Development of a server monitoring tool and defining performance thresholds

Focused on entitled capacity utilization for IBM POWER systems

JOHAN BYRLÉN
ITALO COTTA

BACHELOR´S THESIS

DEPARTMENT OF ELECTRICAL AND INFORMATION TECHNOLOGY
FACULTY OF ENGINEERING | LTH | LUND UNIVERSITY

Department of Electrical and Information Technology

Institutionen för Elektro- och informationsteknik

Development of a server monitoring tool and defining performance thresholds

Focused on entitled capacity utilization for IBM POWER systems

Bachelor thesis:
Johan Byrlén
Italo Cotta

Supervisor: Erik Larsson
Examinator: Christian Nyberg

LTH School of Engineering
Lund University
Box 882
SE-251 08 Helsingborg
Sweden

LTH Ingenjörshögskolan vid Campus Helsingborg
Lunds universitet
Box 882
251 08 Helsingborg

Printed in Sweden
E-husets tryckeri
Lunds universitet
Lund 2017

Abstract

Today we depend on data based services, for example in logistics, at warehouses, and accounting services. When a data based service is delayed it can depend on its servers being overloaded, leading to economic losses. To prevent this, it is necessary to have appropriate monitoring tools that can make the system administrators aware of when performance thresholds have been breached and servers are reaching a critical state.

The focus of this thesis was twofold: to define performance thresholds and to simplify monitoring of critical servers with presentation on a dashboard.

As a result of this, two thresholds were produced. A threshold for high CPU utilization which further improved the existing performance thresholds by making the required time for triggering dependent on the current CPU utilization. This proved to give us a better result in 75% of the tested cases compared to the previous model. The second produced threshold was for low CPU utilization. Through studying servers with more allocated resources than they require it was possible to inform the system administrator to disable or reallocate those resources to another server with higher demand for resources. This reduces server costs and improves performance in some cases.

The primary intention behind implementing a dashboard was to have a way of presenting when the defined thresholds had been breached. After evaluation of the dashboard through a usability test we came to the conclusion that all end-users were satisfied with the prototype and four out of five considered the dashboard to be a tool which they would use in the near future.

Keywords: AIX, IBM POWER, Dashboard, CPU utilization, Performance thresholds, Entitled capacity

Foreword and acknowledgements

We knew from the get-go that we had to be very self-propelled which was challenging because we had to go into uncharted waters regarding AIX and the structure of the POWER system. It did however make the process a lot more rewarding, by going into the unexplored we came out with far greater knowledge than we could have expected from the start.

We would like to express our sincere gratitude to **Ninni Hed** who gave us the opportunity and made this thesis possible at IKEA IT. We are also very grateful to the two supervisors **Magnus Olsson** and **Stephan Åkeborg** which she appointed to us. They contributed a lot to the thesis by giving us the tools and help we needed. An additional special thanks to **Andreas Stehn** for always being around to answer our questions and enlightening us about AIX and the POWER system.

We would like to thank our academic supervisor **Dr. Erik Larsson** at the department of Electrical and Information technology at Lund University for his constructive criticism and expertise guidance throughout this thesis.

Finally, we would like to thank IKEA IT and the very accommodating AIX team in particular for taking time off from their busy schedule to support and give us continuous feedback.

Table of contents

Terminology

AIX Team	End-users of the dashboard prototype. They consist of infrastructure designers, product specialists and system specialists.
Dashboard	Server monitoring tool. More specifically in this case, it presents the thresholds of this thesis.
EC - Entitled Capacity	Amount of CPU cores that an Lpar is allocated through the HMC.
HMC - Hardware Management Console	Entity used to control, manage and fetch data from logical partitions.
iDesk	Internal incident management system which presents incidents based on the old threshold model.
Lpar - Logical partition	An independent virtual machine.
Mockup	A design model of the prototype.
Threshold	A defined performance limit used to define when a transition from one state to another has occurred.

1 Introduction

1.1 Background

Today we depend on data based services, for example in logistics, at warehouses, and accounting services. When a data based service is not delivered it can depend on the servers being overloaded, leading to serious consequences. An example of such consequence would be customers not being able to make payments.

IKEA IT maintains and provides services that are critical for their business operations and therefore require constant server availability. To achieve this availability IKEA IT needs to have appropriate monitoring tools that present performance information from the servers on which the data based services are running. However most of these tools are targeting servers on an individual level, where it is only possible to display performance data from one server at a time. These tools provide a detailed image which is necessary for many tasks, but the downside is that it becomes more time-consuming to work with as the number of servers increase. Therefore, the need for an additional monitoring tool which provides a clear overview of all servers becomes necessary. This monitoring tool will look at servers with performance issues in particular.

To determine if a server is experiencing performance issues, it is important to have clearly defined performance thresholds. A performance threshold is a defined limit which triggers when a server goes from a normal state to what is considered a critical state which requires a system administrator to take action. During an overload it may be necessary to either add more resources to the server (e.g. give access to more CPU cores) or reallocate resources between different services running on the server. It is also equally important to keep track of servers that are not utilizing enough of their allocated resources due to economic costs.

