UNIVERSITY OF OSLO
Department of Informatics

An investigation of alternative methods for improving system monitoring visualization

Kjetil Dyrgrav

Network and System Administration

Oslo University College

May 24, 2011

An investigation of alternative methods for improving system monitoring visualization

Kjetil Dyrgrav

Network and System Administration
Oslo University College

June 29, 2011

Abstract

As mobile platforms become increasingly popular, the area of system monitoring would likely need to adopt some of the design methods and techniques for mobile usage, which includes exaggeration of size and importance, less complexity and perception enchanting techniques. This report investigates and suggests alternative methods for visualizing system monitoring data with main focus on mobile device displays. Alternative methods for system monitoring are identified and suggested. An experiment is conducted, where a classic monitoring tool is thoroughly compared to a new tool especially constructed for exploiting alternative methods of enchanting data. The data from the experiment proves a significant difference in findings when alternative methods are compared to the classic tool.

Acknowledgements

I would like to thank to my supervisor, Frode Eika Sandnes, Ph.D, for encouragement and guidance throughout the project. Sandnes has shown great confidence and been very patient. His insights and knowledge has been indispensable. Even though he is often seven minutes late for supervision hours, he makes up for it by being an amazing supervisor. I wish him the best of luck in his new position as R&D Pro-rector.

I would also like to thank Kyrre M. Begnum, Ph.D, whose idea for a mobile QR-code application for system surveillance led to this project. Begnum has been an inspiring force and a capacity throughout the master program. He has also been very helpful when times were rough. Begnum provided a virtual server for this project.

Finally, I would like to give thanks to my test subjects and the people who helped me acquire them. I really appreciate that you gave of your time to help out. You know who you are.

Kjetil Dyrgrav
Oslo, May 24th, 2011

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Members of the current information society are likely to take certain services for granted. A substantial part of any daily interaction between two or more members of this society is likely to happen via some sort of digital medium. Text messaging, social networks, chat clients, e-mail, VoIP, and most other digital communication services have replaced written letters, traditional phones and face-to-face contact.

Furthermore, inefficient paper dependent information systems such as centralized archives and libraries are virtually on the edge of extinction, constantly under threat of digital annihilation. Today, any modern society is highly dependent on digital services. Consequences may be severe if only one of these services become unavailable; even a limited downtime period may imply dire consequences.

Such services are often hosted in huge data centers. These computer clusters both serve society and its infrastructure, and a great deal of resources are spent on maintaining service availability. Because both server hardware and software are imperfect and failure at some point is unavoidable, precautions must be implemented to optimize server uptime - the ultimate goal, to bring downtime to zero. Needless to say, such systems need maintenance, and perhaps even more important, surveillance. Surveillance requires both information in real time as well as maximum accessibility for system administrators. This makes availability a key factor.

At the same time the use of mobile solutions and services has increased dramatically. Data from Stat Counter shows that the mobile portion of all web browser traffic in the world has increased from 1.7 percent to 4.5 percent in one year (from February 2010 to February 2011). Additionally, there is an uncertainty in terms of overall use of various mobile services, as it is reasonable to assume that mobile applications takes over a portion of the traffic that web browser based solutions used to have.

For instance, 321.000 people have downloaded Trafikanten's (The main provider of public transporation information in Oslo, Norway) mobile real time information application for Android / iPhone / Windows Mobile, despite the fact that there is a HTML version of the travel planner to use on mobile browsers (as well as a normal one).

While the Skiing World Championship unfolded in Oslo, during the winter of 2011, an average day generated 90.000 non-unique hits towards Trafikanten's mobile apps, while 94.000 hits were measured at the web site [1]. This suggests a percentage of 48.9 percent for real time information displayed exclusively on mobile devices. At the same time, 9 out of 10 of all mobile phones sold through NetCom (Norway) in 2011 are smart phones [2]; communication devices who promotes greater accessibility and usability for end-users.

It is reasonable to assume that an increasing number of solutions will be available on mobile platforms, both for practical and financial reasons, as demands for higher availability and efficiency increases. Operation critical systems, as well as monitoring and warning systems will seek to find and exploit the maximum communication performance potential within the framework provided by mobile platforms, as this is essential in such systems.

Mobile displays, features and technology may require different approaches and methods in order to get the message across. For example, on a smart phone users will often have to zoom in to see information intended for non-mobile displays, which could lead to the loss of information as parts of the information is opted out due to screen size limitations.

Graphical representation of time line data is, according to Edward R. Tufte [3], the most widely used visualization method, representing over 75 percent of the total number of visualizations. It's purpose is to make it easier to understand trends over time within a set of numbers. Line charts may vary greatly in appearance and complexity. In systems management, for example, it is common to see multiple sets of data plotted in the same chart, often with different methods of plotting. To make the graphic readable for humans, focus on explanations (legends), titles, guides and supplementary text is required.

Unfortunately, in system administration (such as air traffic control), the trial and error approach is an arguably often used method to avoid future crashes. Log files are investigated to determine how servers performed under and certain critical periods. If a server crashed, a log file or a system monitoring tool is often used in retrospect to analyze data. After data is analyzed and a conclusion has been reached, adjustments are made for the purpose of lowering the risk of a new crash; not unlike that of an air crash investigation. Still, a significant loss of data may have occurred, a break-in may have happened, or money or reputation (which are sometimes interconnected) may have been lost.

Real time visualization is important. It is hard to imagine driving a car safely without a fuel meter or a speedometer. These monitoring displays provide the driver with vital data about the state of the car and the current speed. The fuel meter indicates when you should pull over and refill the gas tank. If the fuel meter is broken, displaying a value which is misleading and wrong, the car's engine will inevitably run out of gas and stop. Consequences may be severe if this happens on a busy highway with heavy traffic.

If the speedometer or the fuel meter is in fact working, but designed in such a way that it is unable to communicate how much gas is left or how fast the car is driving, the situation would be identical to that of having broken instruments, because the outcome of would be similar. The car may run out of fuel, and lives may be in danger.

The Three Mile Island accident in 1979 [4] is up to this date known as the most significant nuclear power plant accident in America. A valve in the power plants cooling system got jammed, which caused coolant for the reactor core to escape. This led to a gradual rise in core temperature. Although this change in value was detected and displayed by indicators in the control room, no countermeasures were initiated by the control room operators. A valve status lamp was wrongly interpreted as indicating valve position, when in fact it indicated power to control the valve. Up until then, the lamp had been illuminated when the valve was open, which also corresponded with valve power. But when the valve got stuck in open position, controllers thought the valve was closed - because of the lack of lamp illumination. Because of this, investigators later concluded that the design of the indicator light was fundamentally flawed (Figure 1.1) and that its contribution to the accident was significant.

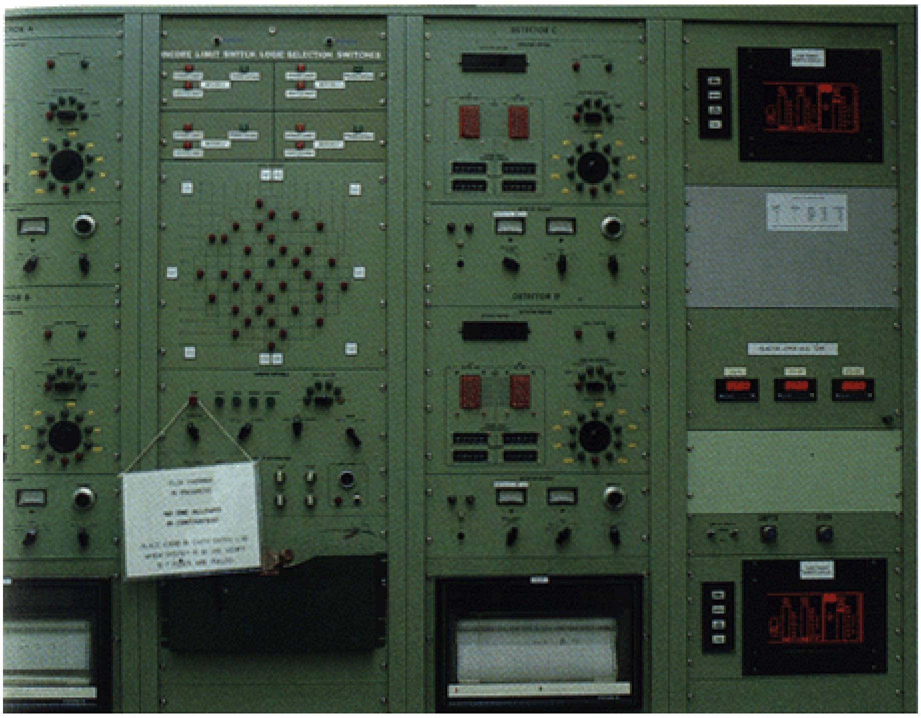


Figure 1.1: Inside a Nuclear Power Plant, by Charlotte Wilcox
(Source: Picture from the book "Inside a Nuclear Power Plant" by Charlotte Wilcox (1996))

In 1992, Flight 148, an Airbus A320 aircraft, crashed into a mountain, near Strasbourg Airport in France. At the time A320 had the most advanced cockpit in the world made by expert engineers. The conditions that day were cloudy and the pilots had only the cockpit instruments for guidance. Air crash investigators later concluded that a flaw in the design of the digital vertical speed indicator screen caused the pilots to misread. The same display was used for two descent modes, Vertical Speed Mode and Flight Path Angle Mode. By pressing a button next to the digital display, the pilots could switch between the two modes. No other indications than a comma inside the number display and some letters (Figures 1.2 and 1.3) not directly in context with the number display separated the vertical speed indicator from the other instrument.

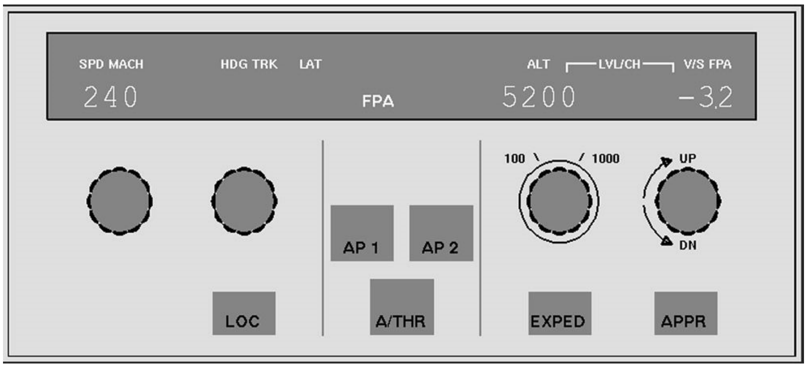


Figure 1.2: Airbus A320-111: Descent mode display in Flight Path Angle Mode (Source: Massachusetts Institute of Technology Department of Aeronautics and Astronautics)

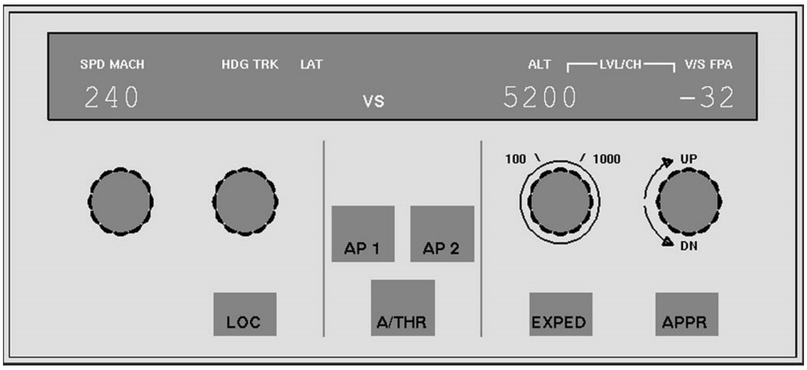


Figure 1.3: Airbus A320-111: Descent mode display in Vertical Speed Mode (Source: Massachusetts Institute of Technology Department of Aeronautics and Astronautics)

(Figures 1.2 and 1.3 shows the different modes) This simple design error led the pilot to believe he was in one mode while he was in another which caused the plane to drop from the air ten times as fast as intended. 87 of the 96 crew and passengers were killed.

In system administration (like in air accidents), the retrospective investigative approach is arguably a often-used method to avoid future crashes by various causes. Log files are read to find out what happened, and not until afterwards are precautions taken to prevent similar crashes or attacks. If these are servers are monitored, maybe they use a monitoring tool which performance entirely depends on the system administrator that installed and configured it.

There are likely tens of thousands of servers in the world today which hosts critical systems. These are presumably constant under surveillance, yet downtime occurs; hardware breaks down, attacks are made, and networks break down due to congestion. For a system administrator to become aware of critical states, it is essential that information is relayed as accurately and as effective as possible. It is also crucial that the information is available to the system administrator as often as possible. Hence, if the information is accessible via a web browser on a mobile device, it should be also be comprehensible and communicate critical states with high performance through that very platform.

1.1 Problem statement

The problem statement of this thesis is as follows:

Investigate and compare conventional and alternative methods for displaying and improving system monitoring visualization for the purpose of strengthening communication performance for mobile device usage.

Chapter 2

Background

Information designers such as McCandless and Holmes produce information graphics that is frequently requested by instances such as high circulation newspapers and magazines. Yet, their methods stand in deep contrast to the minimalistic methods of theorists such as Tufte and Few. Popular system monitoring tools is arguably conservative in the use of visual techniques, as they adopt classical graph tools with little or no data enchantments, embellishments or other visual means. As mobile platforms become increasingly popular, the area of system monitoring would likely need to adopt some of the design methods and techniques for mobile usage, which includes exaggeration of size and importance, less complexity and perception enchanting techniques.

This chapter will address techniques, methods and means used by both monitoring and mobile specific solutions and identify and discuss problems and benefits associated with these. Furthermore, previous research on real time data monitoring and mobile platforms will be investigated and discussed - in order to see if there are design elements that could be suitable in a system monitoring context. Also, challenges for establishing a methodology for visualizing real-time data as opposed to statis data is brough up. Finally, methods for measuring the information displays will be listed, for the purpose of investigating the possibility for test experimental system monitoring methods on a mobile device.

2.1 Design elements

2.1.1 Pictograms and Icons

System monitoring visualization on a mobile device should communicate information with a much clarity and efficiency in order to promote quick reaction in case of an emergency. Perhaps by applying different methods than the case is with desktop monitors or other types of information screens. The mobile context is arguably hard to grasp, but it nevertheless needs to be taken into consideration when designing for such screens. One of the means that researches have adapted for mobile usage such as menu navigation is the use of icons.

To give an example of efficient use of icons; in case of a small fire in a real world scenario, it is crucial to find a fire extinguisher immediately in order to limit the damage as much as possible. A few seconds may be the only difference between extinguishing the fire - and smoking ruins where the building once was. The sign which mark the spot of the fire extinguisher, or show the direction to it, should be visible and easy to interpret and understand. Not just for the owners of the house, but as universally as possible. In hotels, fire extinguisher signs are rarely in writing (or written in the local language).

Some scientists argue that icons have an advantage over text in terms of recognition and recall. Visual perception occupies by far the largest area of the human brain, approximately 80 percent (followed by hearing at 10 percent), making sight the most influential of the senses [5].

The human ability to recall images is superior to that for text [6]. Also, humans find it harder to distinguish one word from another, compared to distinguishing one image from another. Therefore, using icons or pictograms instead of text can help humans make decisions more efficiently. In the case of locating the fire extinguisher, it is arguably less difficult to locate a familiar, red sign with a fire extinguisher pictogram, than a lettered sign which reads (fire extinguisher).

A pictogram is a visual sign:

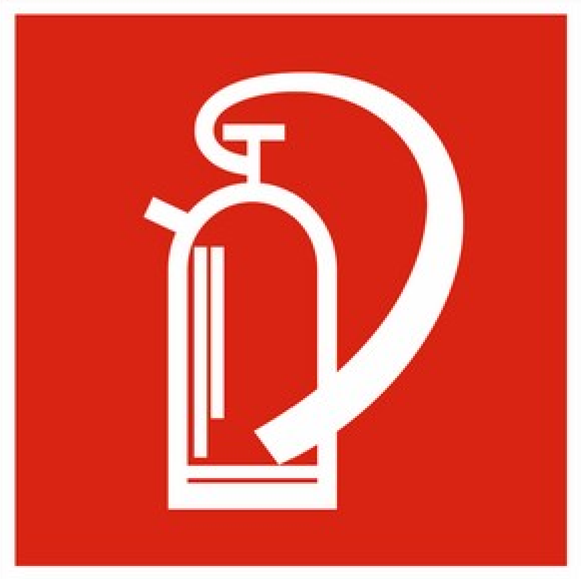


Figure 2.1: A fire extinguisher pictogram of unknown origin)

When put into context, the icon/pictogram may communicate significance:

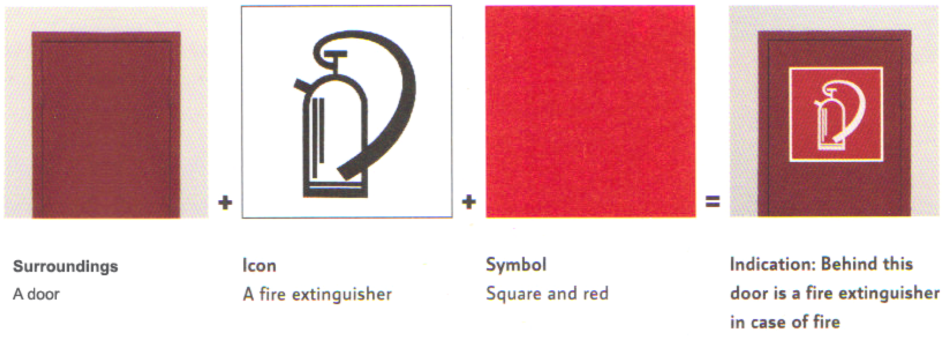

+
+
=
Surroundings
A door
Icon
A fire extinguisher
Symbol
Square and red
Indication: Behind this door is a fire extinguisher in case of fire

Figure 2.2: An icon/pictogram put into context
(Source:[7])

The context around a pictogram may strengthen its usability significantly. For instance, a fire extinguisher pictogram placed outdoor, under a tree would likely cause confusion. On the other hand, a fire extinguisher pictogram placed on a red door in a hallway would make more sense. Without a context or surrounding that defines its parameters, a pictogram would not be able to communicate its meaning clearly [7]. If the surroundings change, so will the meaning of the pictogram.

Adopted within a system monitoring visualization tool, the tool itself would likely define the context of the pictogram, as the usability of the tool will be self explanatory for most users. However, a universal design goal would be to construct a display that is less dependent on context - which radiates intuitivity in form of usability and utility. For example, if the system monitoring display is presented to a random user without prior knowledge of the tools usage.

A potential problem with system monitoring icon or pictograms is the visual appearance of the components that are relevant in such visualization. Components inside a server such as memory, CPU and disk are also made up of several components. For example, in the case with memory, some may associate memory with a single memory chip, others may think of the green memory card with several memory chips attached - which fits the computer motherboard (as in Figure 2.3).

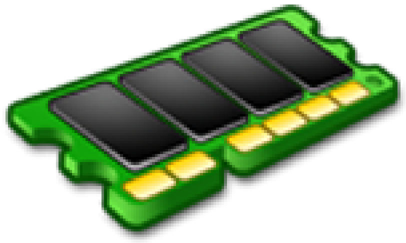


Figure 2.3: A memory card icon
(Source:Iconfinder.com)

In the case with the fire extinguisher, a general convention for how to make a fire extinguisher pictogram exists (although with minor variations), probably because the mental schema of a fire extinguisher in a population is quite similar. This may not be the case with memory. A Google image query reveals no tendency towards any specific approach to memory representation in icons or pictograms.

Another semiotic obstacle is the coloring of components relevant for system monitoring. From experience, most system administrators will likely recognize that some components traditionally have certain colors. Dark green or green is for example a preferred color among producers of different types of computers cards. This may, however, not be the case in every setting where system administrators operate.

A interesting experiment would perhaps to apply icons to a system monitoring display to see if it can increase design performance on a mobile platform.

2.1.2 Imagery

Comics are arguably something that most people are familiar with. The communication in comics applies methods and conventions that it well known, although comic-specific. For instance speech-bubbles, visual exaggerations and descriptive imagery of specific events or states

In comics, it is fairly common to emphasize the impression of the sleep state as the letter Z, often in succession and in capitalized letters. The letters are often placed slightly skewed relative to each other, maybe to reflect the somewhat sharp sound of snoring visually (Figure 2.4). Quite often the Z letters are placed floating above the sleeping cartoon character, not using speech balloons or though balloons which normally represents speech and thought [8].

This may be a way for the comic artist to distinguish sleep from though and speech, since the sound of snoring and sleep is two separate things; snoring could for instance be mimicked in an awake state. Since a spoken sentence and a single though is not an ongoing activity the way sleep is, this way of separating activities from events seem to make sense.

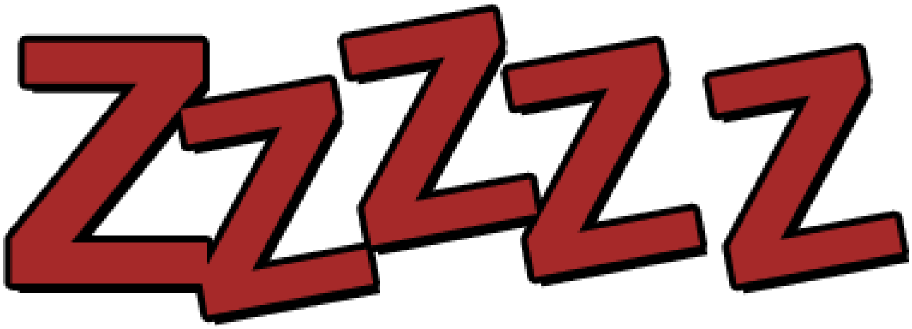


Figure 2.4: Multiple Z's, often used to represent sleep in comics

The letters in (Figure 2.4) is likely known to anyone who is familiar with the imagery and communication between artist and reader found in comics. Such letters are likely to provide the reader with information about the state of the cartoon character they are seen in connection with. The exaggeration-like focus normally found in comics allows this to be communicated to the reader clearly [8].

If the purpose of the visualization was system surveillance and the data in the visualization contained a specific units activity, a row of Z's in could therefore presumably enhance the notion of little or no activity, assuming that the user sees or knows the connection between the notion of sleep and low activity. In a surveillance context where activities were to be monitored, multiple Z's could perhaps represent a state of such activity.

When a certain material that is combustible and/or flammable reaches a certain temperature, it will eventually catch fire, provided the conditions for fire is present. Fire is a state where the material oxidize, releasing light and heat. Flames are the visible light that a fire creates. As the fire burns, the material is gradually consumed and eventually lost. Most people are likely to know the concept and significance of fire and flames, because humans, as with most other mammals, are instinctively adept at responding to fire[9]. It is associated with danger and raises adrenaline levels, causing higher awareness and pulse. Without immediate reaction towards flames lives or values may be lost. This makes fire and flames a potentially powerful indicator to danger.

"Fire" is also quite commonly used in metaphorical terms. If something or somebody is "on fire", it is often another way of saying that somebody or something is hot (without any connection to flames or fire in a physical sense).

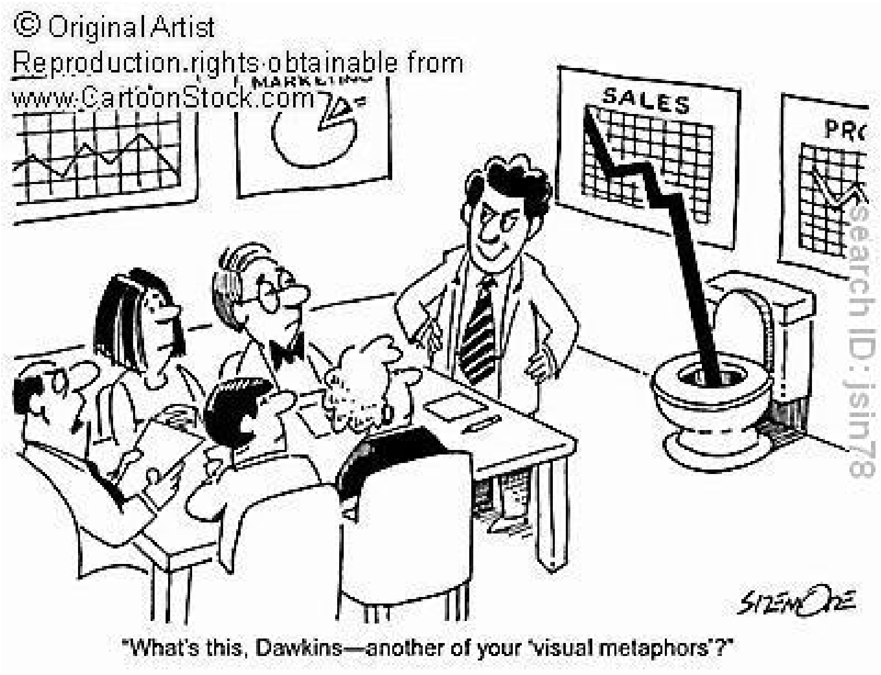
Given human knowledge about fire and flames, flames could potentially represent danger of loss of material or equipment in system surveillance visualization. If equipment used critical temperature or load, it could break down or malfunction, perhaps with similar results (such as loss of values) than after a fire. Therefore, fire imagery should be used with caution and only when levels reach critical states. A too frequent use of fire imagery in system surveillance visualization may contribute to loss of credibility and importance [6].

