Has to click all the counter instances from the graph
	Color scale in value-view should be configurable	Clicking the graph should disable other selections	Prefers the whole visualization on all the time, thinks that one-counter-view is unnecessary, should be able to see all the counters all the time and maybe select an instance to the visualization by clicking the counters in the tree		Cannot understand what information the graph can bring	user-configurable thresholds	Line diagram has sometimes lots of empty space inside on Y-axis	Decile view maybe not so useful	Value view is understandable, decile view is not.	Maybe colors should be reversed
	Maybe there could be less colors (5?)	Clicking the graph should actually select the counter for the line diagram but other details would be updated by hovering the graph	Thinks that red color is deceiving in the cpu example, because gives the impression of error, although lower cpu usage is better		Lack of time made the test go maybe too fast	More explanations and labels, tooltips	Whole visualization is better the more there is rows of data		More PM information	Multiple selection is not intuitive, checkbox maybe better
	Large scale visualization problems -> One alternative could be to visualize SDNs based on the counter that has the most extreme condition there		Line diagram could be clickable or user should be at least able to see an indicator in the diagram when moving in the viz			Absolute values are what operators are interested in mostly			Value view could be really useful for testing network element integration	Is there any use in comparing the counter instances
						Line diagram should be updated when filter is selected			Absolute value could be more interesting than median	View should be used with similar counters to gain knowledge
						Operators should be able to define thresholds for values				Median could be replaced with some configurable value
										Decile view could be more descriptive
										Value contains too much yellow, decile view more calm to the eye :)
										There is potential, but thresholds should be possible to define
										Median approach would be good if similar counters would be compared