It is desirable to have a way of presenting the breached performance thresholds. This can be achieved in various forms with text-based logs or with a graphical user interface. One type of graphical user interface is a dashboard, which is a type of control panel that presents data in different ways e.g. through statistics, charts and graphs. Using a dashboard gives the user a quick view of how all the servers behave at a certain time.

For this thesis the implementation of a dashboard was explicitly requested by the stakeholders because of the use they had planned for the prototype. It was also determined that the dashboard prototype should present an overview in a static manner e.g. non-interactive.

1.2 Purpose

The purpose of this thesis is twofold: to define thresholds related to the aforementioned elements and to improve monitoring of critical servers with presentation on a dashboard.

1.3 Goal

Many performance issues are solved reactively, meaning that the damage is already done and the system administrators must act retrospectively. The goal of this thesis is therefore to develop a solution where the system administrators can work more proactively and prevent performance issues and economical waste from occurring.

To solve the aforementioned problem, development of a dashboard prototype is required. The prototype will present information about when a server is reaching a state where performance issues or economical waste may be arising to the system administrators. For the dashboard prototype to be useful clearly defined thresholds must be defined to differentiate real problems from normal operation.

1.4 Problem specification

The problem that this thesis should solve is to 1) define thresholds that can be used to determine when an AIX system should be considered to be in a critical state. 2) Present the result from the thresholds comprehensively on a dashboard.

In order to achieve our purpose, we ask the following question:
How should the thresholds be defined and how should the result be presented on a dashboard?

1.5 Limitations

The thesis is limited as follows:

- We do not take into account how the system behaves from a security standpoint
- The developed prototype is only intended for IBM POWER systems (see 2.1.2)
- The developed prototype will only read/collect data from IKEA IT servers. It will not write or make any modifications to existing servers
- The developed prototype must be usable without any user interaction (i.e. mouse, keyboard)

2 Technical background

This chapter describes the software and hardware used during this thesis work. The reader is expected to have a basic understanding of computer architecture, software development and Unix.

2.1 Advanced Interactive eXecutive (AIX)

AIX (Advanced Interactive Executive) is a proprietary Unix operating system developed by IBM and it is aimed at enterprise and large-scale operations and only runs on the IBM POWER series.[1] The advantage of using AIX instead of other Unix operating systems is because AIX is considered more reliable when it comes to server stability. A survey of 750 organizational respondents was done in 2017 concluding that 27% of the respondents required a reliability of 99.999% or more per year, which is roughly 5.25 minutes of downtime per server/year. [2] In 2016 AIX had the highest percentage of overall uptime and availability in the Unix family. [3]

2.2 IBM POWER *(Performance Optimization With Enhanced RISC)* microprocessors

The POWER architecture family is a series of high performance microprocessors developed by IBM aimed towards servers and supercomputers.[4] The POWER systems used in the servers that this thesis will focus on are POWER7 and POWER8.

Defining thresholds in POWER systems differs from the x86 platform because of how the hypervisors (entity that creates the logical partitions and distribute its resources) are implemented. In 2013 Keene [5] described the difference between the POWER hypervisor and Intel's x86 equivalent:

"Unlike Intel hypervisors, the Power hypervisor runs as a small piece of firmware code interacting with the hardware and guest VMs running on the platform. The result is minimal overhead, at most 2%, with maximum efficiency for CPU resources, memory and virtual I/O. And because the Power hypervisor was designed in conjunction with the system architecture, it is always running, even for a single OS instance, regardless of the OS."

An important difference for this thesis is that IBM POWER allows virtual servers to utilize more than 100% of their allocated processing power due to the concept of entitled capacity (chapter 2.4), therefore a threshold suited for a x86 system is not a valid solution. [6]

2.3 Logical partition

Logical partitions (Lpars) is a division of a computer hardware resources which is separated as independent servers. Each Lpar runs its own operating system and are not limited to a specific operating system which means that one physical server can consist of several Lpars (maximum 1024 per physical server) running different operating systems.