Figure 2.5: Flames, often associated with danger and destruction
(Source: istockphoto.com)

2.1.3 Metaphors

From language it is common to describe things in metaphorical terms. Something negative, or numbers which has a decreasing trend, might be described as something going down - exceeding or moving towards the lower boundary of an area. Examples on this lower boundary could be a drain, toilet, floor or ground. Something positive, or something that has an increasing trend, might be described as something going up - exceeding or moving towards the upper boundary of an area. This upper boundary could for instance be the ceiling, roof, or sky.

Time charts sometimes appear in comics and some of these seem to play on metaphors when drawing charts. In figure 2.6, a line chart representing sales seem to exceed the boundary of the data area and continue down into a toilet bowl (an interesting observation with the sample comic in 2.6 is that the comic is about visual metaphors)



"What's this, Dawkins—another of your 'visual metaphors'?"

Figure 2.6: A chart with sales numbers going down the toilet (Source: cartoonstock.com)

Figure 2.9 is a time series graph from 1786, drawn by William Playfair. According to Tufte [10] each part of a graphic generates visual expectations about its other parts of the graphic, and these expectations will determine what the eye sees. The exceeding of the grid at the top would perhaps break this expectation and thus add a certain effect to its perception. A possible assumption could be that the point where the time line crosses the boundary of the data area becomes the breaking point of the rectangle that makes up the graph, thus generating extra attention towards that area [11]. The continuation if the data series starts over at the bottom at the graph. Tufte calls this a torus-graphic [3]. This effect could perhaps be avoided by changing the scale of the y axis to accommodate the max values of the visualized data.

2.1.4 First impressions

Humans tend to look for patterns or assign meaning to objects by identification or recognition. This mechanism is called pattern recognition and is an important area within the field of cognitive psychlogy [11]. Pattern recognition depends on information stored in long-term memory. For instance, a shape that resembles an animal with a particular appearance would make no sense to the observer if the observer had no prior knowledge of that specific animal. As a result of the matching process, a decision is made by the observer on the most likely identity of the shape in question.

Figure 2.7is showing a comic which contains a chart with a curved time line, hanging on the wall. Although the curved line is merely black ink, human visual perception - due to the mechanic of pattern recognition - apply meaning to the shape in order to identity and categorize it. A likely assumption would be that this is what the artist make use of in the comic.

If the graph in figure 2.7 for example was displaying costs or expenses, the time line would be showing a positive trend, despite the negative effect of a sour mouth after pattern recognition. Only after having interpreted the graph more thoroughly would the positive.

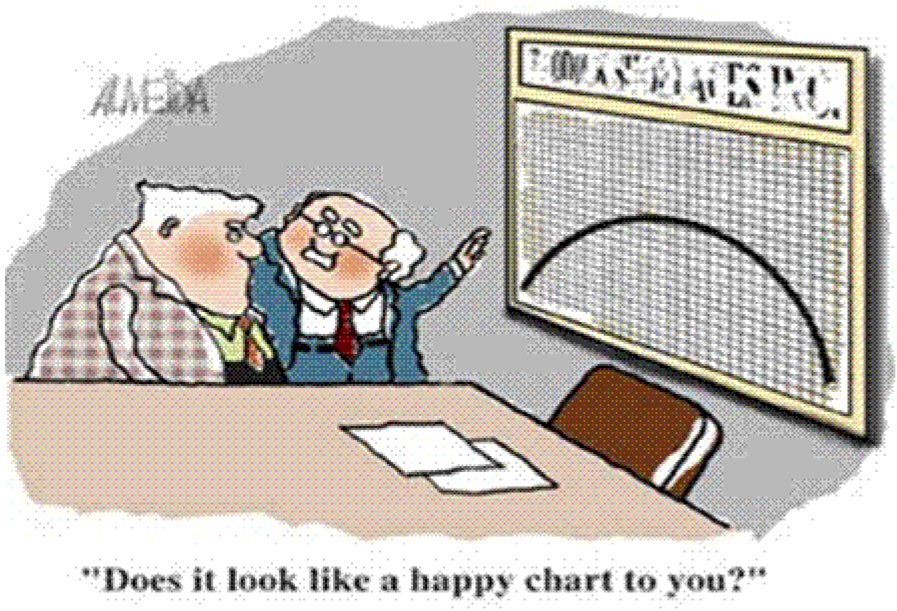


Figure 2.7: An unhappy chart
(Source: cartoonstock.com)

2.1.5 Graph specific

In the introduction of the background chapter, Holmes and Tufte were mentioned as their opinions on information design differs significantly. (Figure 2.8) illustrates the difference in a line chart design between a Holmes chart and a minimalistic Tufte-style approach generated by Microsoft Excel)).

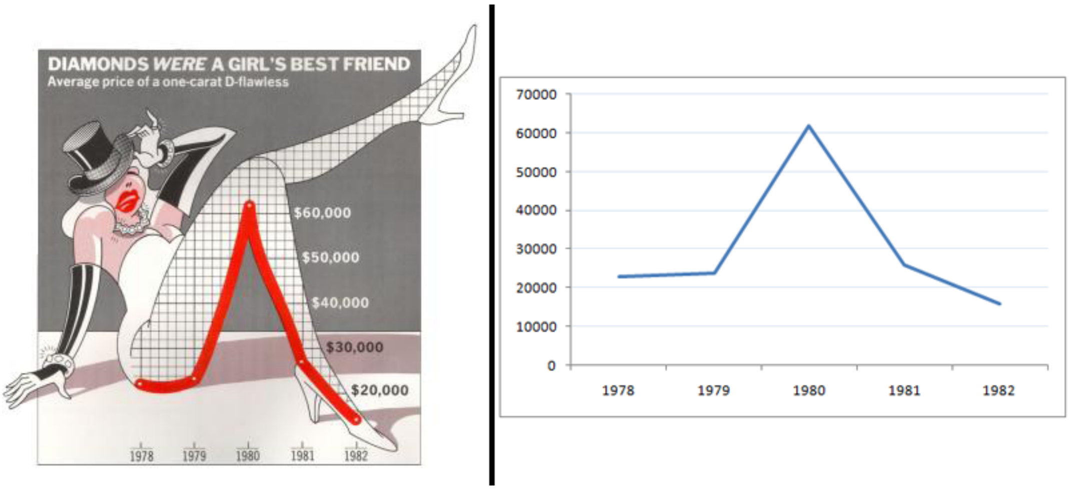


Figure 2.8: Left: chart design by Holmes. Right: Tufte-typical design (Sources: Nigel Holmes: Designers Guide to Creating Charts and Diagrams (1991) and Edward R. Tufte: The Visual Display of Quantitative Information (2001))

Explanation of Figure 2.8: Sample line chart design by Nigel Holmes. Right chart: Minimalistic line chart design without distortion of data as recommended by Edward R. Tufte.

As seen in Figure 2.8, Holmes often adds context dependant embellishment to his graphics in order to communicate on several channels [12]. For instance, the stalking she wears also functions as chart gridlines, perhaps to enchant the presence of graph functionality. While Tufte claims that all embellishment and decoration that interferes with the area in which the data is presented is merely distorting the data [10].

Although long-term recall and seeing things immediately (and putting them in context) is quite different, the methods used for creating embellishment in Holmes-style graphs to help long-term memory resembles methods used in interaction design to enhance perception. For instance, the context used in conjunction with a pictogram (for example a fire extinguisher pictogram on a red door in a hallway) contributes significantly to how the pictogram is perceived [7].

Also, important principles of interaction design such as usability, visibility,

conventions and familiarity, used by for instance Holmes, could arguably contribute to increased predictability and speed up perception [6].
Gridlines are used in graphs to support components, providing support in the visual perception of data values. Originally, one of the main purposes of the gridlines was to assist the drawing of the values inside the data region[13].
Bateman et. al [14] tested interpretation accuracy and long-term recall for both plain (Tufte-style) and embellished (Holmes-style) charts. They found that the test users interpreted the two chart types in a similar fashion, but that Holmes-style charts scored significantly higher on long-term recall.

A data visualization needs an specific area, an empty canvas where numbers in a data set can unfold. This small, confined space is the key to every successful communication of data in data visualization. McCandless claims that information visualization is a form of knowledge compression, a way of squeezing an enormous amount of information and understanding into a small space [15]. Also, Tufte's principles of graphical excellence [10] states that this space should be as small as possible, but never beyond the point where communication suffers.

The properties of the data field where the data is presented, is - arguably - one of the most important factors that affects the way users interpret the data [10]. The relationship between x and y axis length - the aspect ratio - is an often overlooked factor when designing the shape of the graph plot area. Most graph tools used in system monitoring use scalable axis, where the shape of the graph data field is determined by the data set. Since a longer X axis would make the timeline graph height shrink, proportion, an elongated X axis shown on a mobile screen would arguably make the graph harder to read.

It is calculated by dividing height by width. Consequently, a number below one suggests a graph where the width is greater that the height and one would suggests a square. The lower the number, the wider the image. The aspect ratio is commonly displayed as two numbers separated by colon, for instance 3:2 (1:1,5) or 16:9 (1:1,77) - the latter being the standardized ratio for a widescreen television screen.

The golden ratio frequently appears in nature, for instance in plants and throughout the animal kingdom. It is also appears in geometry, for example in the Fibonacci spiral and in pentagrams [16], and in buildings such as The Notre Dame cathedral facade and in Parthenon in Athens. The golden ratio is 1:1.618. The proportion of the golden ratio is supposed to have a pleasing visual effect [6].

The arguable inventor of the line graph, William Playfair (1759-1823), the first in the world to produce line graphs and charts in published writings [17], used wider diagrams in 92 percent of all cases [10], and is often used by Tufte as a template for designing plot graphics (An example can be seen in Figure 2.9). When estimating estimating the relationship between both axes in all of Playfair's diagrams, Tufte found that two thirds had proportions between 1:1.4 and 1:1.8. Furthermore, a graph plot of each of Playfair's proportions used in graphs reveals that two out of three major ratio concentrations corresponded almost exact with the golden rectangle ratio.

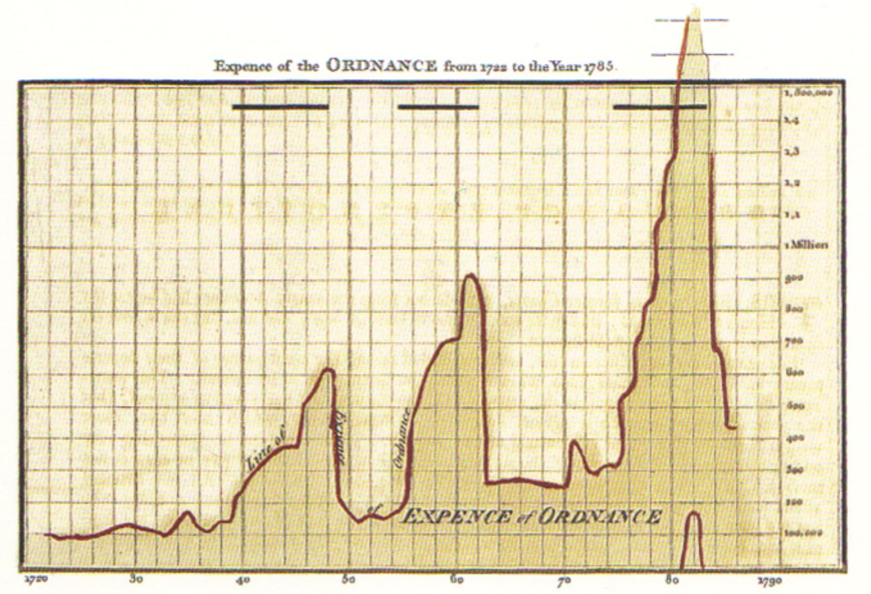


Figure 2.9: Playfair: Commercial and political atlas (1786)
(Source: Edward R. Tufte: Envisioning Information (1990))

Scientists have conducted visual preference tests on rectangular proportions since 1860, without any decisive findings. Helstrup and Kaufman [5] stresses that cognitive conditions often corresponds with biological, medical, sociological, juridical, philological and philosophical conditions. Tufte agrees (IBID), admitting that experiments in perception are highly context dependant. While Tufte is takes care not to proclaim the golden ratio as the secret formula to heighten perception, he admits the presence of a preference in its proximity in these findings.

The aspect ratio can greatly affect the appearance of the rate of change within a graph. Changes in a graph with a lower aspect ratio may give a dramatic impression of the rate of change [13]. Identical data displayed with aspect ratios of 0.5 and 2.0 are both accurate, but the perceptual impact of the two graphs may differ significantly. This effect may very well be used to manipulate readers. An exmaple can be found in Figure 2.10.

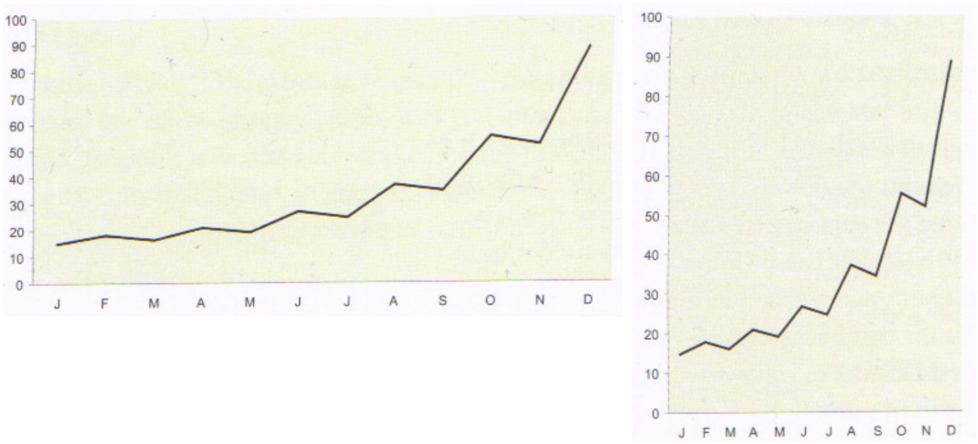


Figure 2.10: Identical data sets displayed in different aspect ratios
(Source: Stephen Few: Show Me the Numbers (2004))

According to Few, a graph's data region consists of two main visual attributes; the aspect ratio and the fill color [13]. Two approaches to highlighting the data region is by either filling the data region with a subtle color, or by coloring the surrounding areas of the graph with a subtle color difference. The same principles of coloring would apply.

Few further states that data should stand out above the other components of the graph, so that the readers eyes are drawn to the data [13]. He further suggests that in order to so, the data objects should me made visually prominent and not overshadowed by a vibrant and colorful background.

Tufte is an advocate for telling the truth about data, meaning that a modest amount of data should be visualized accordingly [3]. Although visualizing or enhancing NULL values or is discouraged from a interaction design point of view (as it gives little or no meaning to represent nothing visually in information design), very little is said about visualizing the state of such data. In Peep, the audio-based network monitoring tool , a modest state with very little traffic were still audible, but rather presented as tranquil background noise contributing to a calm atmosphere[18]

A great risk of visualizing zero values is drawing unintended attention when this may not be of necessity [11]. However, by using familiar graphic elements, the concept of nothing could potentially be communicated more efficiently. For example, if a user have to spend time looking at a the data on time line graph in order to determine low activity, it is reasonable to assume that there may be room for improvements in the visual expression.

Context dependant embellishment seem to improve long term memory [14]. Cognitive psychology suggests that the mechanisms inside the human brain which is used in visual perception are closely connected to long term memory, and both seem to improve by the use of imagery [5][11].

2.2 Other approaches to real time monitoring

Real time data visualization has arguably not been around very long. Different arenas have adapted and developed their own methods, many adjusted and adapted for a special usage. While some methods have arguable become conventions, others seem more experimental. Because real time attributes are hard to visualize, some well known real time monitoring tools may provide some answers.

A brief look at four different approaches to real time visualization may provide examples on how events, states and transitions between states are handled. The description will focus on functionality, interaction design and potential problems with the attributes of real time data. How these techniques potentially could contribute to a real time system monitoring visualization will also be mentioned.

2.2.1 SETI@home

The Search for Extra-Terrestrial Intelligence (at home) is a project initiated by University of California, Berkeley - to provide extra computational power in the search for extra-terrestrial life[19]. SETI@home does this by distributing raw signals from large antenna arrays to volunteers who install the SETI-software on home computers. When installed, the software runs in the background, sending data back and forth as blocks of data is processed.

A visualization option (Figure 2.11) is included in the software, which visualizes the computation in real-time. The technique is aimed at visualizing a process (looping through a signal time-frame), and it displays three dimensions from the data set: processing time, signal frequency and signal power. The transition between data states is visualized by "new" data being pushed into a three-dimensional column chart, through the processing time axis - in a repeating pattern. Data along the frequency axis is visually separated by the coherent colors of the rainbow. The time and frequency axis has a very high data resolution, making it virtually impossible to separate single columns. The chart itself rotates while all the columns go up and down as the signal is processed.

The complexity of the SETI data-set is not fully visualized [20]. The area of the sky from which data is collected, the antenna used for collecting the data, as well as recording time - are all data that is represented by text only. The purpose of SETI@home is finding abnormalities within the signal, but the current visualization method would make this challenging indeed. Based on the assumption that a successful visualization technique can be measured in the rate information is communicated, one may assume that the processing visualization is for decoration purposes only.

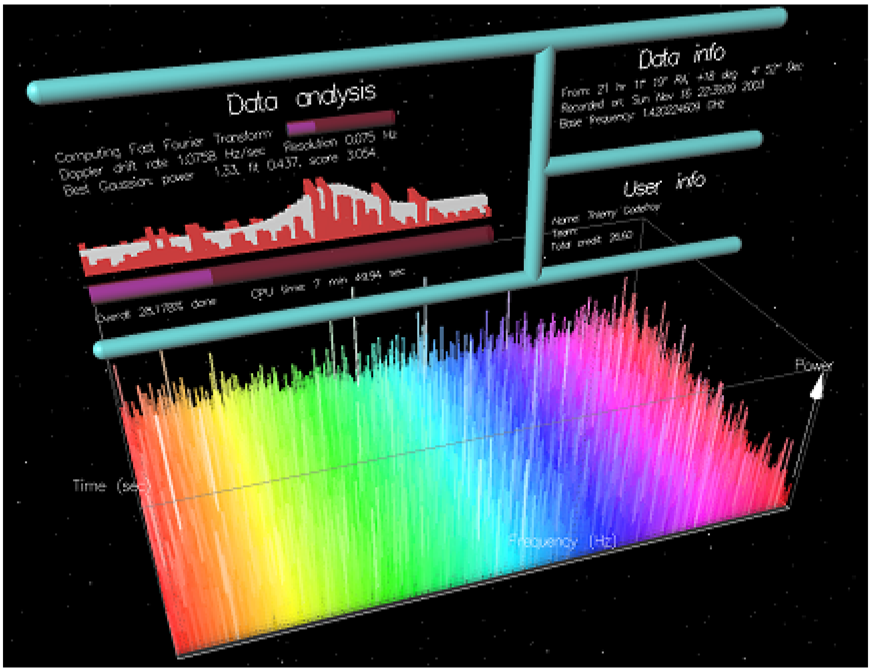

Data analysis
Computing Fast Fourier Transform
Doppler drift rate 1.0758 Hz/sec
Best Gaussian: power 1.33, fit 0.437, score 3.054
Resolution 0.075 Hz
Overall 28.1788% done
CPU time: 7 min 49.94 sec
Data info
From: 21 hr 11' 19" RA, +18 deg 4' 52" Dec
Recorded on: Sun Nov 16 22:38:09 2003
Base Frequency: 1.420224609 GHz
User info
Name: Thierry Godefroy
Team:
Total credit: 28.60
Power
Time (sec)
Frequency (Hz)

Figure 2.11: SETI@home computation visualization
(Source: The SETI@Home Project)

2.2.2 Resource monitoring speedometers

The speedometer visualization technique for computer monitoring (Figure 2.12) is widely used [21]. This real-time monitoring method is a metaphor to keeping track of vehicle speed - which provides hints of usage [6].The speedometer shows data state, which is one-dimensional and low in complexity [22]. The transition between data states is not visualized clearly; only the angle of the needle and its point along the speedometer disc. Transition is hardly detectible (especially minor changes) unless you have a digital number option connected with the speedometer somehow.

An often-seen visual attribute on speedometers, is a red area on the disc that marks critical data values. There is no further critical value indication unless specially adapted. If so, it would differ from its real world metaphor template. There are some major differences in perception between metaphors and real world applications. If you fail at keeping a watchful eye on your PC monitoring speedometer, the consequences will probably be insignificant - as the PC mainly manages itself. On the other hand, if you fail at monitoring your car speedometer, the consequences may be severe. The knowledge of the difference of importance in consequences may lead to different usage [11].

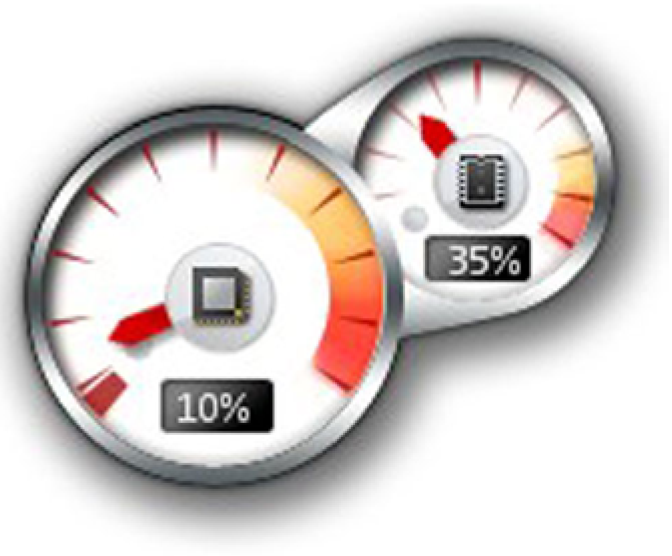


Figure 2.12: Speedometer hardware usage visualization
(Source: CPU Meter Gadget Free)

2.2.3 Live sports monitoring

Live data from sports events through the World Wide Web is becoming more and more popular (Figure 2.13). TV-channels and popular sport web sites compete in having the best sport coverage available. A common example is a sport game with two competing teams. Data complexity is varied, and may depend on a number of factors, such as the type of sport in question. The timeline is common for all live sport data and the event usually takes place within a limited timeframe, which may influence perceptual factors. Also, some scoreboards aim to mimic the real event as closely and precise as possible, which also may influence perception and complexity. A complex communication interface may require a high degree of background knowledge to understand, or else perception will fail. Complexity is also related to how much data the provider has chosen to visualize and how it is presented.

Personal interest may play an important part in choosing the most appropriate live sport coverage application. Some users prefer detailed information (quality), while others may like simplicity and overview (quantity). The level of detail may even be adjustable. State is represented either discretely or as the current state of the game - often equivalent to the current score. Activities may be happenings, not necessarily related to the state or events. Changes between states or events may be presented in such a way that it captures the user's attention.

What makes real time sports scoreboards interesting from an information communication perspective, is that the display must communicate information quite effectively in order to capture a loyal user base, preferably more effective than the competing web sites. Popularity is measurable. If this assumption carries any truth, popular real time scoreboard providers possess significant insight in interaction design - an amount somewhat in proportion with the number of users. As a consequence, one might even dare to suggest that providers of live scoreboards are contributing to pioneering the field of real time information visualization. Evolutionary development - in this setting - would be a natural selection where the weaker performing information visualization fails to survive. A way of ensuring high performance and quality [23]

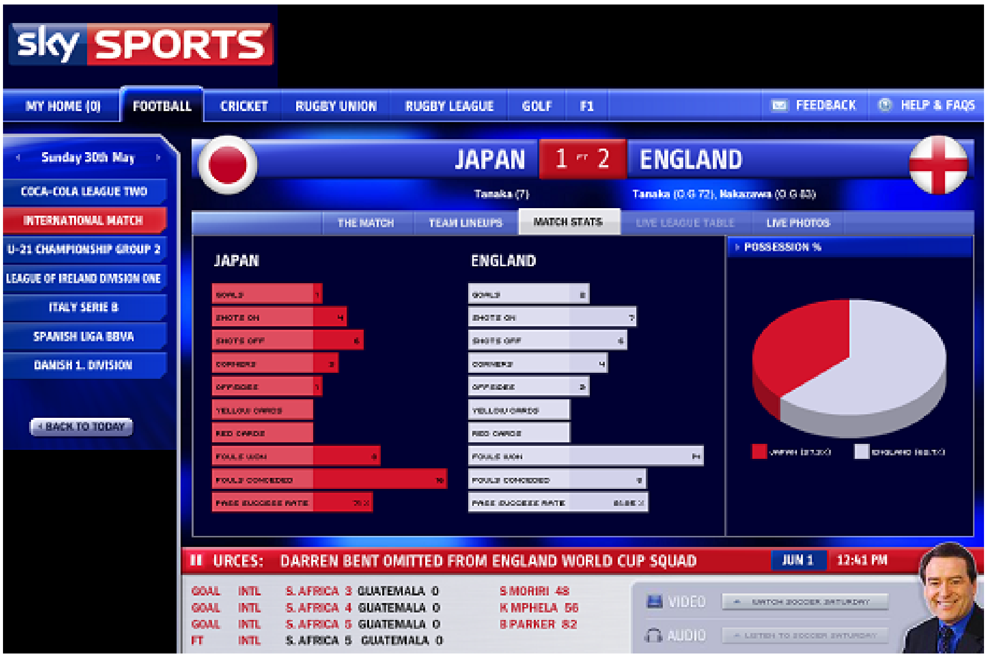
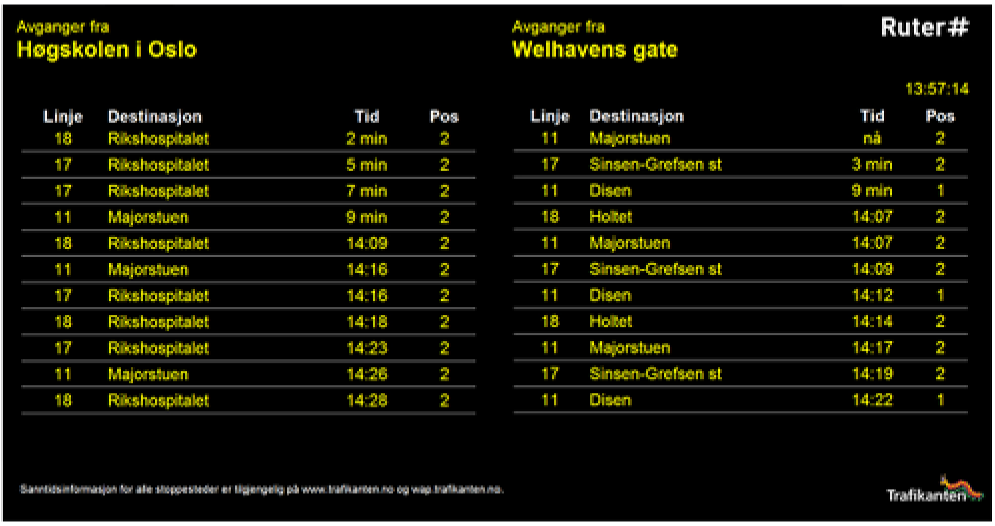


Figure 2.13: Real-time sports monitoring
(Source: Sky Sports)

2.2.4 Public transportation departure screens

Real time information departure screens (Figure 2.14) are one of the most important sources of departure information for public transportation around the proximity of stops/platforms. Such screens are constructed for intuitivism, as they are purposely designed to satisfy the needs of the largest possible group in a population. As a consequence, real time departure screens strictly follow known conventions for such information visualization that seem to be internationally renowned.



Avganger fra
Høgskolen i Oslo

Linje	Destinasjon	Tid	Pos
18	Rikshospitalet	2 min	2
17	Rikshospitalet	5 min	2
17	Rikshospitalet	7 min	2
11	Majorstuen	9 min	2
18	Rikshospitalet	14:09	2
11	Majorstuen	14:16	2
17	Rikshospitalet	14:16	2
18	Rikshospitalet	14:18	2
17	Rikshospitalet	14:23	2
11	Majorstuen	14:26	2
18	Rikshospitalet	14:28	2

Avganger fra
Welhavens gate

Ruter#

13:57:14

Linje	Destinasjon	Tid	Pos
11	Majorstuen	nå	2
17	Sinsen-Grefsen st	3 min	2
11	Disen	9 min	1
18	Holtet	14:07	2
11	Majorstuen	14:07	2
17	Sinsen-Grefsen st	14:09	2
11	Disen	14:12	1
18	Holtet	14:14	2
11	Majorstuen	14:17	2
17	Sinsen-Grefsen st	14:19	2
11	Disen	14:22	1

Sanntidsinformasjon for alle stoppesteder er tilgjengelig på www.trafikanten.no og wap.trafikanten.no.

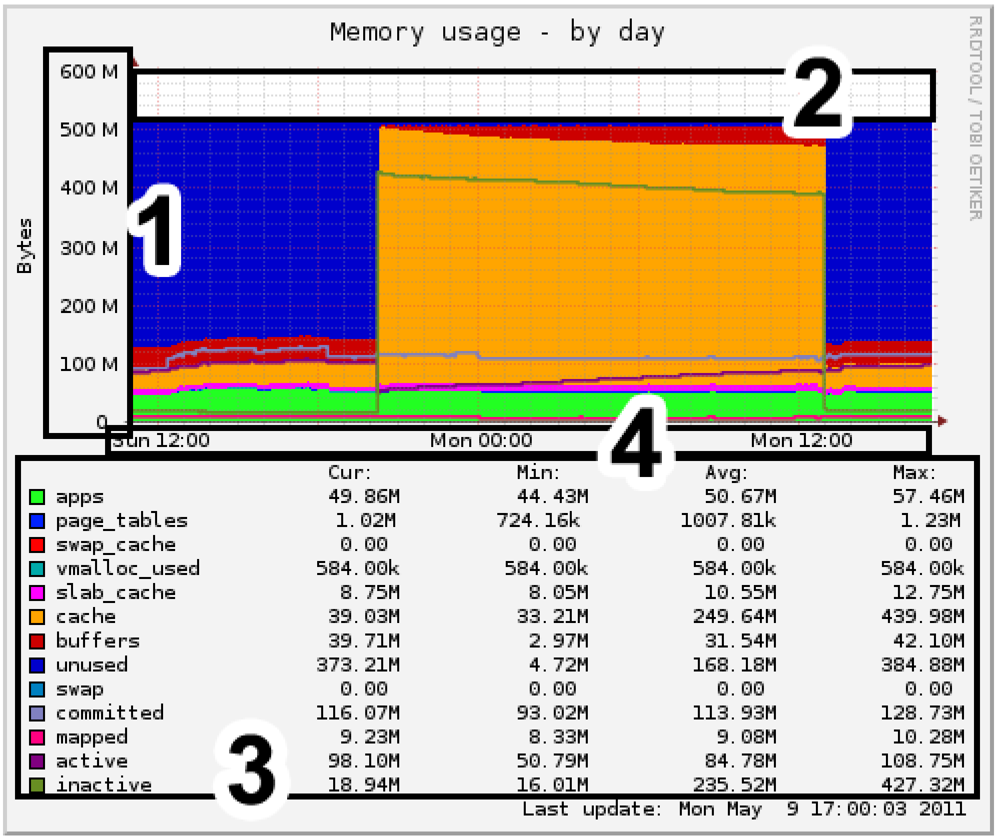
Trafikanten

Figure 2.14: Real-time monitoring of departures
(Source: Trafikanten AS)

On a standard public transport departure screen, two elements constantly change. First, the departures are always listed in a chronological order with the next departure in real time on top, meaning that the order of upcoming departures move upwards as vehicles departs. Whenever this occurs, rows in the table swap position. Secondly, the waiting time, which is always represented by digits, change as the departure time approaches. If a platform notation is present, this is statically attached to the same row as the departure in question. There is no visualization of the transformation between states - or events. Also, there are no visible indications of the presence of real time data other than mental schematics and departure screen conventions [24] [6] [11].

2.3 Other approaches to system monitoring

Munin is a well-known open source system monitoring tool which uses the graph program RRDTool for visualizing data. It widespread deployment and similar usage domain makes it a suitable system monitoring tool for analyzing and to use as basis for comparison.A critcal analysis will be conducted in order to identify potential improvements in visualization



	Cur:	Min:	Avg:	Max:
🟩 apps	49.86M	44.43M	50.67M	57.46M
🟦 page_tables	1.02M	724.16k	1007.81k	1.23M
🟥 swap_cache	0.00	0.00	0.00	0.00
🟦 vmalloc_used	584.00k	584.00k	584.00k	584.00k
🟪 slab_cache	8.75M	8.05M	10.55M	12.75M
🟧 cache	39.03M	33.21M	249.64M	439.98M
🟥 buffers	39.71M	2.97M	31.54M	42.10M
🟦 unused	373.21M	4.72M	168.18M	384.88M
🟦 swap	0.00	0.00	0.00	0.00
🟪 committed	116.07M	93.02M	113.93M	128.73M
🟥 mapped	9.23M	8.33M	9.08M	10.28M
🟪 active	98.10M	50.79M	84.78M	108.75M
🟩 inactive	18.94M	16.01M	235.52M	427.32M

Last update: Mon May 9 17:00:03 2011

Figure 2.15: Memory usage: Munin default installation explained
(Source: Munin Open Project / RRDTool)

The detail level in 2.15 will arguably not optimally fit a mobile screen as detail level is noticeably high.On mobile screens of lesser (than desktop) size, it is important to utilize the space well as there is not much of it. This is usually not as big a problem on most desktop monitors, as larger screens have the capacity to display more information.

This time line graph displays a memory device with max capacity of 512Mb. Yet, the y axis ends on 600 (Figure 2.15: (1)). This leaves the area between 512 and 600 unused; an area which makes up 14,67 percent of the total data area (Figure 2.15: (2)). This suggest that the default setup of the conventional monitoring visualization does not seem to utilize the available space in the data area adequately.

The x axis of the graph (the time interval in a time line graph) spans for thirty hours (Figure 2.15: (4). The graph title reads (by day) which implies a twenty-four hour period. By counting the vertical gridlines and vertical tick marks, six additional hours is revealed - in addition to the twenty-four . There are no other apparent indicator to the true number of hours visualized. For example, at Monday 17:00 (as in figure 2.15), the time interval will span backwards to

Sunday at 11:00.

Every 12th hour there is a major tick mark with an axis label displaying the three first letters in the name of the day, and hour plus minutes.

Figure 2.15: (3) shows the visualization of 13 data sources sharing the same data area. At least six (visible) combined area lines constitute a one hundred percent area which is constantly positioned at 512Mb on the y axis. At least three single-lined time lines are visible above of the area lines. The remaining data set types (according to the legends below the data area, there are thirteen data sets in total) are challenging to identify as either area or line, as they seem to be partially concealed.

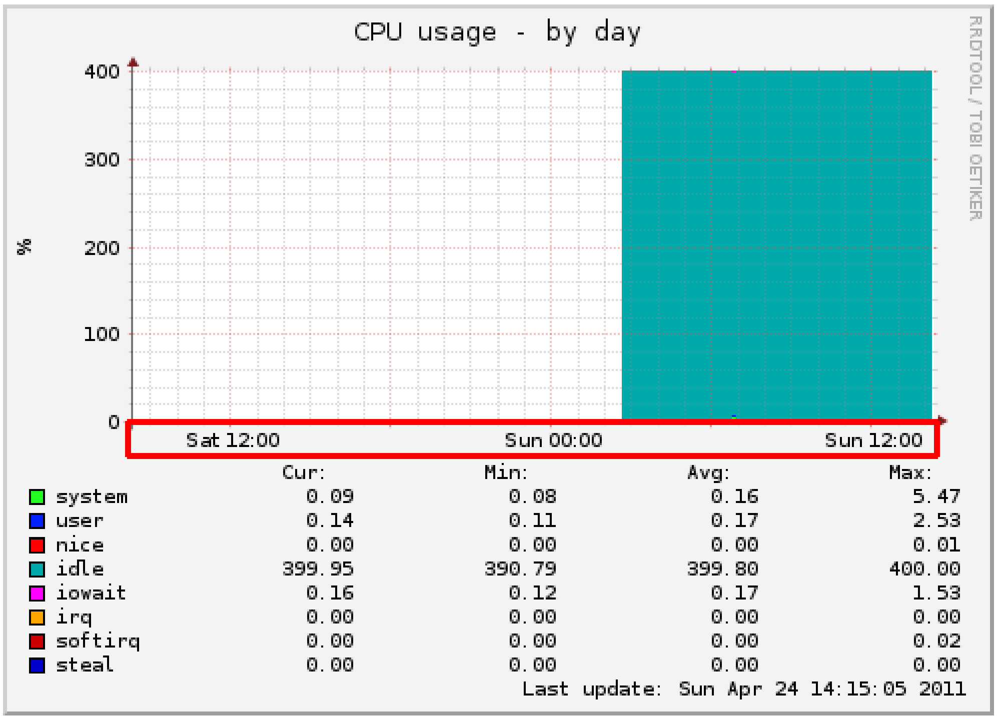
The legends in Munin possess more than one function. In addition to matching colors to data sets, they display current value and statistical properties of the data sets in numbers. Communication on several channels is a well known method of streamlining information display according to Heim [6].

The time interval should perhaps be communicated more accurately to the user in order to avoid possible misperceptions. For example, if an event that influenced the time line occurred more than twenty-four hours ago, this particular event should perhaps no longer be visible on a daily display more than twenty-four hours later. If the event is shown, however, the graph should perhaps have another title to better match the time interval. This possible misconception could be tested in a comparison between the default conventional display and a new display where this is modified.

Assuming that the visualization display communicates clearly what kind of dimension the x axis renders (time), text support for the main tick marks on the x axis should arguably not contain redundant information - as this should be avoided on small screen displays. For example, most major tick marks along a time axis of a certain size would likely occur between hours, when the minutes display is zero. These minute digits are often displayed nevertheless, perhaps to enhance the impression of time. One may assume that tick mark text amplifies the notion of a time axis if this text contains letters, numbers and characters that users may associate with time, such as a semicolon between two pair of digits (hours and minutes), possibly supported by letters indicating the day of week ahead. This method of rendering time is a convention which makes users perceive notations of time more efficiently [6].

In system monitoring visualization, the most frequent used method to visualize data is time line graphs. This makes sense as data sampled at a certain time are (at minimum) two dimensional; the data will contain both value and data sample time. This allows the user to look at values at certain times and thus detect be able to detect patterns in the data. Some monitoring tools (for example tools utilizing the dashboard design approach [25]) use bar charts and pie charts, but since neither of these takes time into consideration, it reduces the user's ability to make comparisons based on changes in value.

The figure below (figure 2.16), is a Munin installation visualized through RRDTool showing CPU activity during a 24 hour period (Milestone labels along the time line are highlighted red):

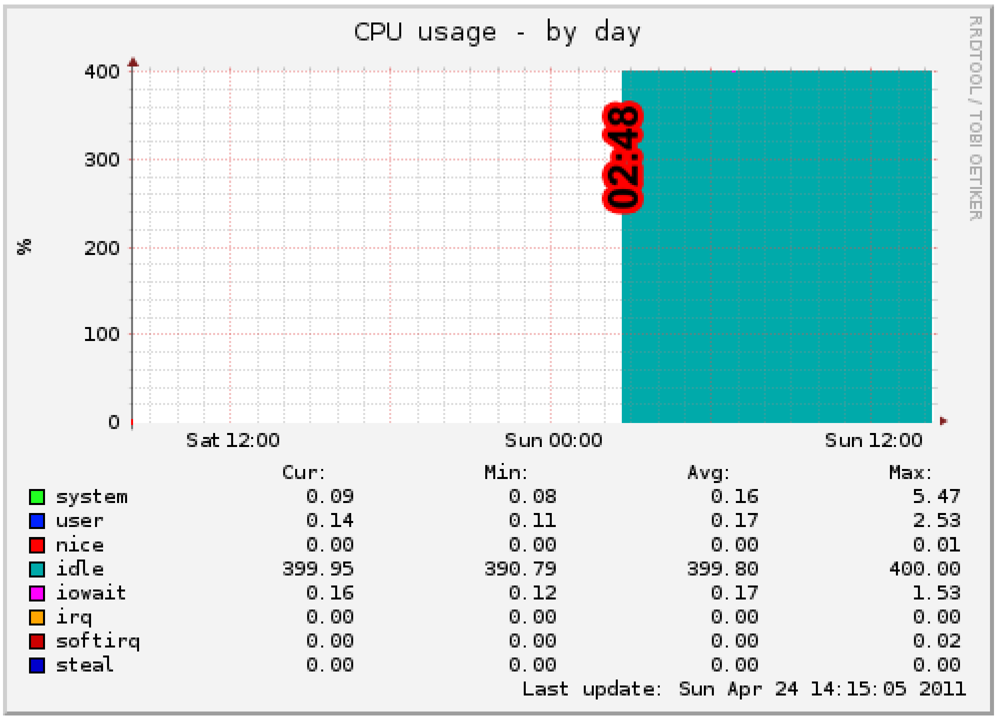


	Cur:	Min:	Avg:	Max:
🟩 system	0.09	0.08	0.16	5.47
🟦 user	0.14	0.11	0.17	2.53
🟥 nice	0.00	0.00	0.00	0.01
🟦 idle	399.95	390.79	399.80	400.00
🟪 iowait	0.16	0.12	0.17	1.53
🟧 irq	0.00	0.00	0.00	0.00
🟥 softirq	0.00	0.00	0.00	0.02
🟦 steal	0.00	0.00	0.00	0.00

Last update: Sun Apr 24 14:15:05 2011

Figure 2.16: CPU usage: Munin default installation explained
(Source: Munin Open Project / RRDTool)

Figure 2.16 illustrates the problem of pinpointing event times in traditional time line visualizations with standard axis . The exact time of the sudden increase in CPU activity in the example in 2.16 seem hard to locate as there is no apparent marker available. By closer inspection, there is a highlighted gridline every sixth hour. One approach to finding the approximate time of the event is to count the thin, black lines in between the red six hour markers. By doing this, the user may find that an estimate for the CPU increase is between 02:00 and 03:00. A smaller time frame and higher granularity in the sample rate would not remove the problem of pinpointing an exact time, as the visual measurements are the same. It is reasonable to assume that there is room for improvements in regards to enchant timestamped events such as sudden significant changes. Such as for instance in the manipulated Figure 2.17:



	Cur:	Min:	Avg:	Max:
🟩 system	0.09	0.08	0.16	5.47
🟦 user	0.14	0.11	0.17	2.53
🟥 nice	0.00	0.00	0.00	0.01
🟦 idle	399.95	390.79	399.80	400.00
🟪 iowait	0.16	0.12	0.17	1.53
🟧 irq	0.00	0.00	0.00	0.00
🟥 softirq	0.00	0.00	0.00	0.02
🟦 steal	0.00	0.00	0.00	0.00

Last update: Sun Apr 24 14:15:05 2011

Figure 2.17: CPU usage: Manipulated Munin default installation (Source: Munin Open Project / RRDTool)

2.4 Real time data visualization challenges

An apparent difference between static and real time data is that the real time data values change. This property may or may not be visible for the user (for instance depending on the refresh rate of the data or in the way the data is presented in order to emphasize it's real time properties, but this difference gives real time data certain additional attributes compared to static data. Static data are in a fixed position.

To emphasize dynamics or changes that are taking place on the information display, designers have traditionally added elements to information design that has to do with time such as metaphors for displaying or measuring time[26], in order to enhance the impression of ongoing activities. This has for instance often been a load bar moving from left to right, or a ticking clock with clearly visible movement indicators as for example the second hand on an analog clock - or a blinking colon between hours and minutes on a digital clock.

A similar perceptual function is present in operating systems where the appearance of the cursor changes to for instance a rotating time glass or rotating arrows, in order to provide the user with the notion that something is about to happen in near future and that a process is taking place. The way the user expects a certain display of a specific information visualization to behave is therefore also a property of the data, since this says something about how the real time data should be communicated without loss of information.

The state of the data is another important attribute. State describes the current value of the data at a given time. Real time data may change state as value changes, which mean that the data in question can exist in several states over a time period, as the concept of time is added. The change between states is an attribute which is not present in static data, as the data is constant. The change in state of real time data could either be described as an event, or as a transition between states.

Also, events could be separate incidents, occurring proportionally with the current data state at a specified time within a time frame. These may or may not influence the value of the data, as events may be either descriptive or influential. Events are never continuous, as opposed to states. Both states and events may be attributes which describes the data. States are simply descriptive and may not influence the data directly, but they may in effect trigger events or changes in state, in case of for instance states of time limited durations.

The state of data could for instance be a general description of a data value, for instance if the value is currently within a range which is considered normal. The definition of the range in question could also be the state of the data, which implies predefined ranges of values in which data could be described. An event could for example be an increase or a decrease of a data value. Transitions between states could for instance be when the state of data changes from normal to critical.

When visualizing real time data, it is important to consider is how to process events, states and the transition between states. In addition, features and relations between data are properties which are visually challenging - as they may vary in magnitude, certainty, complexity, and importance [27]. For example, an important event could lead to a dramatic increase of a data value, passing one or several thresholds. An unimportant event could be the same (or even greater) increase - but not, however, passing a certain threshold. This requires a predefined set of ranges where the data may *live*, which again suggests a threshold-defining ontology or a policy.

Burns and Hayes [28] emphasize the changes in data behavior within a real-time system model .States can be both discrete and continuous. Activities have durations, while events are always instantaneous and natural way of expressing change within a system. To achieve full communication performance, these attributes may be presented quite differently in real-time visualization techniques - as context changes [24].

In theory, data may change dimensionality (e.g. changes in axis, scaling) as well as complexity (e.g. change of data properties) between states [29]. Therefore, to fully support real time functionality, new approaches to constructing methods will be necessary as well as classification of potential changes in real time data sets.

Michael Gilfix and Alva Couch [18] have presented a solution to some of these visualization problems, albeit sonically. They created an audio-based real-time network monitoring tool called Peep, using sound to provide real time information about the network. The sounds in Peep have different features and purposes; the texture, flavor and frequency of the audio represent network performance, while the feel of the sound represent the total state. A key necessity of Peep is the ability to remain pleasant for the listener over longer periods of time. This is achieved by playing pleasant and tranquilizing natural sounds as it projects current state to the listener in real time. Humans are instinctively apt at recognizing when background sounds change. The natural sounds are split into three categories: Events are things that occur once and are represented by single peeps or chirps. States are ongoing events; continuous sounds that may alter as ongoing events alter. Finally, heartbeats represent the ongoing state, which are specific sounds in intervals. It would be of great use to identify and the visual counterparts of Peep's attributes.

The degree of real time authenticity (how *real* the real time is) in a display of data depends on several factors. Combaz et.al [30] introduces a quality manager and a scheduler for real time systems which differentiates between real time and virtual time, making slow systems running real-time applications more efficient and enhancing user experience. However, there is a significant difference between real time data simulation (virtual real time) and real time (now). If the user is aware of this, simulated real-time may be suffer from less attention it is granted less importance.[11] [24]. In communication, perception is essential and vision is the most important tool in a mesh of sensory systems [5].

Real time visualization shows data (supposedly) in movement. Movement is presumably allegeable for visualization. One of the key characteristics of vision is the perception of movement [11]. This occurs when an object changes position or form relative to its background. The figure-ground notion is a well-known gestalt principle of perception [6]. Figure-ground movement perception works in two ways: the object stands out because it moves relative to the background, or the backgrounds moves while the object stands still - producing the same effect.

Another non-gestalt movement perception phenomenon is the ocular pursuit: the ability to detect moving objects within our field of vision [11]. If a bird or a flying snowball suddenly appears, we automatically start tracking it with our eyes, if not only for a short moment. The same effect appears if someone uses a flashlight inside a dark room; we track the movement of the light. As movement is one of the key characteristics of a real time data set, it sounds reasonable that perception of movement and perception of real time information are closely connected.

2.5 Visualization framework

It is hard to predict how new information visualization methods will work due to the fact that there is no defined underlying theory to support the construction of such methods [27]. If such theories existed, one might assume that one or several formulas for creating efficient visualization methods also existed. The field of information visualization is connected to a variety of other sciences: Human-computer interaction (HCI), graphics, design, psychology, business methods and computer science - among others. Most affected scientists calls for a strengthening of the field of information visualization [31].
Attempts have been made to construct classifications of visualization techniques by examining data set types that are compatible with these techniques. A known data domain simplifies the process of choosing a method. However, such classifications do not provide assistance in applying and implementing these techniques. Two abstract models have been introduced in an attempt to model the visualization transformation process.

Chi introduces a data state reference model [32] to assist in finding methods for visualizing data - and help taxonomization of information visualization techniques [33]. The data state model describes the complete transformation process from raw data to visualization. As we can see from (Figure 1), visualization in the data state model is, according to Chi, the end product of a four stage process. Transformation steps between each stage abstracts information without changing the underlying data structure. The purpose is to isolate dependencies in the data structure to help construct new information visualizations - which is presumably achieved by dissecting the real time data.

Further, Chi applies the data state model to a number of known visualization methods [33]. The end value (final visualization) should reflect the data domain as much as possible. Some of the visualization methods in Chi's paper include complex data sets in multichannel presentations. Real time data seem to fit this model. The analytical abstraction stage of the data state model describes meta-data (data about data) which, if properly defined - presumably should cover the challenges that real time data introduces; states, events and activities. For example, a real time data event itself may be described as metadata for real time data. This may very well fit the analytical abstraction in Chi's model (Figure 2.18)

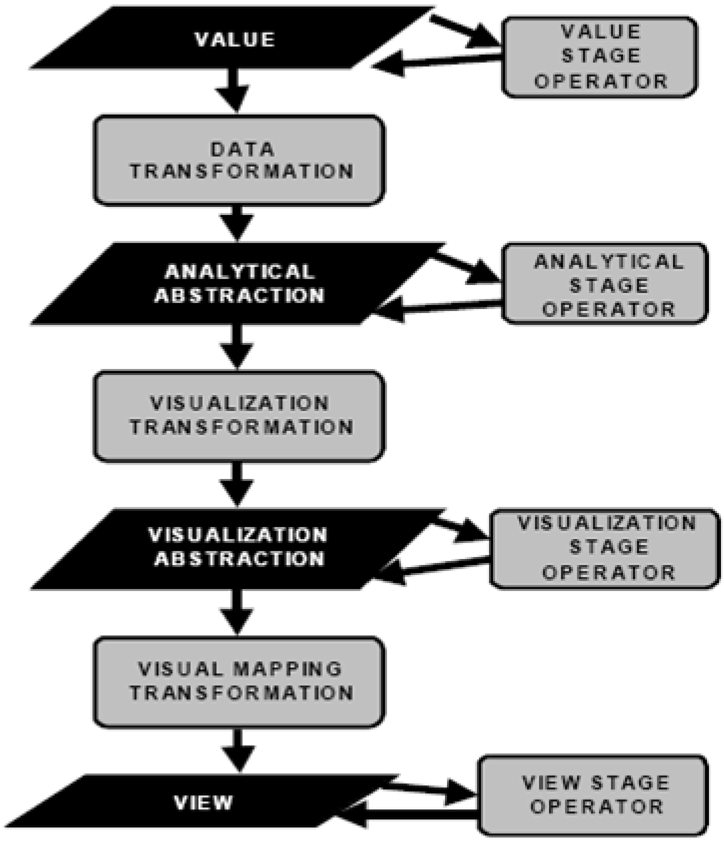

VALUE
VALUE STAGE OPERATOR
DATA TRANSFORMATION
ANALYTICAL ABSTRACTION
ANALYTICAL STAGE OPERATOR
VISUALIZATION TRANSFORMATION
VISUALIZATION ABSTRACTION
VISUALIZATION STAGE OPERATOR
VISUAL MAPPING TRANSFORMATION
VIEW
VIEW STAGE OPERATOR

Figure 2.18: The Data Stata Model
(Source: E.H.Chi)

The data flow model is another way of modeling the visualization transformation process. It is well established, especially in scientific visualization, and its capabilities and expressiveness is well-understood. The difference between the flow and the state model is that the latter captures distinct data states, while the data flow model captures the order of distinct processes that comprises a visualization process. However, it does not explore the relationship between view and value [34]. Real time data in all its relative complexity compared to static data will probably need a more thorough explanation. The data flow model may not be able to meet this requirement. A classification of real time data may be necessary.

2.6 Classification of information

According to MacKay [35] there are three types of information content:

- Selective information content, which helps the receiver make a selection from a set of possibilities

- Descriptive information content, which helps the user build a mental model.
- Semantic information content, which is between the physical signals and the users - and how they respond, which includes perceptional and cognitive psychology.

Presumably, real-time information possesses properties that fit all three information types, maybe with emphasis on the last- as user perception is essential in such communication. This suggests that testing of communication performance and proper classification of information could be a vital part of such a finding new ways to visualizing real-time data.

2.7 Measuring information visualization

System monitoring is of great importance and the design of system monitoring tools and its data visualization techniques is crucial to the interpretation of the data. In order to find out whether a specific information visualization is successful or not in communicating the intended information, the design's performance should be measured. There are several ways to gather data about a design's performance. Usability testing is arguably the most renowned method according to theorists such as Jacob Nielsen. Although not an interface, the monitoring prototype with the alternate visualization methods share many attributes typically used in mobile design, such as imagery, icon usage and information enlargement and enhancement, its interpretation and usage could therefore qualify as usability testing. A comparison to a classic visualization method would indicate whether or not the alternative visualization methods suggested are more effective in system monitoring specific usage.

2.7.1 Measuring usability

Accordning to Nielsen [36], usability is a quality attribute that assesses user-friendliness. There are several perceptual factors affecting the usability of an information display. Simplicity, memorability, predictability, and visibility are but a few design principles that may influence perception[6]. People have different cognitive capabilities, and a design that is easy for one person to understand and use, may be incomprehensible to another. A usability test can evaluate a design based on predetermined usability criteria. The data from such as test could either be quantitative (measurable) or qualitative (e.g. opinions).

Both data types can contribute to improving a design. Usability testing through controlled experiments is an important part of evaluation and a contribution to design improvement. During testing, statistically significant differences in time, error rate and satisfaction may be discovered, and observations recorded during the tests could become the basis for design refinement. Such benefits are often appealing to researchers. [37].

Testing performance of information visualization does however imply certain challenges. Tasks performed during laboratory sessions need to be simple enough to be accomplished within a predictable time frame, and at the same time be specific enough for performance to be measured [37]. For a convincing result, utility needs to be demonstrated in a real setting (or at least as real as possible), preferably within a relevant test domain with a set of users that belongs to that specific domain.

For researchers, choosing and preparing convincing examples and scenarios is of great importance, as well as using realistic datasets (of a certain size) and tasks. To achieve this in system monitoring, data sets should be sampled from actual data, test users should have some experience with the test context, and scenarios should be as authentic as as possible. The context is often the most difficult factor to recreate or mimic in an experiment.

A realistic mobile experiment context introduces complications when it comes to testing [38]. A real usage context would vary from time to time. For instance, a mobile device can be used in a variety of light conditions which could affect the perception of display content. Also, external circumstances or activities in which mobile users are taking part in can make it difficult to give the mobile device undivided attention.

A person would often have fewer cognitive resources available in a situation where it is natural to use a mobile phone. For instance, on a bus, sidewalk or at an airport. Although sounds and environments could be created with a great amount of effort, the test subjects's emotions and state of mind in that particular situation would be hard to recreate in a test lab. A user's normal work environment would perhaps provide at least some degree of realism as opposed to a constructed test lab, as most people use mobile phones while at work.

Quantifiable data is useful when looking for solutions to specific problems [39], as it is arguably easier to compare two discrete values, than interpreting subjective data. Such usability tests are often simple (data gathering by for instance counting or time taking) and can be applied to all data that can be measured in numbers.

A statistical paired two-sample t-test allows the researcher to compare to sets of numerical/quantitative data and determine whether or not the data sets are statistically equal or different. The P-value is a t-test parameter which rates the relationship and 5 percent (0.05) is considered to be a normal limit for determining significance. This suggests that the gathering quantitative measurements and performing t-tests potentially could help determining any difference between two visualization methods.

Nielsen [39], claims that usability is typically measured relative to the test subjects' performance on a given set of tasks, including:

- Success rate (whether users can perform the task at all)
- The time a task requires
- Error rate

The success rate is the simplest of all usability metrics: whether or not the test subject manages to complete a given task, and is calculated using the percentage of tasks that subjects complete correctly. A success rate does not indicate how well subjects perform the task or why they may fail. In order to rate the performance of a successful task, a useful and straightforward metric would be the time a user spends on completing a particular task.

Nielsen (IBID) suggests two ways of measuring the difference between two designs. The first method is for measuring the time it takes to perform a certain task. For instance, completing a task may take 100 seconds with one design, and 200 seconds with another. The difference in performance between the designs would then be 100 percent. Secondly, if there is a chain of tasks to complete, the tasks might often be unevenly performed, as test users for example could have individual interpretations of how to complete the task. If so, each task should be calculated separately.

For instance, if task one differs 100 percent between design one and two, task two differs 10 percent and task three differs 190 percent, the geometric mean of the differences would provide a better indication of the total difference in performance, as it excludes high values, and accounts more accurately for cases where metrics are negative. If the example above were split into three subtasks with 100, 10 and 190 percent difference, the result would instead be 57.5 percent, as opposed to 100 percent.

Furthermore, Nielsen [40] proposes at least twenty test subjects for quantitative usability testing. A balanced experiment requires rotation of the display sequence, which means that each image in the test should have the same position in the specific order an equal amount of times. For instance, if there are three displays in a balanced, quantitative experiment, each display should appear as number one, two and three in the sequence, totaling an equal amount of times in each position, making the total number divisible by the number of displays.

2.7.2 Mobile usability

Mobile screens of smaller size presumably increase the complexity of usability testing further. Singh et.al [41] found that test users comprehension score - reading user agreements on mobile screens were 20,3 percent lower that reading the same user agreement on a desktop screen, the only difference being the screen size.

Nielsen[42] identifies four problems with mobile usability: Small screens, awkward input, download delays and badly designed interfaces. A stand-alone system monitoring display (in its own right) would not suffer from awkward input, but if implemented in an application, or as part of a web site designed for mobile usage, this would likely apply.

Following user testing on mobile devices, Qiu et.al [43] addressed the difficulties with establishing a highly efficient user friendly design method for screens of limited size. In order to test the content of web sites using different techniques, information content had to be adapted a small screen display by using presentation optimization. The presentation optimization was based on certain psychological and statistical rules that determined the rank of importance. It should be reasonable to assume that importance and significance are likely to play a key part in determining the outcome of information visualization testing on mobile devices.

In a user test, Chittaro et.al [44] compared an alternative bar chart design especially constructed for mobile usage to a traditional bar chart design, using statistical significance in determining difference in design performance. Despite preliminary instructions about the alternate bar graph functionality, 3 out of 20 test subjects failed to understand the alternate design during testing, as opposed to 1 out of 20 on the traditional design. This suggests that experience is likely to affect the test result, and that it is potentially more powerful than short term memory during user testing. This may be useful information when making instructions for test users.

Chapter 3

Approach

3.1 Experiment preparations

In order to see if the proposed graph embellishments have an impact on users, a comparison to a classic visualization of the same data is desirable. Any difference between design performances can be detected and analyzed more accurately if based on quantifiable data. Therefore, it s desirable to gather usability and compare metrics from the two visualization methods: time line graphs with state and event enhancing embellishment and classical visualization of system monitoring.

3.1.1 Data types

System monitoring is often not about solving problems, or performing certain tasks. More often than not, system monitoring is about discovery. Seeing and analyzing trends and patterns in data - as well as establishing data states, depends on a reliable monitoring tool. Therefore it is reasonable to assume that qualitative or subjective data in a system monitoring usability test is unnecessary. Qualitative data would likely be a better and more realistic indication of the performance of a system monitoring display, as successful communication does not depend on a user's personal opinion. Therefore, the tests will be designed to acquire numerical data.

The alternate displays in this test adopt metaphoric imagery which is arguably new to system monitoring visualization. In addition to the metrics of the success rate, descriptions and comments uttered by the test subjects that specifically deals with the interpretation or description of this imagery were considered to be of interest. Especially the use of language when describing states, as this could give an indication as to how, for instance, a critical condition was perceived, as opposed to whether it was perceived or not.

Also, any correlation between the times spent interpreting the displays and the descriptions of the imagery, would be of interest. Therefore, additional specific phrases or comments made by the test user were noted for a somewhat qualitative addition to the gathered data sets (although counting word is arguably quantitative). The words and phrases that are uttered should be recorded cumulatively, in case the same test subject mentions more than one word or phrase. Also, any mentioning of changing line color and or the red warning frame is useful numerical data; in order to see whether or not these are easy to spot.

3.1.2 Display sequence

A balanced experiment would give each factor in the testing an equal amount of importance, significance and weight. Since this test contains more than one time line graph, it is desirable to compensate for any occurrence of skewing in results due to learning and memory usage after testing the first image. Therefore, the display order or sequence of the time line graphs in the test should be altered/rotated in an attempt to even out such factors. A sequential order of test displays, as the example in Table 3.1, may therefore promote data quality.

Example of sequence with 4 different displays				
1	Display 1	Display 2	Display 3	Display 4
2	Display 2	Display 3	Display 4	Display 1
3	Display 3	Display 4	Display 1	Display 2
4	Display 4	Display 1	Display 2	Display 3

Table 3.1: Example of sequence with 4 different displays

3.1.3 Postioning

Most displays are arguably oriented either horizontally or vertically which means that any rotation of the smart phone display during testing is undesirable. Therefore the web browser will be set to only read pages in a horizontal position. The advantage of having the phone handed out on a table before each view is that the facilitator can place the mobile phone in a correctly aligned position in terms of presenting the display to the test users, so that they doesn't have to spend any time or energy on screen alignment orientation. Although a smart phone is supposed to be hand held, having it placed on a table will arguably equalize conditions.

Also, since the phone is showing a display, as opposed to an interface, there is no need for smart phone functionality related to buttons or menus, which is arguably more accessible to the user when the phone is in a hand held position, as this is accustomed usage of a mobile device. A potential problem with having the smart phone on the table as opposed to in a hand held position is that it may remove some of the test subject's expectations to normal mobile usage, which again may influence user performance. However, when measuring success rate in seconds, the time saved on not enabling auto rotation may outweigh any loss of user performance as a consequence of not holding the device.

3.1.4 Mobile screen adjustment

In order to conduct a comparison experiment on two visualization methods, a desired prerequisite is that these should belong to the same usability domain and be compared in equal terms. Since the classic tool in this test possesses interface functionality (scrolling, links and menus), these should be removed from the experiment, leaving merely the graph image displaying the monitored data, since the focus of the thesis is the visualization method, and the basis for the measurements.

This suggests that links, menus and all navigational functionality should be removed from the classic tool. Furthermore, the graph from the classic monitoring tool is displayed as a PNG image file, which suggests that an similar display method (image) should be used for the other display in the comparison experiment - for the purpose of ensuring equality.

3.1.5 Expectations

Normal usage of a system monitoring tool may perhaps suggest that the user has prior knowledge about the device in question. The user may have some kind of knowledge prior to seeing the visualization that implies that the user possesses an expectation of what to see prior to seeing a visualization that may help speed up perception. For instance, a link displaying the name of the device may have been clicked in order to see the graph of that specific device. If the test user has no expectations of what to see in a display, the speed of perception would perhaps depend more on a successful design, as fewer cognitive resources are available.

Removing such expectations from the user while still establishing a system monitoring context may therefore improve data quality in a test of comparison. If the test user has little prior knowledge of a system monitoring visualization, this would arguably also improve context realism, as discovery is quite essential in system monitoring.

3.1.6 Test subjects

Given that the domain of the experiment is system monitoring, test subjects should have at least some experience with the domain of system monitoring to ensure a more viable result. IT-professionals that work in operations and have system monitoring experience - or master students in network and system administration would both be members of this domain.

Identities and job (or study) affiliations of the test subjects are likely to be subordinate when dealing with a perceptual user test and are therefore anonymized. When a connection to the domain of system monitoring is established, demographic data about the user is likely to become less interesting.

Experience with the classic visualization tool used in the testing may prove more useful, because this could say something about results being influenced by experience. However, experience will not be used as a decisive factor for determining whether or not a design is successful. Instead, experience could be used as a demographic-like indication after success rate has been determined.

Also, a pre-made written instruction could prevent any difference in perception of the instruction as a result of potential variation in a verbal improvisation.

3.1.7 Data collection

When collecting complex data, a voice recorder allows the facilitator to maintain focus on the test procedure as opposed to writing down answers and perform manual time taking. Analyzing a voice recording with audio playback software is likely to be more accurate than having a stop watch timing test subjects answers, as the software supports timing functionality used in the timing of tasks. This would also allow the facilitator to remain more focused during the experiment. Therefore, data from the user testing could be extracted by analyzing voice recorder sound files. Using audio playback software, answers from test users will be logged along with the time (in seconds) the test users spent on completing the points on a check list that constitutes the success rate.

3.1.8 Test scenarios

When visualizing data, it is important that the data set matches the visualization method, or else communication could fail. Therefore, the data sets used in the experiment had to fit the domain of the following data visualization methods suggested in the background chapter:

- Value-dependant line color (describes changing value states)
- Red frame along the edge (describes current state if critically high)
- Event timestamps (describes events)
- Snoring embellishment (describes idle or low activity state)
- Flame embellishment (describes critical usage state)
- Through-the-roof embellishment (describes critical usage state)

The experiment has to be constructed with simulated conditions that could somehow apply these methods. The data sets used in the testing has to contain critical states, low activity and clearly visible events or changes in the time line data. At the same time it is important to maintain some degree of realism in the data sets, adding credibility to the results. Testing and planning of what embellishment to use should be done with relative ease on real time data from the virtual server.

In order to test the effect of these methods, they should not be configured all at once to a single graph, but rather distributed in different graph scenarios. Chances are that too many embellishments would overlap or cancel each other out in an information overload. Also, embellishment that describes similar conditions in data state, such as the through-the-roof and the flame method (on critical values) would not be subjects to comparison if tested in the same time line graph.

Hence, the three most visual (in terms of size and appearance) state describing embellishments (Snoring, Flame and Through-the-roof) should be put in separate graphs. Value dependant line color was considered less eye catching than other data state embellishments and should be applied to all values above ninety percent. The snoring embellishment should perhaps have been applied to the scenario with the longest lasting idle state, or the value closest to zero. However, the idea behind the snoring imagery is to enhance the notion of idleness, hence a value greater than zero is considered a better test value than a zero value - in order to test if the user perceived a low activity state.

The most perceptually challenging embellishment is considered to be the event timestamps, as they are the smallest in size. The timestamps should be configured together with the snoring method, as the timestamps does not contain any dramatic imagery as opposed to the critical state methods.

The red critical state frame along the edge should be combined with the though-the-roof method, as the latter is considered more visual when placed at the end of the time line, in an attempt to create a somewhat dramatic context or effect. Flames are - arguably - imagery of a more dramatic character than the trough-the-roof effect, and a red critical state frame in conjunction with this effect is considered to even out any perceptional effect of the embellishments.

3.1.9 Constructing an alternative graph tool

An approach to displaying the data enhancing embellishment presented here would be to construct a configurable data visualization tool; a platform where these assumptions and visualization methods could be tested. Such a prototype tool should apply design elements described in the background chapter.

The tool should preferably be created on a platform that is web friendly, as browser access is desired. Parameter input from a web browser, without the need of input from other applications would ease configurability and arguably usability (the Ajax-powered Google Charts Tools provides an example of this functionality, where graph data and appearance is determined by parameters sent through an URL) [45] A mobile device may perhaps not contain or support programming specific functionality such as resources for recompiling and terminal support.

GD is a library for PHP5 that combines drawing with normal PHP functionality. GD (Graphic Draw) is an open source C written PHP5 library module that uses the concept of image streaming to project dynamic images onto a web browser. No image files generated by PHP is saved to disk (this is, however, an option). The output from a PHP script is instead interpreted in the browser as an image file.

From the command line on Linux operating systems, PHP may be used to perform shell-like operations, which makes it suitable for task automation. GD supports a variety of common graphic file formats, including JPEG, GIF, PNG SWF and TIFF. These attributes suggests that PHP5:GD may be a potentially efficient and fast tool of displaying script output.

The graph tool would need system data to display; regular sampling of system data. PERL could fetch and store data from system command outputs, and cron could automate the process. Basic, low complexity architecture would be sufficient for such a tool.

Visualizing fewer data sets in the same graph would arguably lower complexity and enchant communication. This suggests that different usage data should merge into fewer data sets (preferably as few as possible). Since merging is desirable, the tool should collect data from alternative sources that already takes use of aggregated data. In this way, no subjective influence on choosing the most appropriate data sets would occur. top (for CPU and memory) and df (for disk) are two common programs for displaying device usage on Linux.

data source input

```
Disk: df
CPU: top  -b -n 1 \%CPU column
Memory: top  -b -n 1 \%Memory column
```

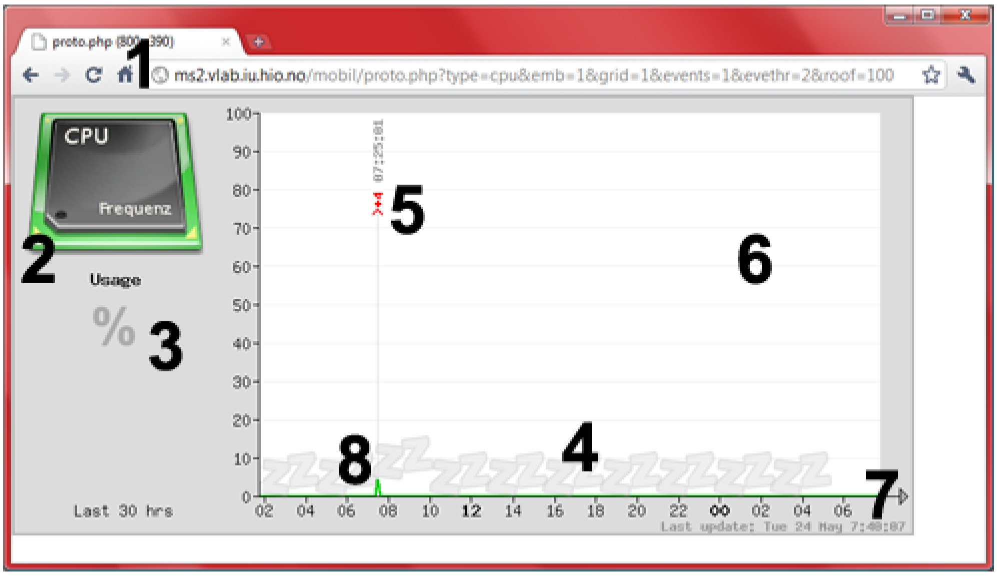
At this point, most of the planning had been conducted, and the project depended on the acquisition of a configurable tool in order to commence with the planning of the experiment.

Chapter 4

Results

4.1 Graph tool completion

In order to apply the visualization methods, imagery and embellishments discussed earlier, a tool had to be constructed. The guidelines in the approach were followed. The tool was gradually developed in stages. First the data collection script was written.



ms2.vlab.iu.hio.no/mobil/proto.php?type=cpu&emb=1&grid=1&events=1&evethr=2&roof=100

Figure 4.1: The alternative graph tool prototype explained

Figure 4.1 shows the prototype graph tool monitoring live data from a virtual server - as seen from a web browser. A short explanation of functionality and the idea behind follows (with explaining numbers in Figure 4.1):

1: The parameters in the visible URL is the configuration functionality of the graphs visual embellishment. The following parameters was programmed for this experiment:

Parameters explained

```
type  = (disk, cpu, memory)
emb = (1/0) on/off+
events  = (1/0) Timestamp on/off
evethr = (0-100) Event threshold. Determines when timestamp enchantment activates.
grid = (1/0) on/off Faded horizontal gridlines
roof = (1/0) on/off Lower data area 10\% from the top (Through-the-roof embellishment,
                                          creates the illusion of the time line going over max level )
warn = (0-100) Warning level. Determines when the value dependant line color changes
                                         and when a red frame appers around the graph indictating
                                         critical usage
```

2: Use of device icon imagery as opposed to text.
3: All usage is in percentage, filling the data area.
4: Embellishment for low (Snoring) and high (Flames) usage.
5: The timestamp embellishment helps the system administrator pinpoint increases or decreases in activity. Increasing value is visualized using a red color, decreasing value use green. The change in value is visualized numerically prior to the timestamp (in seconds)
6: Faded horizontal gridlines, as recommended by Tufte.
7: Timestamp of data source file last modified (how old the data are)
8: The time line is value dependant in regards to color. Green for low value and red for high (determined by warn level).

The prototype would run on the same server as the Munin installation, collecting data simultaneously for comparison purposes (Figure 4.2):

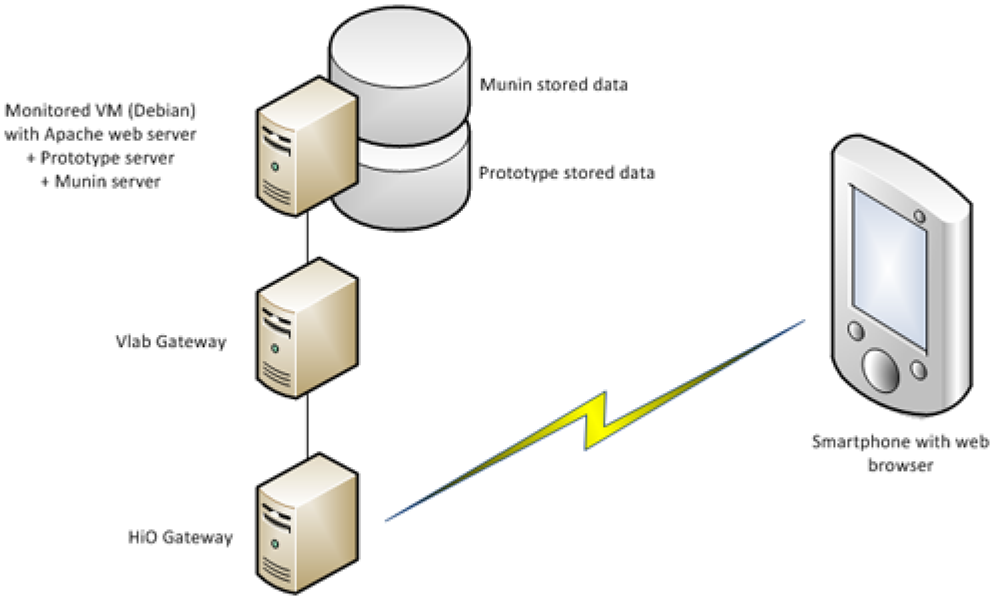

Munin stored data
Monitored VM (Debian) with Apache web server + Prototype server + Munin server
Prototype stored data
Vlab Gateway
Smartphone with web browser
HiO Gateway

Figure 4.2: The alternative tool prototype topology explained

4.2 Scenario design

Based on the assumptions in the approach chapter, the visualization setups were linked with one of the following monitored hardware devices: memory, disk and CPU - resulting in the following experimental setup scenarios (the choice of device is justified with regards to real world scenarios, although not without some improvisation):

Scenario 1: Normal memory usage (ten to twenty percent)
Visualization method: Event time stamp, Snoring.
Background: Memory usage is often low, yet constantly changing with normal usage, giving the impression of real time data.

Scenario 2: Full (close to one hundred percent) disk usage that changes to low, and then to half full.
Visualization methods: flames, value dependant line color
Reason: Disk usage arguably the most static of the three devices, suitable for a controlled manipulation of disk size over a thirty hour period.

Scenario 3: Idle CPU activity that rises to full (close to one hundred percent) activity
Visualization methods: Through-the-roof , value dependant line color, red warning frame.
Reason: CPU usage is often low with normal usage, yet spiky in appearance. A sharp spike at the end should be able to mimic a realistic event.

In order to prepare these scenarios for a user experiment, activities on the virtual test server were initiated and recorded by both monitoring tools over at least thirty hour period, in order to fill up the designated time span (enough to fill 30 hours) with data.

This is how the scenarios were constructed:

4.2.1 Scenario 1: Memory usage

The monitor tool recording of Scenario1 implied commencing normal activities such as for instance copying files and running several terminals. This was initiated while the monitoring tool recorded memory values. The intension behind the scenario was to create at least one time line graph based on normal activity. The result can be seen in Figure 4.3 and Figure 4.4).

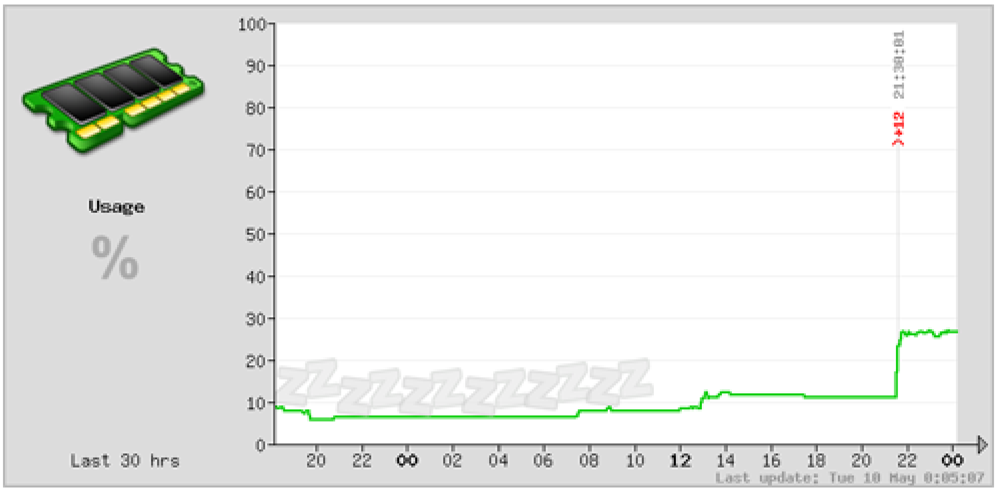


Figure 4.3: Scenario 1: Alternate visualization of memory usage

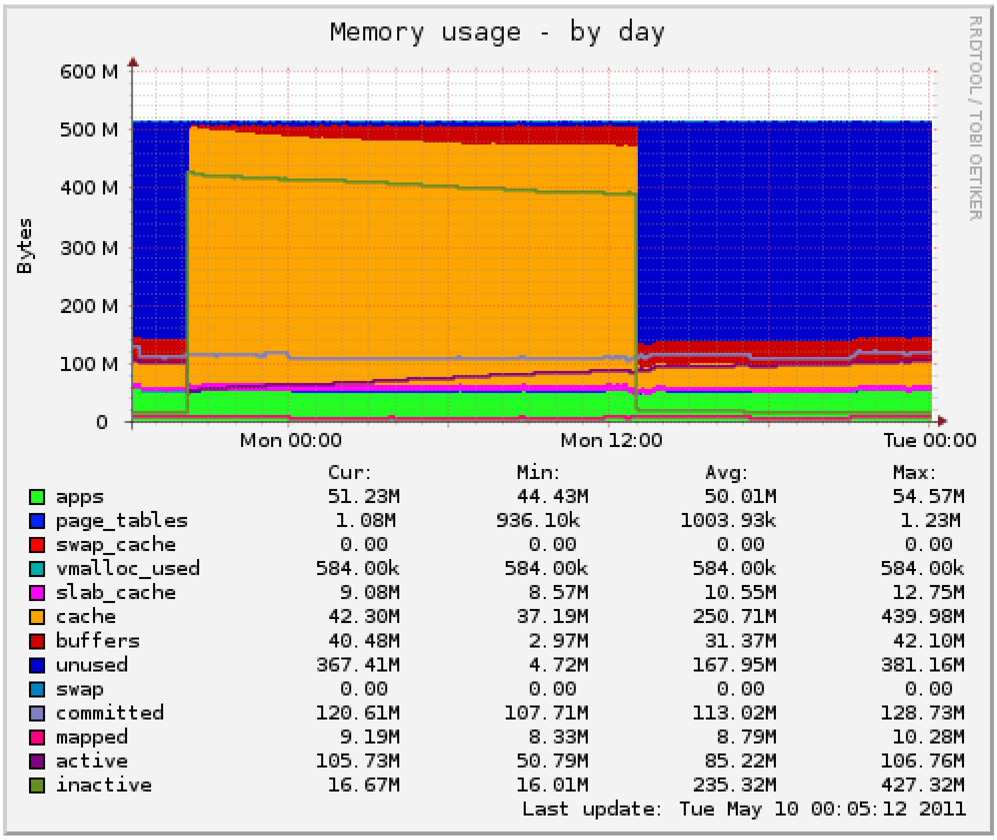
Explanation, Figure 4.3: In the beginning of the monitored time frame, starting the previous day, memory activity was low. The hardcoded snoring embellishment was initiated at 18:00 the previous day, lasting for about 16 hours, until memory usage passed the ten percent usage border and the snoring disappears. At exactly 21:30, three CPU-consuming Perl scripts were initiated which resulted in a twelve percent increase in memory usage, visualized by the event timestamp.

The following configuration of the alternative graph tool was used:

Scenario 1: configuration of the prototype graph tool

```
URL = ms2.vlab.iu.hio.no/mobil/proto.php
&type=memory
&emb=1
&grid=1
&events=1
&evethr=10
&roof=100
&warn=90
```

Simultainiously, the classic system monitoring tool visualized the same activity (Figure 4.4).



	Cur:	Min:	Avg:	Max:
🟩 apps	51.23M	44.43M	50.01M	54.57M
🟦 page_tables	1.08M	936.10k	1003.93k	1.23M
🟥 swap_cache	0.00	0.00	0.00	0.00
🟦 vmalloc_used	584.00k	584.00k	584.00k	584.00k
🟪 slab_cache	9.08M	8.57M	10.55M	12.75M
🟧 cache	42.30M	37.19M	250.71M	439.98M
🟥 buffers	40.48M	2.97M	31.37M	42.10M
🟦 unused	367.41M	4.72M	167.95M	381.16M
🟦 swap	0.00	0.00	0.00	0.00
▨ committed	120.61M	107.71M	113.02M	128.73M
🟥 mapped	9.19M	8.33M	8.79M	10.28M
🟪 active	105.73M	50.79M	85.22M	106.76M
🟩 inactive	16.67M	16.01M	235.32M	427.32M

Last update: Tue May 10 00:05:12 2011

Figure 4.4: Scenario 1: Classic visualization of memory usage

Explanation, Figure 4.4: The y axis in Figure 4.4 differs from Figure 4.3, showing megabytes as opposed to percentage. The big, orange 16 hour area which roughly coincides with the snoring embellishment in Figure 4.3 is a period where the disk on the virtual server was ninety-five percent full. The orange area represents memory cache, not actual usage. Although the memory sampled from the memory column output of top arguably does not tell the whole truth of Linux memory usage, Figure 4.3 communicates a far less dramatic state. The dark blue area is a visualization of the opposite of memory usage (unused memory). A grey line area near the bottom of the data field is committed memory usage, which is similar to that what is displayed by top.

4.2.2 Scenario 2: Disk usage

As prevoiusly stated, the purpose of Scenario 2 (Figure 4.5) and (Figure 4.6) was to visualize three different states of disk usage, the first one critically high, and a normal present state. The scenario was created using the dd command to quickly create and delete several huge files.

In the alternate visualization, all values above ninety percent were considered critical and visualized with the embellishment parameter turned on (Figure

4.5):

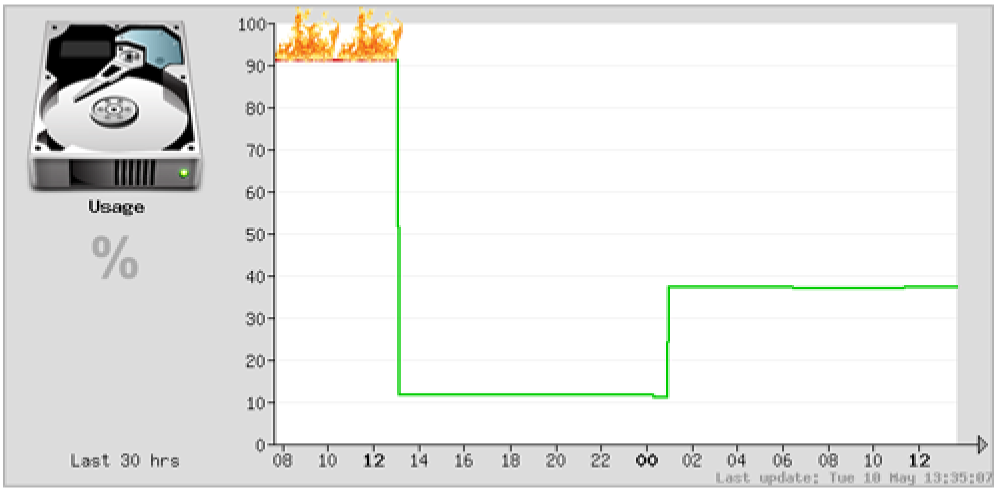


Figure 4.5: Scenario 2: Alternate visualization of disk usage

Explanation, Figure 4.5: The critical period (triggering flames and a red time line color) lasted until 13:00 the day before, when the files were deleted and usage was back to about 11 percent. These files occupied a total of ninety-five percent of the total disk space. Just before midnight, new files were created using dd, filling up a total of thirty-six percent of disk space. The current state of disk usage at the time of the screenshot sample was normal, not triggering any warning visualization effects.

The following configuration of the alternative graph tool was used:

Scenario 2: configuration of the prototype graph tool

```
URL = ms2.vlab.iu.hio.no/mobil/proto.php
&type=disk
&emb=1
&grid=1
&events=0
&evethr=0
&roof=100
&warn=90
```

Simultainously, the classic system monitoring tool visualized the same activity (Figure 4.6):

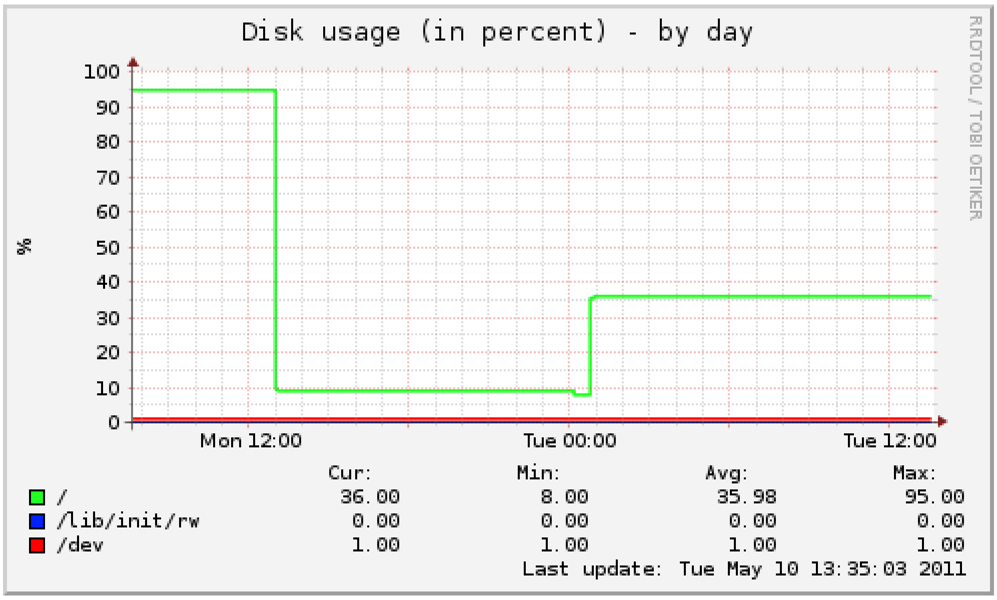


Figure 4.6: Scenario 2: Classic visualization of disk usage

Explanation, Figure 4.6: The major difference between the two visualizations, apart from the flame embellishment and the graph framework, is the extra timeline representing the /dev area on the virtual server. The y scale is identical to Figure 4.5 and the time line is of similar appearance.

4.2.3 Scenario 3: CPU usage

The purpose of scenario three (Figure 4.7) and (Figure 4.8) was to visualize two different states of CPU usage, the first one idle, and then rising to critically high - with a critical present state. The scenario was created using a Perl calculation script with the purpose of generating as much CPU usage as possible, running simultaneously in three different processes. The script was first tested using the `time` command, the modified (by increasing the number of loops) to last at least two hours. The result can be seen in Figure 4.7:

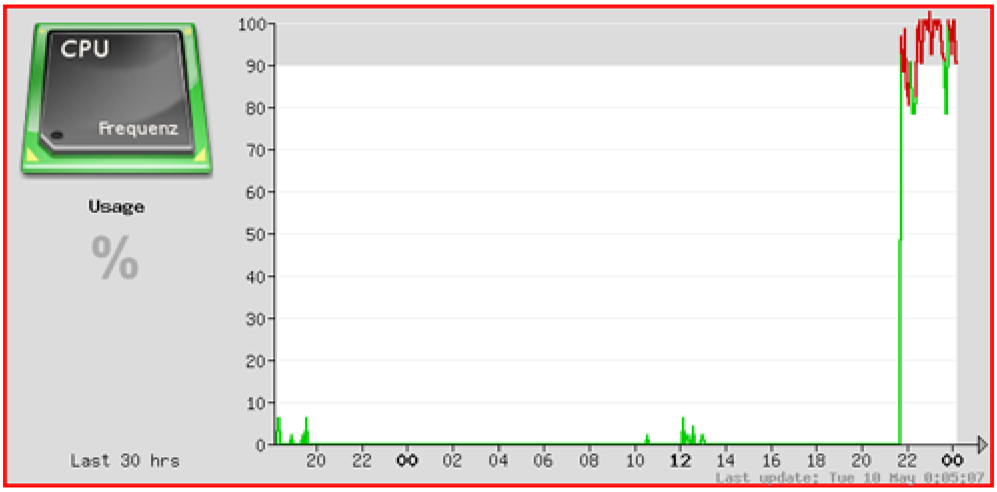


Figure 4.7: Scenario 3: Alternate visualization of CPU usage

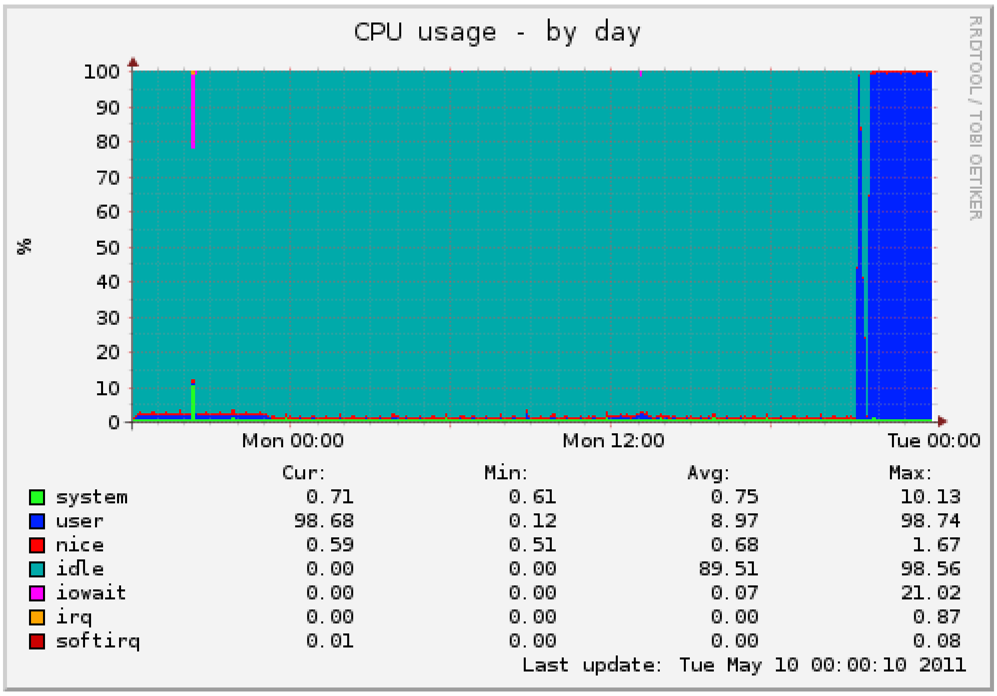
Explanation, Figure 4.7: State is normal (except from a few spikes) up until approximately 21:30, where CPU usage becomes critically high, still in critical state when the snapshot is taken. As a consequence, the red frame around the graph appears.

The following configuration of the alternative graph tool was used:

Scenario 3: configuration of the prototype graph tool

```
URL = ms2.vlab.iu.hio.no/mobil/proto.php
&type=cpu
&emb=0
&grid=1
&events=0
&evethr=0
&roof=90
&warn=90
```

Simultainously, the classic system monitoring tool visualized the same activity (Figure 4.8):



	Cur:	Min:	Avg:	Max:
🟩 system	0.71	0.61	0.75	10.13
🟦 user	98.68	0.12	8.97	98.74
🟥 nice	0.59	0.51	0.68	1.67
🟦 idle	0.00	0.00	89.51	98.56
🟪 iowait	0.00	0.00	0.07	21.02
🟧 irq	0.00	0.00	0.00	0.87
🟥 softirq	0.01	0.00	0.00	0.08

Last update: Tue May 10 00:00:10 2011

Figure 4.8: Scenario 3: Classic visualization of CPU usage

Explanation, Figure 4.8: The y axis in Figure 4.8 is identical to that of Figure 4.7, showing percentage. The data area is dominated by a turquoise area, replaced by a blue area at the end. The turquoise area is a visualization of the (arguably) opposite of CPU usage (idle CPU). The blue area is actual CPU usage triggered by user.

4.2.4 Data and display preparation

All scenarios were recorded by both the classic monitoring tool and the prototype. To ensure that any comparison between the classic display and the alternative display are made mainly on the visualization method (assuming that basis for the data is somewhat aligned), the data sets were fixed (locked) during testing to ensure identical hardware states.

As the monitored server continues to push data to both tools, changes in the data sets may occur between presentations during user testing. If a comparison between the two solutions were made using real time data, the experiment could be exposed to such differences. When the desired outcome was achieved, a snapshot of all graphs was created.

During the experiment, the web browser displayed the URL of the test display. Therefore, any hint of display functionality was concealed width by applying filename encoding in preparation for the experiment. These were the six displays that were used as test displays in the experiment:

Display overview and filename encoding				
Figure number	Display type	Content	Graph name	Filename
Figure 4.3	Alternative	Memory usage	Mem 2	vis1.png
Figure 4.4	Classic	Memory usage	Mem 1	vis4.png
Figure 4.5	Alternative	Disk usage	Disk 2	vis3.png
Figure 4.6	Classic	Disk usage	Disk 1	vis6.png
Figure 4.7	Alternative	CPU usage	CPU 2	vis5.png
Figure 4.8	Classic	CPU usage	CPU 1	vis2.png

Table 4.1: Display overview and filename encoding

4.3 Success criteria

With the scenarios ready, the objectives for achieving success rate on the displays were set. The tests user's perception of the different displays was measured with regards to success rate and time. In order to measure success rate, three criteria were set: two graph specific and one system monitoring specific. When the criteria were met, success was achieved, and the time that was used to meet the success criteria were measured. The graph specific criteria and the system monitoring criterion were split in two tests which in combination would measure any difference in design in each scenario. The criteria were defined as objectives that in total constituted the tests.

4.3.1 Test 1: Graph functionality (basic usage)

Objective 1: The test subjects identifies and mentions the domain of the graph: the specific device that is monitored (Memory, CPU or Disk) and what property of that specific device that is monitored. Fulfilling this objective would indicate that the test subject understands what kind of device and what kind of data the graph is displaying. This check point may occur after seeing explanatory text or graphics, and could be a measure of graph design performance.

Objective 2: The test subjects proves that they have understood the time span, time scale and time indication in the graph by either mentioning this specifically or that they correctly identify any events time stamps (meaning determining time of any changes in the time line, either approximately or accurately). The time span does not have to be identified entirely precise, since this would give the classic display a disadvantage (as gridlines will have to be counted to discover the thirty hour time span). Fulfilling this objective would indicate comprehension of both axis as well as graph functionality. Furthermore, it may occur after seeing explanatory text or graphics, and could be a measure of graph design performance.

4.3.2 Test 2: Trends, patterns and current state (monitoring specific)

Objective 1: The test subject recognizes and mentions any trends or states in values in the timeline graph. For example low, medium, high or critical usage. The test user must also identify the present state of data values. This is objective may arguably be the most important for system monitoring.

4.3.3 Score calculation

In order to determine success rate, specific criteria (objectives) were set to establish a measurement of perception. The test subjects would pass an objective they completed this objective as a result of understanding another objective. This also applied between tests within the same display, but not between displays. In case of doubt, as for instance if any mumbling occur on the audio recording, and if the test user has completed previous versions of the same monitoring tool (either the classic or the alternative) with success, the test user will pass.

The following answers constituted an achieved success:

Scenario 1: Memory usage, classic and alternate views:
Test 1, Objective 1: Memory usage in percent (alternative display) and Mb (classic display)
Test 1, Objective 2: An increase at 21:30 on the alternate display (almost invisible in the classic display). Increase at 20:00 the day before and decrease 13:00 on the classic display.
Test 2, Objective 1: All-over low memory usage.
Comments: The classical display (Figure 4.4) shows memory apps usage as a green area near the bottom which roughly corresponds with the time line in the alternative display (Figure 4.3). Since the two displays does not correspond with regards to objective two in test one, the criteria are adjusted.

Scenario 2: Disk usage, classic and alternate views:
Test 1, Objective 1: Disk usage in percent (both displays)
Test 1, Objective 2: Decrease at 13:00, increase at 01:00 (both displays)
Test 2, Objective 1: High or critically high usage in the beginning, now normal usage. Both must be mentioned in order to pass.

Scenario three: CPU usage, classic and alternate views:
Test 1, Objective 1: Disk usage in percent (both displays)
Test 1, Objective 2: An increase at 21:30 (both displays)
Test 2, Objective 1: Idle or low usage in the beginning, now critical usage. Both must be mentioned in order to pass.

4.3.4 Interpretation

As the user comments on graph content during the test, the order in which test subjects describes content will likely differ. The basic usage test and the monitoring specific tests are independent measurements, but run simultaneously. The times spent on these tests would overlap. At the start of each of the six displays, both timers start. When or if the test user complete one of the tests, the timer of that particular test stops, while the other continues until completion. In this way, the priority of the user is measured; whether they choose to complete test one or two first.

The times in all objectives are approximates. If the test subject answers somewhere in the proximity of the given time, it will count as a success.

The monitoring specific test were constructed in order to measure the (presumably) eye catching data embellishment in the alternative design to discover if there is a measurable difference in state description between the two methods. Any lack of difference between times in tests one and two may suggest that the test two was completed last.

If one or both objectives are achieved (or failed), the test is complete and a success or a failure is determined. The time the user spends on Test 1 is measured by stopping the timer at the point where it becomes clear that the user had completed the required objective(s). The time the user spent on Test 2 is - in similar fashion - measured by stopping the timer at the point where it becomes clear that the user have achieved the required objective.

4.4 Setup and execution

With the scenarios, tests and objectives ready, the setup and execution of the experiment could commence.

4.4.1 Location

The experiments were performed in a confined area (either a meeting room or an office) with only the facilitator and test subject present. The test subject was sitting down in front of a table or an office desk. It was ensured that the area had no strong glares or reflections from windows, lamps or other light sources, in order to keep distractions to a minimum. The door to the office or area was closed or area was closed, whenever possible.

4.4.2 Equipment

Test device: Smartphone with touch-sensitive screen, zoom and auto-rotation capabilities.
Model: HTC Desire
Weight: 135 grams
Screen size: 3.7 inches (94mm)
Screen resolution: 480 x 800 WVGA.
Display method: PNG picture files displayed by Android Web Browser v.2.2 (see Figures 4.9 and 4.10)

4.4.3 Participants

The main recruitment channel for obtaining participants was e-mail to key employees at different locations. These key employees then relayed the request. The rest of the participants were recruited in person at the locatation where they either worked or studied.

Identities of test subjects and work place were kept anonymous throughout this report. The number of the test subject, name, work place (or location) and time were recorded before the start of the test on a paper schema. The reason for writing down this data was to keep track of the changing sequence of displays used in testing and the filename of the sound recording (which had an incremental number in combination with the file name). Demographical test subject data such as sex, age and amount of experience was not recorded because this is presumably be of less importance than for instance the test subjects visual acuity and mobile experience.

Given that the total number of test displays was six and the recommendation of twenty test users for quantitative usability testing, the number of test users had to be both divisible by 6 and greater or equal to 20 in order to ensure a balanced experiment. This made the total number of test users in this experiment 24, which produced the test sequence shown in Table 4.2.

Sequence of test displays						
Subject no	Disp 1	Disp 2	Disp 3	Disp 4	Disp 5	Disp 6
1	Mem 2	CPU 1	Disk 2	Mem 1	CPU 2	Disk 1
2	CPU 1	Disk 2	Mem 1	CPU 2	Disk 1	Mem 2
3	Disk 2	Mem 1	CPU 2	Disk 1	Mem 2	CPU 1
4	Mem 1	CPU 2	Disk 1	Mem 2	CPU 1	Disk 2
5	CPU 2	Disk 1	Mem 2	CPU 1	Disk 2	Mem 1
6	Disk 1	Mem 2	CPU 1	Disk 2	Mem 1	CPU 2
7	Mem 2	CPU 1	Disk 2	Mem 1	CPU 2	Disk 1
8	CPU 1	Disk 2	Mem 1	CPU 2	Disk 1	Mem 2
9	Disk 2	Mem 1	CPU 2	Disk 1	Mem 2	CPU 1
10	Mem 1	CPU 2	Disk 1	Mem 2	CPU 1	Disk 2
11	CPU 2	Disk 1	Mem 2	CPU 1	Disk 2	Mem 1
12	Disk 1	Mem 2	CPU 1	Disk 2	Mem 1	CPU 2
13	Mem 2	CPU 1	Disk 2	Mem 1	CPU 2	Disk 1
14	CPU 1	Disk 2	Mem 1	CPU 2	Disk 1	Mem 2
15	Disk 2	Mem 1	CPU 2	Disk 1	Mem 2	CPU 1
16	Mem 1	CPU 2	Disk 1	Mem 2	CPU 1	Disk 2
17	CPU 2	Disk 1	Mem 2	CPU 1	Disk 2	Mem 1
18	Disk 1	Mem 2	CPU 1	Disk 2	Mem 1	CPU 2
19	Mem 2	CPU 1	Disk 2	Mem 1	CPU 2	Disk 1
20	CPU 1	Disk 2	Mem 1	CPU 2	Disk 1	Mem 2
21	Disk 2	Mem 1	CPU 2	Disk 1	Mem 2	CPU 1
22	Mem 1	CPU 2	Disk 1	Mem 2	CPU 1	Disk 2
23	CPU 2	Disk 1	Mem 2	CPU 1	Disk 2	Mem 1
24	Disk 1	Mem 2	CPU 1	Disk 2	Mem 1	CPU 2

Table 4.2: The complete test sequence in this experiment

The pattern in Table 4.2 repeats itself every sixth time to ensure a balanced experiment.

All test users was either IT professionals with at least some degree of system monitoring experience from both private and public sector, or master students in network and system administration with similar monitoring experience. None of the test subjects had any detailed knowledge of the project. 23 out of 24 test subjects commented in Norwegian. One test subject commented in English.

The facilitator informed the test subject about having a voice recording present during the test, and that all recorded material were due for deletion after transcription. Following consent, a portable, digital voice recorder was placed on a table near the test subject. The voice recorder then documented the test.

4.4.4 Instructions and execution

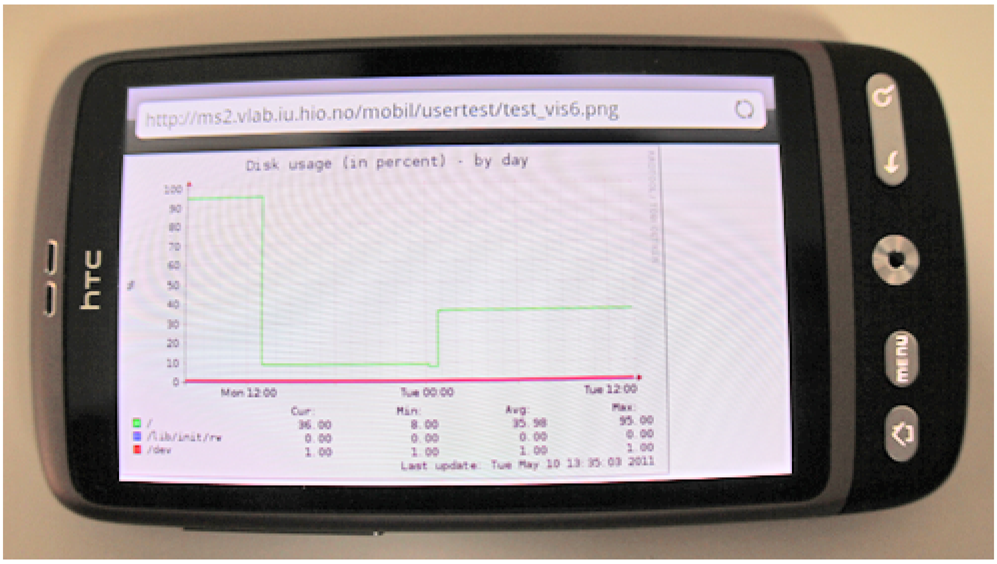
Before each test, a written instruction was read aloud to the test subject. The written instruction read as follows (translated from Norwegian):

"You are about to see six displays from a system monitoring tool - presented on a smart phone. The smart phone will be put on a table in front of you for you to look at. Your task is to describe what you see and think aloud. Focus on what you think is important. Any questions will not be answered during the testing. Do not pick up the phone. If you want to, you can touch the screen and rotate it on the table. When you are finished, let the facilitator know, and you will shortly be handed a new display until the test is finished."

in order to not reveal or give any hint of the identity of the three classic test displays, the following question was asked (and recorded) after the testing of a subject was complete:

"Have you had any experience with the Munin (classic) system monitoring tool?"

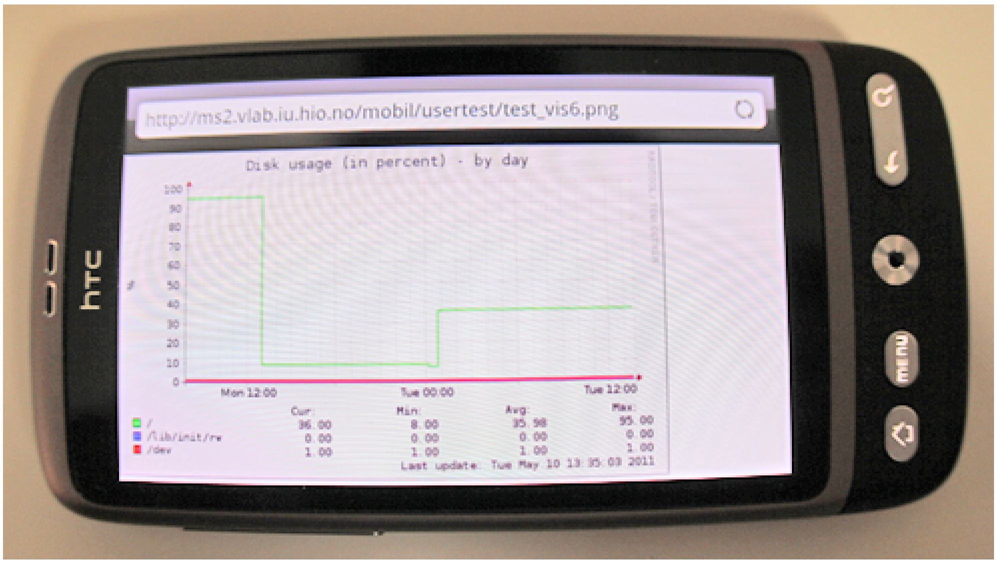
The excecution of the experiment was then carried out as described in this chapter and the approach. Here are two samples of the test device (Figures 4.9 and 4.10), as observed by the participants during the experiment:


http://ms2.vlab.iu.hio.no/mobil/usertest/test_vis6.png
Disk usage (in percent) - by day
100
90
80
70
60
50
40
30
20
10
0
%
Mon 12:00
Tue 00:00
Tue 12:00
htc
menu

	Cur:	Min:	Avg:	Max:
🟩 /	36.00	8.00	35.98	95.00
🟦 /lib/init/rw	0.00	0.00	0.00	0.00
🟥 /dev	1.00	1.00	1.00	1.00

Last update: Tue May 10 13:35:03 2011

Figure 4.9: A classic display (Disk 1) as seen on the mobile test device



	Cur:	Min:	Avg:	Max:
🟩 /	36.00	8.00	35.98	95.00
🟦 /lib/init/rw	0.00	0.00	0.00	0.00
🟥 /dev	1.00	1.00	1.00	1.00

Last update: Tue May 10 13:35:03 2011

Figure 4.10: A alternative display (Disk 2) as seen on the mobile test device

Explanation, Figures 4.9 and 4.10: This is two of the displays that the test subjects commented on in the experiment. The URL is clearly visible in the top of the screen.

Chapter 5

Findings and analysis

For the purpose of greater accessibility, analytical comments on the findings have been put in the same context as the data that these describe: the results from the experiment. These analytical comments are test specific, only describing data derived from single tests, not wholes. The discussion chapter will later put any findings into context.

The results from the experiment are presented here. Additional findings/discoveries related to the specific tests will be listed in conjunction with the test data.

In order to get an overview, the naming convention used in some of the graphs will be explained.

The six displays tested in this experiment had the following naming convention (examples provided):

- Classic displays = [short device name] + 1
  (e.g. "Mem 1" = Classic display of memory)
- Alternative displays = [short device name] + 2
  (e.g. "Disk 2" = Alternative display of disk)

This naming convention was applied for the sake of keeping track of the experiment. The graphs and tables presented in this chapter will use the type (classic/alternative) in situations where the naming convention is otherwise redundant.

5.1 Experiment results

Since experience was disregarded when performing t-tests, as the amount of participants was considered too low to split up, the bar charts displaying the results from the tests comparing the classic display and the alternative display only contain standard deviation plots on top of the overall bars (as these were used as basis for the tests). Experience was instead used as a basis for clues about usage patterns and to add futher depth to the gathered data.

5.1.1 Test 1 (basic usage)

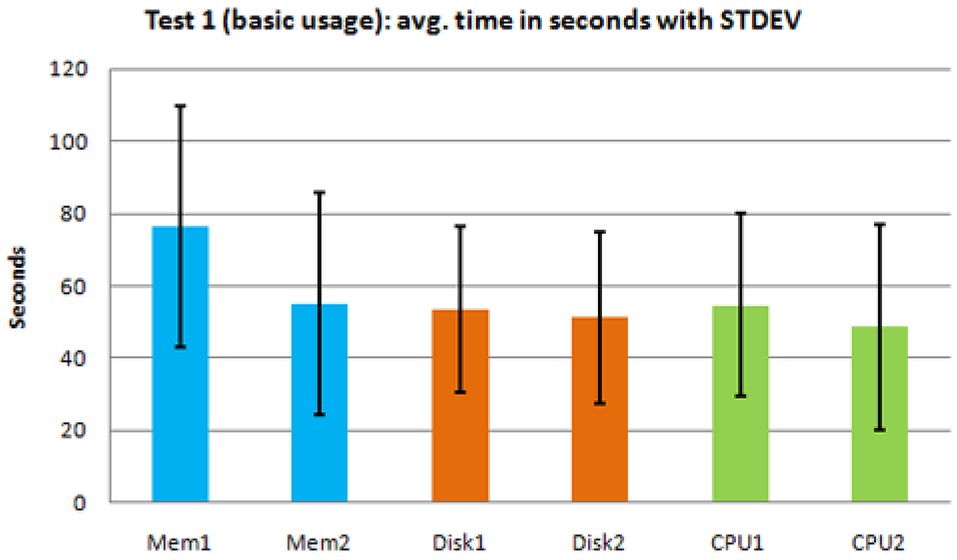


Figure 5.1: Test 1 (basic usage): Avereage time spent in seconds.

The time data (Figure 5.1) from the basic usage test shows a consistant, but varying difference between classic displays and alternative displays. In the memory scenario (Scenario 1), the difference in time is most evident. In the disk scenario (Scenario 2), the difference is noticeably smaller. Applying a higher level of detail to the analysis, the variation of the alternate displays of memory (Scenario 1) and CPU (Scenario 2) seems somewhat higher, considering the amount of seconds compared to the other displays.

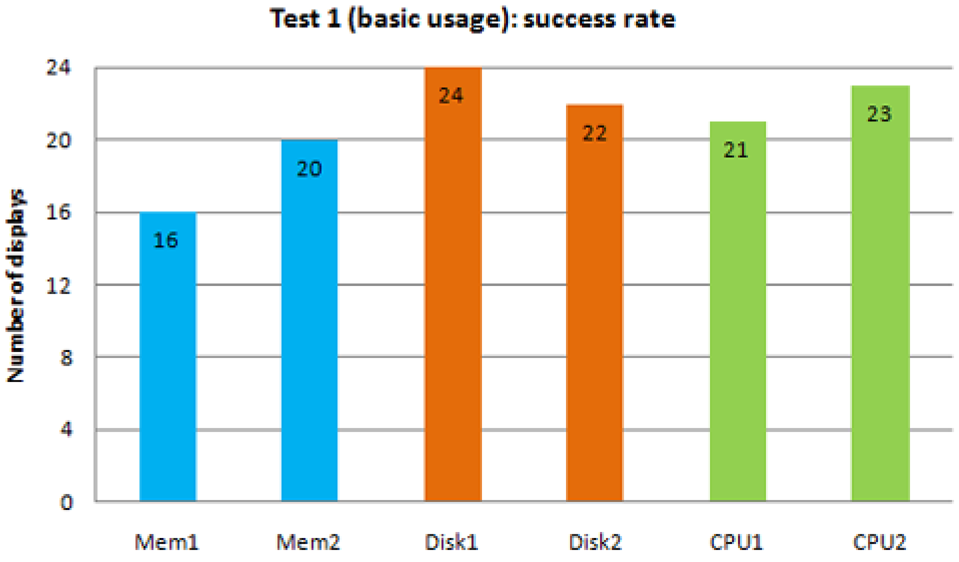


Figure 5.2: Test 1 (basic usage): Success rate

The success rate of Test 1 (Figure 5.2) varies between alternative and classic displays. In Scenario 2, all test subjects (24 out of 24) proved understanding of basic usage in the classic display, as opposed to 22 out of 24 on the alternative display. In Scenario 1 (memory) and 3 (CPU), the tendency is opposite; the alternative display performs better. Overall, Scenario 1 (memory) seem to perform slightly worse than the three other scenarios in the basic usage test.

Test 1 (basic usage): Results						
	Scenario 1		Scenario 2		Scenario 3	
	Mem 1	Mem 2	Disk 1	Disk 2	CPU 1	CPU 2
Avg. seconds	76,50	55,05	53,71	51,27	54,76	48,61
STDEV	33,42	30,58	22,96	23,68	25,07	28,42
Success rate	16/24	20/24	24/24	22/24	21/24	23/24

Table 5.1: Test 1 (basic usage): The results in numbers

5.1.2 Test 2 (monitoring specific)

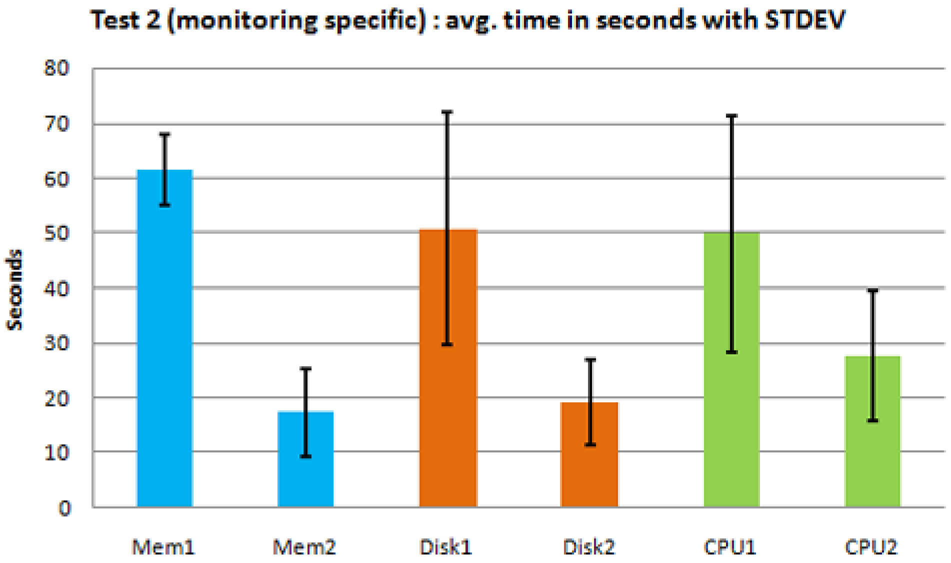


Figure 5.3: Test 2 (monitoring specific): Avereage time spent in seconds.

In Test 2 (the monitoring specific test), shown in Figure 5.3, there is an apparent tendency; classic displays seem to perform worse than alternative displays. This is backed up by t-tests performed on the equal pairs of data (Figures 5.7 and 5.10). The t-tests reveal a significant difference between the classic display and the alternate display in the monitoring specific tests in Scenario 2 (disk) and Scenario 3 (CPU). The monitoring specific test in Scenario 1 (memory) arguably did not produce sufficient data for a statistical significance test, but a hint of a similar tendency is evident. There seem to be an observable difference between Scenario 2 (disk) and 3 (CPU) as the alternative display of CPU seems to take somewhat longer. The variance on time spent is large in the classic displays is noticeable higher than in the alternative displays (disregarding Scenario 1, which produced inadequate data).

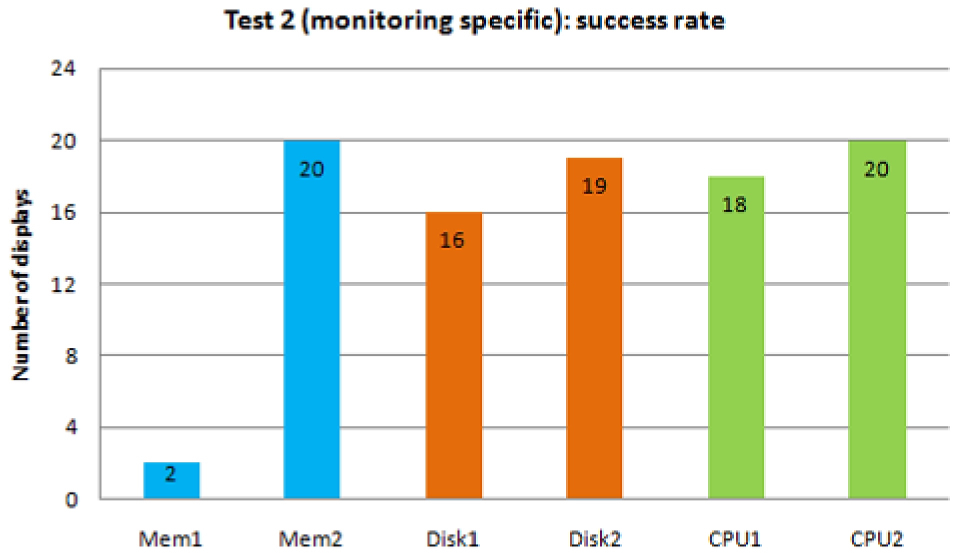


Figure 5.4: Test 2 (monitoring specific): Success rate

Success rate in Test 2 (monitoring specific) (Figure 5.4) seem to reflect some of the tendency as in the time spent (Figure5.3), although with quite smaller differences between classic and alternative displays. The success rate of the alternative displays is slightly higher than the classic displays in Scenario 2 and 3. The success rate among alternative displays is quite similar in all scenarios.

Test 2 (monitoring specific): Results

	Scenario 1		Scenario 2		Scenario 3	
	Mem 1	Mem 2	Disk 1	Disk 2	CPU 1	CPU 2
Avg. seconds	61,50	17,40	50,75	19,16	49,83	27,65
STDEV	6,36	7,96	21,24	7,80	21,54	11,80
Success rate	2/24	20/24	16/24	19/24	18/24	20/24

Table 5.2: Test 2 (monitoring specific): The results in numbers

The success rate is overall lower in Test 2 as opposed to Test 1 (comparison of Figure 5.1 and Figure 5.3), the greatest difference being the classic display of memory usage (Scenario 1) and classic display of disk usage (Scenario 2). The success rate of Test 1 and 2 was equal in the alterative display of memory (Scenario 1).

5.1.3 Scenario 1 (memory)

Test 1 (basic usage) results from the Mem 1 (classic) and Mem 2 (alternative) displays (with standard deviation on overall seconds used):

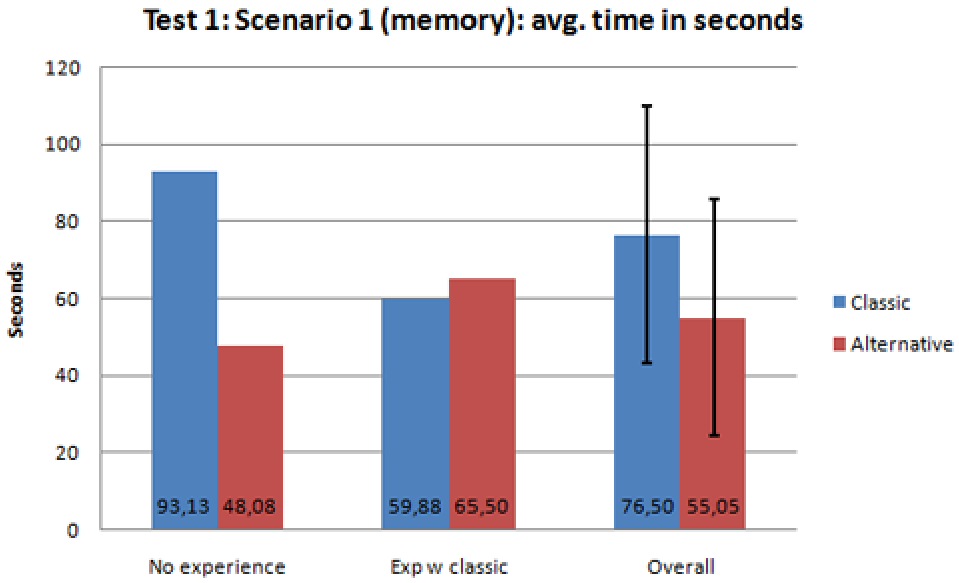


Figure 5.5: Scenario 1 (memory), Test 1: Average time spent in seconds

Users that have experience with the classic tool, tends to spend less time overall in completing Test1 on both types of displays (classic and alternative). In Scenario 1 (Figure 5.5), Test 1 is completed in shorter time amongst test users with experience. For test users without experience, time spent on Scenario 1 (memory) was approximately twice as high on the classic display as the time spent on the alternative display. The variance in time was similar and quite large in both display types.

Test 2 (monitoring specific) results from the Mem 1 (classic) and Mem 2 (alternative) displays (with standard deviation on overall seconds used):

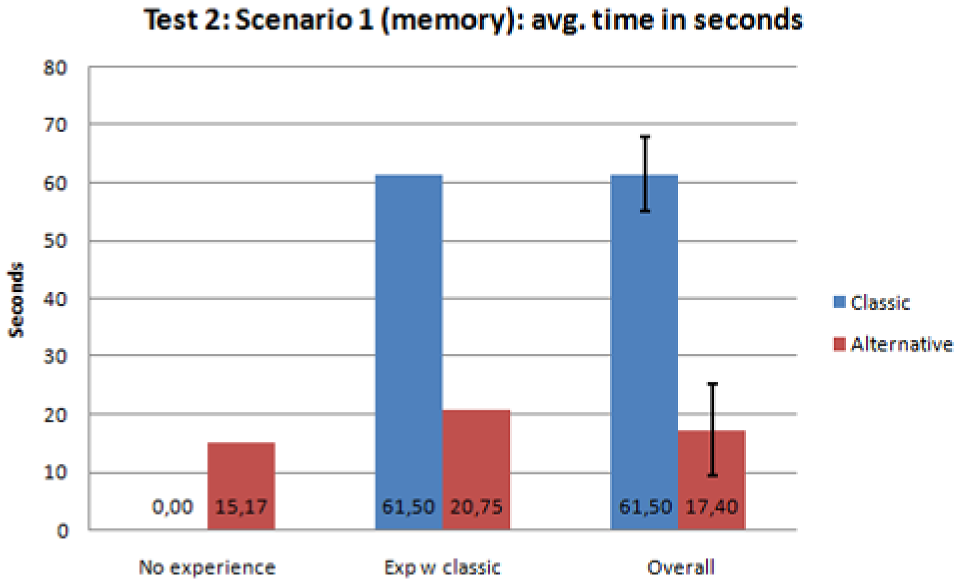


Figure 5.6: Scenario 1 (memory), Test 2: Average time spent in seconds

In the monitoring specific test 5.6 in Scenario 1, results were affected by that only two users were able to achieve success, leaving the basis for the data regarding the classic display arguably insufficient. Tests users with no classic tool experience spent noticeably less time than users with experience.

Scenario 1 (memory): Results				
	Test 1		Test 2	
	Mem 1	Mem 2	Mem 1	Mem 2
No experience	93,13	48,0	-	15,17
Exp w/classic	59,88	65,50	61,50	20,75
Overall	76,50	55,05	61,50	17,40
STDEV (Overall)	33,42	30,58	6,36	7,96
Success rate	16/24	20/24	2/24	20/24

Table 5.3: Scenario 1 (memory): The results in numbers

T-test: Paired Two Sample for Means		
Test 1 (basic usage)	Mem 1	Mem 2
Mean	78	63,21428571
Variance	1239,538462	1057,565934
Observations	14	14
Pearson Correlation	0,418428754	
Hypothesized Mean Difference	0	
df	13	
t Stat	1,511903452	
P(T<=t) one-tail	0,077240699	
t Critical one-tail	1,770933383	
P(T<=t) two-tail	0,154481398	
t Critical two-tail	2,160368652	

Table 5.4: Scenario 1 (memory), Test 1: Two paired t-test

The outcome of the t-test in Table 5.4 reveals no significant difference between the classic display and the alternate display since (t=1.51;p<.154;df=13).

No t-test was performed on Test 2 (monitoring specific) in Scenario 1 (memory), since the success rate was 2 of 24 (as seen in Figure 5.3 and Figure 5.6). This was concidered unsufficient data for a reliable outcome of a paired two sample t-test.

5.1.4 Scenario 2 (disk)

Test 1 (basic usage) results from the Disk 1 (classic) and Disk 2 (alternative) displays (with standard deviation on overall seconds used):

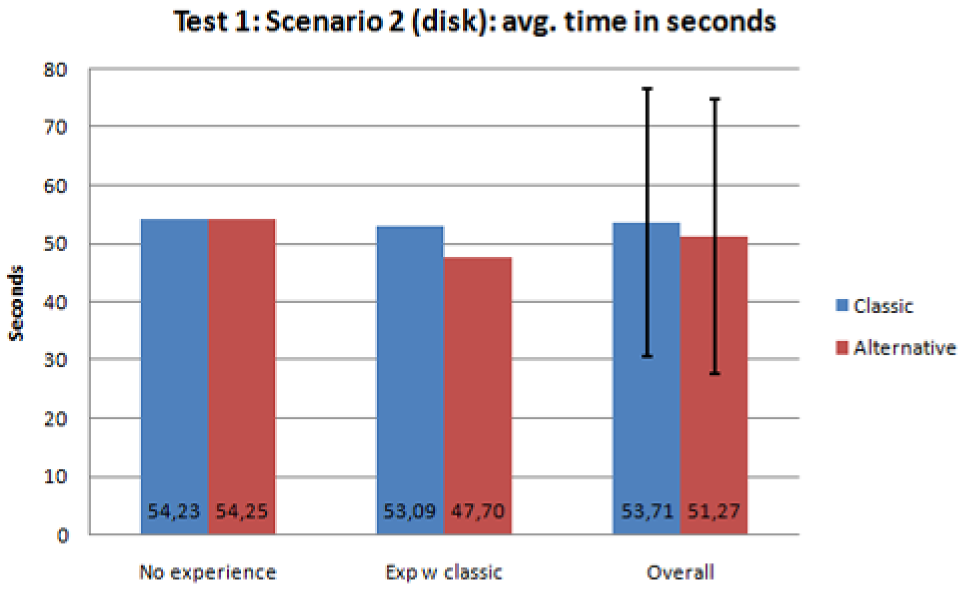


Figure 5.7: Scenario 2 (disk), Test 1: Average time spent in seconds

In Scenario 2 (disk) (Figure 5.7), the cross-experience results were quite similar in terms of basic usage (Test 1). There is no difference between people without tool experience in time usage. Among test users with experience, time spent on the alternative display was somewhat less (approximately 5 seconds in average) than the classic display. The variance in time was similar and quite large in both display types.

Test 2 (monitoring specific) results from the Disk 1 (classic) and Disk 2 (alternative) displays (with standard deviation on overall seconds used):

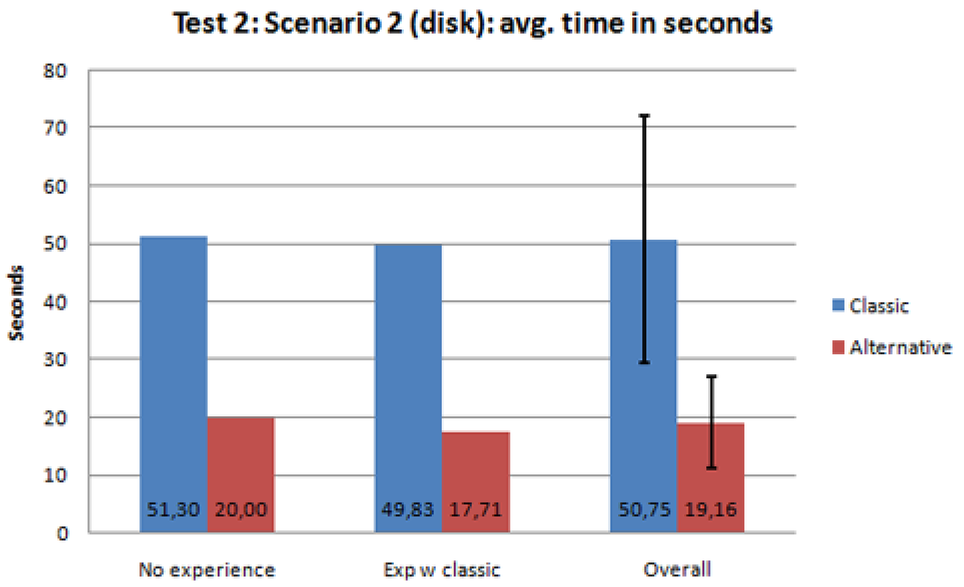


Figure 5.8: Scenario 2 (disk), Test 2: Average time spent in seconds

The monitoring specific test (Test 2) (Figure 5.8) in Scenario 2, also showed a very similar results across experience. Test users with experience performed marginally better on both display types. A t-test (Figure 5.7) proved statistical significance in overall difference between the performances of the two display types.

Scenario 2 (disk): Results				
	Test 1		Test 2	
	Disk 1	Disk 2	Disk 1	Disk 2
No experience	54,23	54,25	51,30	20,00
Exp w/classic	53,09	47,70	49,83	17,71
Overall	53,71	51,27	50,75	19,16
STDEV (Overall)	22,96	23,68	21,24	7,80
Success rate	24/24	22/24	16/24	19/24

Table 5.5: Scenario 2 (disk): The results in numbers

T-test: Paired Two Sample for Means		
Test 1 (basic usage)	Disk 1	Disk 2
Mean	55,22727273	51,27272727
Variance	517,6125541	560,7792208
Observations	22	22
Pearson Correlation	0,764327996	
Hypothesized Mean Difference	0	
df	21	
t Stat	1,161988764	
P(T<=t) one-tail	0,129138363	
t Critical one-tail	1,720742871	
P(T<=t) two-tail	0,258276727	
t Critical two-tail	2,079613837	

Table 5.6: Scenario 2 (disk), Test 1: Two paired t-test

The outcome of the t-test in Table 5.6 reveals no significant difference between the classic display and the alternate display since (t=1.16;p<.258;df=21).

T-test: Paired Two Sample for Means		
Test 2 (monitoring specific)	Disk 1	Disk 2
Mean	50,75	19,9375
Variance	451	62,32916667
Observations	16	16
Pearson Correlation	-0,21521509	
Hypothesized Mean Difference	0	
df	15	
t Stat	5,09360219	
P(T<=t) one-tail	6,60852E-05	
t Critical one-tail	1,753050325	
P(T<=t) two-tail	0,00013217	
t Critical two-tail	2,131449536	

Table 5.7: Scenario 2 (disk), Test 2: Two paired t-test

The outcome of the t-test in Table 5.7 reveals a significant difference between the classic display and the alternate display since (t=5.09;p<.000;df=15).

5.1.5 Scenario 3 (CPU)

Test 1 (basic usage) results from the CPU 1 (classic) and CPU 2 (alternative) displays (with standard deviation on overall seconds used):

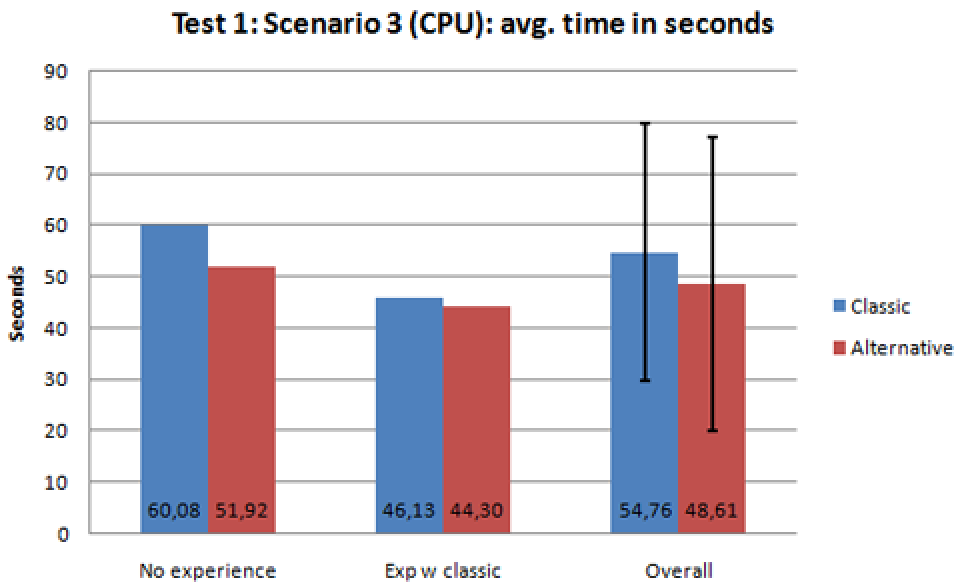


Figure 5.9: Scenario 3 (CPU), Test 1: Average time spent in seconds

Scenario 3 (Figure 5.9) produced data that also showed a difference between classic tool experiences. User with experience performed better on both display types (classic and alternative). Both groups performed somewhat better on the alternative display than the classic display. The difference between display types among user with experience was noticeably smaller than among the non-experienced.

Test 2 (monitoring specific) results from the CPU 1 (classic) and CPU 2 (alternative) displays (with standard deviation on overall seconds used):

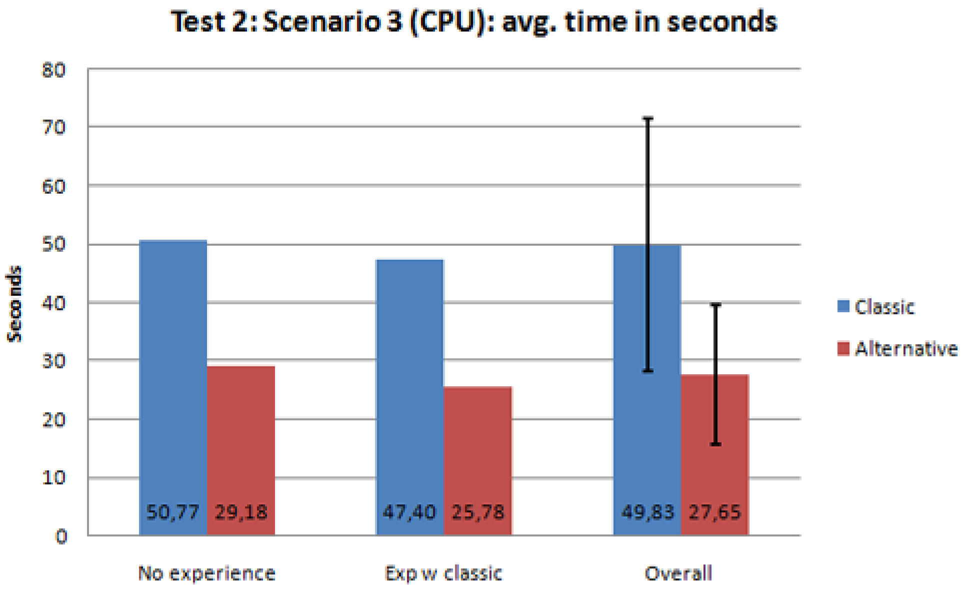


Figure 5.10: Scenario 3 (CPU), Test 2: Average time spent in seconds

The monitoring specific test in Scenario 3 (Figure 5.10), showed that the alternative display performed almost twice as well as the classic display type. A t-test (Figure 5.10) proved statistical significance in overall difference between the performances of the two display types. Overall, users with experience performed marginally better, approximately 3-4 seconds on both display types. The variance in time was somewhat larger on the classic display than the alternative display.

Scenario 3 (CPU): Results				
	Test 1		Test 2	
	CPU 1	CPU 2	CPU 1	CPU 2
No experience	60,08	51,92	50,77	29,18
Exp w/classic	46,13	44,30	47,40	25,78
Overalll	54,76	48,61	49,83	27,65
STDEV (Overall)	25,07	28,42	21,54	11,80
Success rate	21/24	23/24	18/24	20/24

Table 5.8: Scenario 3 (CPU): The results in numbers

T-test: Paired Two Sample for Means		
Test 1 (basic usage)	CPU 1	CPU 2
Mean	54,76190476	47,61904762
Variance	628,4904762	866,547619
Observations	21	21
Pearson Correlation	0,785120818	
Hypothesized Mean Difference	0	
df	20	
t Stat	1,78510721	
P(T<=t) one-tail	0,044709946	
t Critical one-tail	1,724718218	
P(T<=t) two-tail	0,089419891	
t Critical two-tail	2,085963441	

Table 5.9: Scenario 3 (CPU), Test 2: Two paired t-test

The outcome of the t-test in Table 5.9 reveals no significant difference between the classic display and the alternate display since (t=1.78;p<.0.08;df=20).

T-test: Paired Two Sample for Means		
Test 2 (monitoring specific)	CPU 1	CPU 2
Mean	49,1875	29,9375
Variance	445,2291667	134,3291667
Observations	16	16
Pearson Correlation	0,094644604	
Hypothesized Mean Difference	0	
df	15	
t Stat	3,334406211	
P(T<=t) one-tail	0,00226394	
t Critical one-tail	1,753050325	
P(T<=t) two-tail	0,004527881	
t Critical two-tail	2,131449536	

Table 5.10: Scenario 3 (CPU), Test 2: Two paired t-test

The outcome of the t-test in Table 5.10 reveals a significant difference between the classic display and the alternate display since (t=3.33;p<.004;df=15).

5.1.6 Summary of timed results

Experimental results: Time difference between displays and significance				
		Display		
Scenario	Test	Classic	Alternative	Significance
1 (memory)	1 (basic usage)	76,5 sec	55,0 sec	not significant
2 (disk)	1 (basic usage)	53,7 sec	51,2 sec	not significant
3 (CPU)	1 (basic usage)	54,7 sec	48,6 sec	not significant
1 (memory)	2 (monitoring specific)	61,5 sec	17,4 sec	not measurable
2 (disk)	2 (monitoring specific)	50,7 sec	19,1 sec	p<0.05
3 (CPU)	2 (monitoring specific)	49,8 sec	27,6 sec	p<0.05

Table 5.11: Summary: Time difference and significance

The time data comparison listed in Table 5.11, shows that although the completion of the alternative displays differ in time from the classic displays, there is no significant difference between the two data sets. The difference in seconds seems to be noticeably larger in Scenario 1 (memory) than in Scenario 2 (disk) and 3 (CPU), resulting in a difference of approximately 21 seconds, as opposed to 2 and 6 seconds. Although the difference between the alternative and classic display in Scenario 1 are not statistically significant, a time difference is observable. Applying a high level of detail, the variation of the alternate displays of memory (Scenario 1) and CPU (Scenario 2) seems somewhat higher, considering the amount of seconds compared to the other displays.

5.1.7 Frequency of words and phrases

Word frequency: Snoring embellishment (Scenario 1)	
"Sleep"/"Sleeping"	19 / 24
"Low"	17 / 24
"Idle" [eng.]	5 / 24
"Quiet"	1 / 24
"Calm"	2 / 24
No specific mentioning of imagery	1 / 24

Table 5.12: Word frequency: Snoring embellishment

Word frequency: Flame embellishment (Scenario 2)	
"Burn"/"Burning"/"Fire"/"Flames"	19 / 24
"High"	13 / 24
"Critical"/"Critically"	18 / 24
No specific mentioning of imagery	3 / 24

Table 5.13: Word frequency: Flame embellishment (Scenario 2)

Word frequency: Through-the-roof embellishment (Scenario 3)	
(Over) "max"/ "max level"	3 / 24
"Up"/"Straight up"	3 / 24
"Though/into the roof/ceiling/air"	14 / 24
"High"	12 / 24
"Critical"	5 / 24
No specific mentioning	5 / 24

Table 5.14: Word frequency: Through-the-roof embellishment (Scenario 3)

Frequency: Mentioning of embellishments			
	Scenario 1	Scenario 2	Scenario 3
Mentioning of changing line colors	N/A	2 / 24	8 / 24
Mentioning of red frame around graph	N/A	N/A	5 / 24
Mentioning of timestamp	10 / 24	N/A	N/A

Table 5.15: Mentioning of embellishments

The word- and phrase frequency summary showed that the Snoring embellishment was linked to the terms "sleep" and "low" most often. The fire embellishment was connected to "flames"/"fire" and "critical"/"critically" most often, with "high" as the third most popular term. In Scenario 3, the Through-the-roof method (Table 5.14) resulted in the phrase "through/into the roof/ceiling/air" as the most frequent description closely followed by "high". The mentioning of state enhancement visualizations' frequency in Table 5.15, shows that these more often than not is not mentioned. The least mentioned visualization seems to be the red frame around the graph (from Scenario 3).The timestamp in Scenario 1 was mentioned 10 times, but it the data does not say whether or not it was applied more times without being mentioned.

Chapter 6

Discussion and conclusion

The three alternative displays used in this experiment applied four suggested techniques/embellishments to enhance critical or critically high data states: Flames, Through-the-roof, changing line color and the red frame around the graph. The frequency of words used for description showed that 18 out of 24 participants mentioned the term (critical) when seeing the flames visualization method, as opposed to 5 participants when seeing the scenario with the Through-the-roof method. This may have influenced the monitoring specific measurement; average time spent on the Flame method was 19,1 seconds as opposed to 26,5 seconds with the Through-the-roof method. The data indicates that flames or fire imagery is more closely associated with a critical state, and that this again possibly indicates an influence perception performance.

The red frame around the graph, which purpose was to indicate a critical indication, was mentioned by 5 out of 24 subjects. While arguably occupying a substantial amount of total pixel space, it may have been obscured by other visual means sharing the same display. Also, the red frame did not interfere with visualized data, as it occupied the screen space closest to the mobile screen border. It is difficult to conclude whether or not the red frame had any influence on perception of the alternative display in Scenario 3 (CPU), or whether or not the method could have worked differently on a desktop monitor.

The same could arguably be said about the value dependant line color, which was mentioned 8 out of 24 times. In retrospective, it may have been hidden behind the flames embellishment, as only 2 out of 24 people mentioned it in Scenario 2 (disk).

A noticeable result during the word frequency counting was an often-heard phrase uttered by the participants when describing the Through-the-roof method. 14 of 24 test subjects used either the exact phrase as the name given the method (Through-the-roof), or something related, such as (straight into the air) [norwegian: rett i været], very close to what was described in the background chapter when this method first was explained. However, even though this effect was clearly recognized, no apparent correlation was drawn towards the term (critical).

When the snoring embellishment (in Scenario 1) was described by the test users, nobody used the term (snoring). Instead, 19 of 24 participants used the term (sleep) or (sleeping). This may indicate that the test user were familiar with the cartoon-like metaphor imagery for sleep. Furthermore, 20 of 24 test subjects completed the monitor-specific test in the snoring-embellishment scenario, at the lowest average time (17,4 seconds) for all scenarios. This makes it is reasonable to assume that the snoring embellishment for low activity data state made a positive impact on perception, in terms of interpreting the visualized data correctly and establishing a state.

In basic usage testing, the classic display types that applied null-value visualization, performed somewhat worse on time spent, than the classic display without null-value visualization - when compared to the alternative display. This may or may not be a coincidence, as several factors could influence the result.

The classical tool seemed to almost completely fail to communicate data state in the monitor specific test in Scenario 1 (memory), as 22 out of 24 subjects did no to recognize the state of memory usage. This may suggests that either the criteria used in establishing success was prejudiced or skewed against the classic tool, or maybe that the interpretation of memory usage in Linux is difficult to establish. On the other hand, the classic tools' visualization of memory (Figure 4.4 produced a somewhat complex expression, which also may have influenced the result. This question remains open.

A related problem is the merging of data used in this test to lower complexity. As stated, test subjects had considerable problems establishing data state in the classic display of memory. The time lines in the two display types did not seem to belong to the same device, due to reserved memory being accounted for differently by a different tool, which again raises the problem with choosing memory as visualized device in this experiment. On the other hand, maybe data should have been collected from the same source, even though this source possess no apparent definition of total memory usage. Visualizing memory (Usage) seems to require more than a single time line. Yet, due to the fact that failure was severe in determining state, there should be room for improvement.

The similar result in time data from Scenario 2 (disk) (Figure 5.7 could be related to the two displays being of relative similar appearance, when compared to the greater differences in time data in the other scenarios. Arguably, real life disk usage time lines (meaning the visualized data values) are not as square in appearance as in this experiment. More variable data values in disk usage could perhaps have produces another result.

The time the test users spent overall may have been a longer than intended, as interpreting a graph normally may occur more quickly. In fact some test subjects commented for well over two minutes, including irrelevant aspects of the displays. It is likely that some of the participants interpreted the task incorrectly, as some of the participants' focus were much wider than what was intended, sometimes on details that were irrelevant to the interpretation of the graph. A better designed instruction, perhaps with greater emphasis on importance and context might have resulted in a quicker interpretation and a different result. One could argue that the test subjects in question did not discriminate any display as they repeated the interpretation of the task during the entire experiment. One could also argue that absence of expectations contributed to longer perception during the entire test, since the monitored device was not revealed. In a real life monitoring situation, a system administrator is probably likely to spend less time.

The mobile platform could also have contributed to more time spent than expected, due to factors such as screen size. The decision to conduct the experiment on mobile equipment was made with regards to availability and the rapid growth of mobile technology and usage. Furthermore, future development of system monitoring tools is likely to be available on mobile platforms.

A mobile experiment context proved a challenging task to plan for, as assumptions and unknown factors complicates the planning phase. A controlled test environment using a desktop-sized monitor - although arguably more time-consuming to plan for and set up - is likely to be far less complex than a realistic mobile test environment. Researchers should continue to search for a viable methodology for mobile usability testing.

Measuring frequency and interpreting words and terms proved to be a challenge. For instance, due to factors such as mumbling, hesitation (as for instance stopping in the middle of a sentence) - or that the test user decides to have a change of mind (and perhaps take back what is already uttered). In a reproduction of a similar test, clearer guidelines for how to interpret test subjects oral answers would prove useful. Despite the fact that counting words is a quantitative task, there are many factors to consider when interpreting oral answers. For instance, letting the test user fill out a paper form with answers may have proved a better way to gather data.

Performing paired t-tests which decided statistical significance played an important part in determining inequality between data sets. In order to establish a sufficient basis for these statistical tests, difference in experience among test subjects was ignored to get enough data for the tests. In an experiment such as this where both success rate and time is measured, a paired t-test decreases the number of participants for significance calculation, as it only takes data where success is achieved in both displays into account. A test of different design (which ensured valid pairs of data) or an increased number of participants could potentially add credibility to the data. That being said, the two established differences in the monitoring specific test appear reliable.

The differences in experience proved not to be particularly determining factor, except from in the arguably most challenging display; the memory visualization (Scenario 1). Subjects with experience used over half a minute less in average on the basic usage test on the classic display.

I would have been interesting to see if there were experience dependant relationships that the test could establish, by increasing the number of participants in each group. Although trends are visible within the current data, a larger amount of data would secure a more reliable result and perhaps establish experience related findings.

This result from the experiment could potentially lead to alternative approaches to system monitoring visualization techniques. By applying metaphors and imagery, developers of visualization tools could take use of knowledge and predict likely reaction patterns at end-users. Although visualization theorists suggests that graphs should be without data-interfering embellishments (sometimes referred to as (chart junk)), other researchers have established improved memorability when well-designed graph embellishments has been applied to data visualization. In critical situations where risk is a considerable factor, the discovery of data state (for instance a critical state) should outweigh the desire to render the data correctly in visualization.

6.1 Conclusion

Alternative approaches for enhancing the visual expression of visualized data in system monitoring have been identified.

In order to empirically test the alternative visualization based on the principle of enchantment, an alternative monitoring tool was constructed. This tool was designed for a mobile platform and could exploit these methods through a web browser.

An experiment in form of a comparison between displays from a classic system monitoring visualization tool and the alternative tool was planned and conducted. The classic system monitoring visualization was adapted to fit a mobile screen to ensure equality across methods and subjects.

Scenarios were created using authentic data sets from three different devices: memory, disk and CPU. The sampled data was then visualized by both tools. The alternative monitoring tool applied the different visualization techniques suggested in the report - to different scenarios. Criteria were set to establish success, and in order to gather quantitative data, time was measured - both for basic graph usage and monitoring specific usage. Words and terms that the test subjects uttered when describing the embellishment in the alternative display were also included.

Differences in perception between the two display types were discovered using statistical t-tests. In basic graph usage the time difference between the classic display and the alternative display was insignificant, while the alternative display proved to be a significant improvement in monitoring specific usage.

Results and recommendations for future experiments has been mentioned and discussed.

The experiment established that use of imagery and embellishment significantly improved perception in terms of interpreting the state of visualized data in a mobile system monitoring context. Researchers and developers should consider making use of imagery and embellishments for the purpose of enhancing data states in system monitoring.

Bibliography

[1] Trafikanten RPS. April 2011. Sanntidsseminar for Norge, Trafikanten og Foreningen 177.

[2] Trond Heggelund. Article in Mobile Business Magazine Norway, april 2010. http://www.mobizmag.no/2011/04/07/9-av-10-solgte-mobiler-er-en-smarttelefon/.

[3] Edward R. Tufte. *Envisioning Information*. Graphics Press LLC, 1990.

[4] J Stanley Smith and James H. Fisher. Three mile island. *JAMA: The Journal of the American Medical Association*, 245(16):1656–1659, 1981.

[5] Geir Kaufmann Tore Helstrup. *Kognitiv psykologi*. Fagbokforlaget, 2000.

[6] Steven Heim. *The Resonant Interface*. Addison Wesley, 2007.

[7] Rayan Abdullah and Roger Hubner. *Pictograms, Icons and Signs*. Thames and Hudson, second edition, 2006.

[8] Ann Marie Seward Barry. *Perception, Image and Manipulation in Visual Communication*. State University of New York Press, Albany, 1997.

[9] John Lycett Louise Barrett, Robin Dunbar. *Human evolutionary psychology*. Princeton, NJ, US: Princeton University Press, 2002.

[10] Edward R. Tufte. *The Visual Display of Quanitative Information*. Graphics Press LLC, 2009.

[11] Michael W. Eysenck. *Principles of Cognitive Psychology*. Erlbaum (UK) Taylor and Friends, 1993.

[12] Nigel Holmes. *Nigel Holmes on Information Design*. Jorge Pinto Books Inc., 2006.

[13] Stephen Few. *Show Me the Numbers - Designing Tables and Graphs to Enlighten*. Analytics Press, 2004.

[14] Scott Bateman, Regan L. Mandryk, Carl Gutwin, Aaron Genest, David McDine, and Christopher Brooks. Useful junk?: the effects of visual embellishment on comprehension and memorability of charts. In *Proceedings of the 28th international conference on Human factors in computing systems*, CHI ’10, pages 2573–2582, New York, NY, USA, 2010. ACM.

[15] David McCandless. The beauty of data visualization. The beauty of data visualization, june 2010.

[16] M. Livio. *The golden ratio : the story of phi, the world's most astonishing number*. January 2002.

[17] Michael Friendly. Milestones in the history of data visualization: A case study in statistical historiography. In Claus Weihs and Wolfgang Gaul, editors, *Classification - the Ubiquitous Challenge*, Studies in Classification, Data Analysis, and Knowledge Organization, pages 34–52. Springer Berlin Heidelberg, 2005. 10.1007/3-540-28084-7_4.

[18] Michael Gilfix and Alva L. Couch. Peep (the network auralizer): Monitoring your network with sound. In *Proceedings of the 14th USENIX conference on System administration*, pages 109–118, Berkeley, CA, USA, 2000. USENIX Association.

[19] David P. Anderson, Jeff Cobb, Eric Korpela, Matt Lebofsky, and Dan Werthimer. Seti@home: an experiment in public-resource computing. *Commun. ACM*, 45:56–61, November 2002.

[20] Mitra Basu and Tin Kam Ho. *Data Complexity in Pattern Recognition (Advanced Information and Knowledge Processing)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[21] Guillermo A Francia, III. Visual security monitoring gadgets. In *Proceedings of the 5th annual conference on Information security curriculum development*, InfoSecCD '08, pages 40–43, New York, NY, USA, 2008. ACM.

[22] Emerson Rabelo, Maria Dias, Clelia Franco, and Roberto C. S. Pacheco. Information visualization: Which is the most appropriate technique to represent data mining results? *Computational Intelligence for Modelling, Control and Automation, International Conference on*, 0:1228–1233, 2008.

[23] Jakob Nielsen. Iterative user-interface design. *Computer*, 26:32–41, November 1993.

[24] Ann-Mari Torvatn. *Kommunikasjon for ingeniører*. Tapir Akademisk Forlag, 2001.

[25] Stephen Few. *Information Dashboard Design - The Effective Visual Communication of Data*. O'Reilly, 2006.

[26] Steven C. Seow. *Designing and Engeneering Time*. Addison-Wesley Professional, first edition, 2008.

[27] Natalia Andrienko T.J. Jankun-Kelly. Helen C. Purchase and Matthew Ward. *The Theoretical Foundations of Information Visualization*, volume 4950/2008. Springer Berlin / Heidelberg, July 2008. pages 46 -64.

[28] Alan Burns and Ian Hayes. A timeband framework for modelling real-time systems. *Real-Time Systems*, 45:106–142, 2010. 10.1007/s11241-010-9094-5.

[29] Maria Dias Cleila Franco, Emerson Rabelo and Roberto C. S. Pacheco. Information visualization: Which the most appropriate technique to represent data mining results? *CIMCA 2008, IAWTIC 2008 and ISE 2008*, 2008.

[30] Jacques Combaz, Jean-Claude Fernandez, Joseph Sifakis, and Loïc Strus. Symbolic quality control for multimedia applications. *Real-Time Syst.*, 40:1–43, October 2008.

[31] Tamara Munzner, Chris Johnson, Robert Moorhead, Hanspeter Pfister, Penny Rheingans, and Terry S. Yoo. Nih-nsf visualization research challenges report summary. *IEEE Computer Graphics and Applications*, 26:20–24, 2006.

[32] Ed Chi and Huai-Hsin. *A framework for information visualization spreadsheets*. PhD thesis, 1999. AAI9921428.

[33] Ed H. Chi. A taxonomy of visualization techniques using the data state reference model. In *Proceedings of the IEEE Symposium on Information Vizualization 2000*, INFOVIS '00, pages 69–, Washington, DC, USA, 2000. IEEE Computer Society.

[34] Ed H. Chi. Expressiveness of the data flow and data state models in visualization systems, 2001.

[35] Donald M. Mac Kay. *Information, Mechanism and Meaning*. The MIT Press, 1969.

[36] Jakob Nielsen. Usability 101: Introduction to usability. *Jacob Nielsen's Alertbox*, August 2003.

[37] Catherine Plaisant. The challenge of information visualization evaluation. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '04, pages 109–116, New York, NY, USA, 2004. ACM.

[38] Luca Chittaro. Visualizing information on mobile devices. *Computer*, 39:40–45, March 2006.

[39] Jakob Nielsen. Usability metrics. *Jacob Nielsen's Alertbox*, Januar 2001.

[40] Jakob Nielsen. Putting the user in user-interface testing. *IEEE Softw.*, 13:89–90, May 1996.

[41] Evaluating the readability of privacy policies in mobile environments. *International Journal of Mobile Human Computer Interaction (IJMHCI)*, pages 55–78, May.

[42] Jakob Nielsen. Mobile usability. *Jacob Nielsen's Alertbox*, July 2009.

[43] Mei Kang Qiu, Kang Zhang, and Maolin Huang. An empirical study of web interface design on small display devices. In *Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence*, WI '04, pages 29–35, Washington, DC, USA, 2004. IEEE Computer Society.

[44] Luca Chittaro and Alberto Camaggio. Visualizing bar charts on wap phones. In Fabio Paterno, editor, *Human Computer Interaction with Mobile Devices*, volume 2411 of *Lecture Notes in Computer Science*, pages 101–101. Springer Berlin / Heidelberg.

[45] Google Inc. Google chart tool. http://code.google.com/apis/chart/interactive/docs/.

Chapter 7

Appendix

7.1 Prototype graph script (PHP5 with GD)

proto.php

```php
<?php

// -- get variables --
// get device type (disk(default),cpu,memory)
$type = $_GET[’type’];

// display embelishment (1)
$emb = $_GET[’emb’];

// display events (1)
$events = $_GET[’events’];

// determine event threshold (0-100)
$evethr = $_GET[’evethr’];

// display horizontal gridlines (1)
$grid = $_GET[’grid’];

// display roof 90% of normal size (90)
$roof = $_GET[’roof’];

// set warning visualzation level (0-100) default = 90
$warninglevel = $_GET[’warn’];
if ( $warninglevel == NULL ){
    $warninglevel = 90;
}

function drawGraph($type,$emb,$events,$evethr,$grid,$roof,$warninglevel) {

    // graph coordinate constants and helpers

    $yStart = 355;
    $xStart = 220;
    $counter = 0;
    $findnumlines = 0;
    $maxr = 0;
    // $eventcounter = 0;

    // create canvas

    $img = imagecreatetruecolor(800, 390);

    // define graph colors

    $white = imagecolorallocate($img, 255, 255, 255);
    $red = imagecolorallocate($img, 255,0,0);
    $grey = imagecolorallocate($img, 150, 150, 150);
    $lightgrey = imagecolorallocate($img, 220, 220, 220);
    $gridgrey = imagecolorallocate($img, 240, 240, 240);
    $green = imagecolorallocate($img, 0, 255, 0);
    $black = imagecolorallocate($img, 0, 0, 0);

    // hardcoded configurations

    if ($type == "memory"){
        $src = imagecreatefrompng(’memory.png’);
        $theFile = "memdata.txt";
    }
    elseif ($type == "cpu") {
        $src = imagecreatefrompng(’cpu.png’);
        $theFile = "cpudata.txt";
    }

    // set default to disk, ease of url access

    elseif ($type == "disk" || $type != NULL || $type == NULL ) {
        $src = imagecreatefrompng(’disk.png’);
        $theFile = "diskdata.txt";
    }

    imagefilledrectangle($img, 0, 0, 800, 390, $lightgrey);

    if ( $roof == 90 ) {
        imagefilledrectangle($img, 219, 50, 769, 355, $white);
    } else {
        imagefilledrectangle($img, 219, 15, 769, 355, $white);
    }

    $fire = imagecreatefrompng(’firesmall.png’);
    $zz = imagecreatefrompng(’zz.png’);
    $percent = imagecreatefrompng(’percent.png’);
    $arrow = imagecreatefrompng(’arrow.png’);

    // postion the device type image, x axis arrow and percentage

    imagecopy($img,$percent,70,185,0,0,40,38);
    imagecopy($img,$src,10,0,0,0,160,160);
    imagecopy($img,$arrow,770,348,0,0,26,15);

    // postion the y helpers & gridlines (grid = optional)

    for ($x=0;$x<11;$x++){
        $drawYhelp = $yStart - ( $x * 34 );
        imageline($img, 214,$drawYhelp,219,$drawYhelp,$black);
        if ( $grid == 1 && $x != 0 ){
            imageline($img, 220,$drawYhelp,769,$drawYhelp,$gridgrey);
        }
        if ( $x == 0 ){
            imagestring($img, 4, 204, $drawYhelp - 7, $x * 10, $black);
        } elseif ( $x == 10 ){
            imagestring($img, 4, 189, $drawYhelp - 7, $x * 10, $black);
        } else {
            imagestring($img, 4, 196, $drawYhelp - 7, $x * 10, $black);
        }
    }

    // configure brushes for various purposes including the main time line

    $brush_size = 2;
```

```
$brush = imagecreatetruecolor($brush_size,$brush_size);
$brush_darkgreen = imagecolorallocate($brush,0,205,0);
$brush_darkred = imagecolorallocate($brush,205,0,0);
$brush_red = imagecolorallocate($brush,255,0,0);
$brush_black = imagecolorallocate($brush,0,0,0);
$brush_grey = imagecolorallocate($brush,180,180,180);

// open file, read only

$fh = fopen($theFile,"r") or die("Failed to open $theFile");

// check length, we only want the first 30 hrs, 360 lines

exec("wc -l $theFile",$out);
$splitout = preg_split("/\s+/",$out[0]);
$linestotal = intval($splitout[0]);

if ( $linestotal > 360 ) {
    $skiplines = $linestotal - 360;
} else {
    $skiplines = 0;
}

// read file

while(! feof($fh)){

    $linedata = fgets($fh);

    // read bottom 360 lines in csv file

    if ( $skiplines < 1 ){
        list($fulldate, $timestamp, $datavalue) = split(",", $linedata);
        $hour = substr($timestamp,0,2);
        $minute = substr($timestamp,3,2);

        if ($datavalue != "" ){
            $r = floatval($datavalue);
            if ( $r > $maxr ){
                $maxr = $r;
            }
        }
        $drawXstart = $xStart + ( $counter * 1.5277 );
        $drawXstop = $drawXstart + 1;
        if ($counter == 0){
        $prevr = $r;
        }

        // scale to fit into data area

        $drawr = ( $r * 3.4 );
        $drawprevr = ( $prevr * 3.4 );
        $drawmaxr = ( $maxr * 3.4 );

        // check for difference and apply timestamp event visualization
        // concider currect vertical position of time line

        $diff1 = number_format(($prevr - $r), 0);
        $diff2 = number_format(($r - $prevr), 0);
        if ( (($diff1 > $evethr) || ($diff2 > $evethr)) && $events == 1 ){
            if ( $r <= 55 ){
                if ( $roof == 90 ){
                    imagefilledrectangle($img, $drawXstart-4, 16, $drawXstart+6, 48, $lightgrey);
                    imagefilledrectangle($img, $drawXstart-4, 49, $drawXstart+6, 114, $white);
                } else {
                    imagefilledrectangle($img, $drawXstart-4, 16, $drawXstart+6, 114, $white);
                    }
                    imagestringup($img, 3, $drawXstart-5, 75, $timestamp, $grey); // timestamp
                    if ( $diff1 > 1 ){
                        if ( $diff1 < 10 ){
                            imagestringup($img, 3, $drawXstart-5, 105, ’<-’.$diff1, $green);
                            imageline($img, $drawXstart + 1, $yStart - $drawr, $drawXstart + 1, 104, $lightgrey);
                        } else {
                            imagestringup($img, 3, $drawXstart-5, 113, ’<-’.$diff1, $green);
                            imageline($img, $drawXstart + 1, $yStart - $drawr, $drawXstart + 1, 112, $lightgrey);
                        }

                    } else {
                        if ( $diff2 < 10 ){
                            imagestringup($img, 3, $drawXstart-5, 105, ’>+’.$diff2, $red);
                        } else {
                            imagestringup($img, 3, $drawXstart-5, 113, ’>+’.$diff2, $red);
                        }
                        imageline($img, $drawXstart + 1, $yStart - $drawr, $drawXstart + 1, 104, $lightgrey);
                    }
                } else {
                    imagefilledrectangle($img, $drawXstart-4, 258, $drawXstart+6, 355, $white);
                    imagestringup($img, 3, $drawXstart-5, 348, $timestamp, $grey); // timestamp
                    if ( $diff1 > 1 ){
                        if ( $diff1 < 10 ){
                            imagestringup($img, 3, $drawXstart-5, 286, ’-’.$diff1.’>’, $green);
                        } else {
                            imagestringup($img, 3, $drawXstart-5, 286, ’-’.$diff1.’>’, $green);
                        }
                        imageline($img, $drawXstart + 1, $yStart - $drawr, $drawXstart + 1, 237, $lightgrey);
                    } else {
                        if ( $diff2 < 10 ){
                            imagestringup($img, 3, $drawXstart-5, 286, ’+’.$diff2.’>’, $red);
                            imageline($img, $drawXstart + 1, $yStart - $drawr, $drawXstart + 1, 267, $lightgrey);
                        } else {
                            imagestringup($img, 3, $drawXstart-5, 286, ’+’.$diff2.’>’, $red);
                            imageline($img, $drawXstart + 1, $yStart - $drawr, $drawXstart + 1, 259, $lightgrey);
                        }
                    }
                }
                // $eventcounter++;
            }

            // start main time line

            // apply check for warning value

            if ( $prevr >= $warninglevel && $r >= $warninglevel ){
                imagefill($brush,0,0,$brush_darkred);
                imagesetbrush($img,$brush);
                imageline($img, $drawXstart,$yStart-$drawprevr,$drawXstop,$yStart-$drawr,
IMG_COLOR_BRUSHED);
            } else {
                imagefill($brush,0,0,$brush_darkgreen);
                imagesetbrush($img,$brush);
                imageline($img, $drawXstart,$yStart-$drawprevr,$drawXstop,$yStart-$drawr,
IMG_COLOR_BRUSHED);
            }

            // end main time line

            $prevr = $r;
            $counter++;

            // apply embellishment

            if ($counter % 33  == 0 ){
```

```
            if ( $maxr < 10 && $emb == 1 ){

                // sleep if low activity

                imagecopy($img,$zz,$drawXstart-50,$yStart-($drawmaxr + 40),0,0,55,40);
                $maxr = 0;
            } elseif ( $maxr >= 90 && $emb == 1 ) {

                // flames if high activity

                imagecopy($img,$fire,$drawXstart-50,$yStart-($drawmaxr + 52),0,0,55,54);
                $maxr = 0;
            }
        }

        // apply and pinpoint hour helpers to x axis

        if (($hour % 2 == 0) && ($minute == "00")){
                if ( $hour == "00" || $hour == "12" )
            {
                imagestring($img, 5, $drawXstart-8, $yStart+5, "$hour", $black);
            } else {
                imagestring($img, 4, $drawXstart-8, $yStart+5, "$hour", $black);
            }
            imageline($img, $drawXstart, $yStart+1, $drawXstart, $yStart+5, $black);
        }
        }
    $skiplines--;
}

fclose($fh);

// postition the x and y axis helplines

imageline($img, 219,15,219,355,$black);
imageline($img, 219,355,775,355,$black);

// apply warning visualization to graph if current value has reached warning level

if ( $r >= $warninglevel && $warninglevel != NULL ){
    $brush_size = 4;
    $brush = imagecreatetruecolor($brush_size,$brush_size);
    imagefill($brush,0,0,$brush_red);
    imagesetbrush($img,$brush);
    imagerectangle($img, 1, 1, 799, 389,IMG_COLOR_BRUSHED);
} else {
    imagefill($brush,0,0,$brush_grey);
    imagesetbrush($img,$brush);
    imagerectangle($img, 1, 1, 799, 389,IMG_COLOR_BRUSHED);
}

imagestring($img, 6, 70, $yStart-200, "Usage", $black);
imagestring($img, 4, 55, $yStart+5, "Last 30 hrs", $black);

// get formatted timestamp of file (last modified)

$datestring = "Last update: " . date("D j M G:i:s", filemtime($theFile));

imagestring($img,3,576,375,$datestring,$grey);

header(’Content-Type: image/png’);
imagepng($img);

// clear memory

imagedestroy($img);
```

```
    imagedestroy($src);
    imagedestroy($brush);
    imagedestroy($arrow);
    imagedestroy($zz);
    imagedestroy($fire);

}

drawGraph($type,$emb,$events,$evethr,$grid,$roof,$warninglevel);

?>
```

7.2 Device data collection script (Perl)

fetch.pl

```
#! /usr/bin/perl

# Script to fetch data from system by fetching system status snapshots
#
# Sources:
#
# ** Memory: Sum of the "top -b -n 1" %MEM column
# ** CPU: Sum of the "top -b -n 1" %CPU column
# ** Disk: Sum of the "df" 1K-blocks column minus
#    the sum of the "df" Availible column
#
# (Somewhat hardcoded to uncomplicate /etc/crontab usage)

use Getopt::Std;

$opt_string = 'hd';
getopts( "$opt_string", \%opt ) or usage() and exit 1;

if ( $opt{'h'} ) {
    usage();
    exit 0;
}

$DEBUG = 1 if $opt{'d'};

open(FILE1,'>>/var/www/mobil/memdata.txt') or die ("failed to open memdata.txt\n");
open(FILE2,'>>/var/www/mobil/cpudata.txt') or die ("failed to open cpudata.txt\n");;

$hourstamp = `date +%X`;
$datestamp = `date +%D`;
chomp $hourstamp;
chomp $datestamp;

# Dodge crontab influence
sleep 5;

open(CPUMEMINFO,"top -b -n 1 |");
while( $line = <CPUMEMINFO> ){
    if ( $line =~ /^\s+\d+\s+\S+\s+\S+\s+\S+\s+\d+\s+\d+\s+\d+\s+\S+\s+(\S+)\s+(\S+)/ ){

        $cpu = ($1);
        $memory = ($2);
        # debug("Memory: $memory\n");
        # debug("CPU   : $cpu\n");
        $totmem += $memory;
        $totcpu += $cpu;
    }
}
```

```

print FILE1 $datestamp . ’,’ . $hourstamp . ’,’ . $totmem . "\n";
print FILE2 $datestamp . ’,’ . $hourstamp . ’,’ . $totcpu . "\n";

debug("written to memory file: $datestamp,$hourstamp,$totmem\n");
debug("written to cpu file: $datestamp,$hourstamp,$totcpu\n");

close(FILE1);
close(FILE2);

open(FILE3,’>>/var/www/mobil/diskdata.txt’);

$totdisk = 0;
$freedisk = 0;

open(DISKINFO,"df |");
while( $line = <DISKINFO> ){
    if ($line =~ /\S+\s+(\d+)\s+\d+\s+(\d+)/ ){
        $totdisk += $1;
        $freedisk += $2;
    }
}

$diskpercentage = sprintf "%.2f",((($totdisk - $freedisk) * 100) / $totdisk);
print FILE3 $datestamp . ’,’ . $hourstamp . ’,’ . $diskpercentage . "\n";
debug("written to disk file: $datestamp,$hourstamp,$diskpercentage\n");

close(FILE3);

sub usage {
    # prints the correct use of the script
    print "Usage:\n";
    print "-h Usage\n";
    print "-d Debug\n";
}

sub debug {
    print $_[0] if ( $DEBUG )
}
```