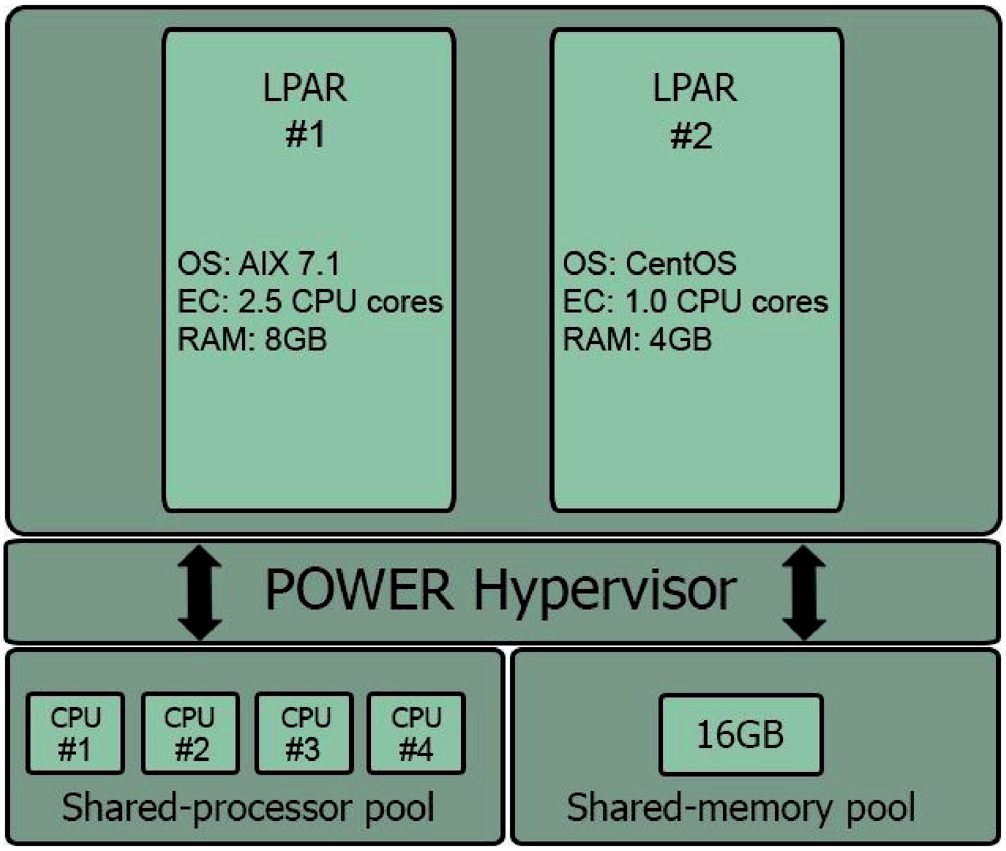


Figure 2-1: Example setup and distribution of resources on a physical server.

Figure 2-1 illustrates a physical server running multiple logical partitions. In this example Lpar1 runs a production service on AIX 7.1 and Lpar2 runs a test environment on CentOS (a Linux distribution). If the physical server itself has a total capacity of 16GB physical memory it may assign 8GB to Lpar1, 4GB to Lpar2 and keep 4GB in reserve. Of the total 4 cores, Lpar1 receives 2.5 cores, Lpar2 receives 1.0 core and the remaining 0.5 cores will be left in the shared-processor pool (more about this in 2.4). The two Lpars are completely isolated even though they are running on the same hardware. That means if one Lpar crashes, it will have no affect on the other Lpar. [7]

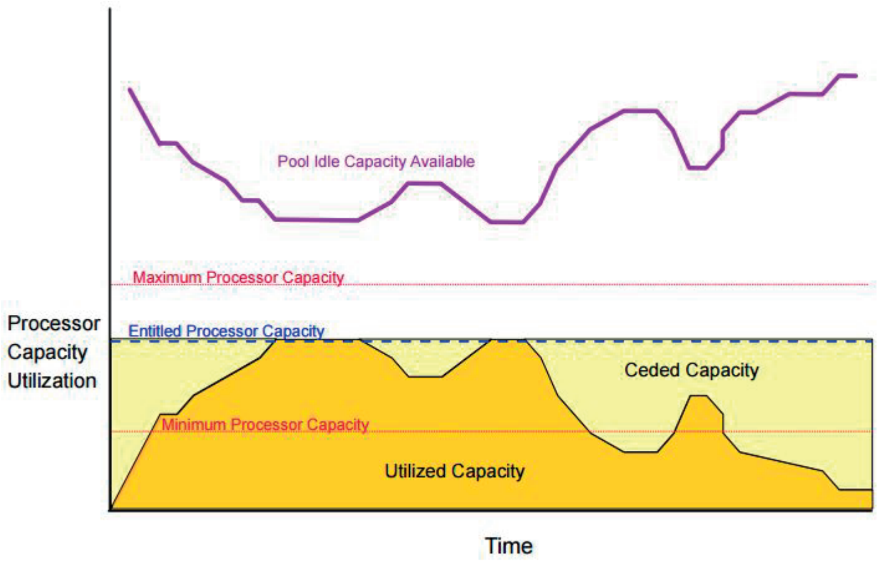
As shown in the previous example an Lpar is able to operate with fractions of physical CPU cores. Operating systems can only interpret CPU cores of an integer value, thus to give the operating system the illusion that it is using a whole CPU cores, virtual processors are used. For example, a virtual processor can take a 0.4 fraction of a physical core and split it into two virtual processor cores each using a fraction of 0.2 of a whole core.

2.4 Entitled capacity

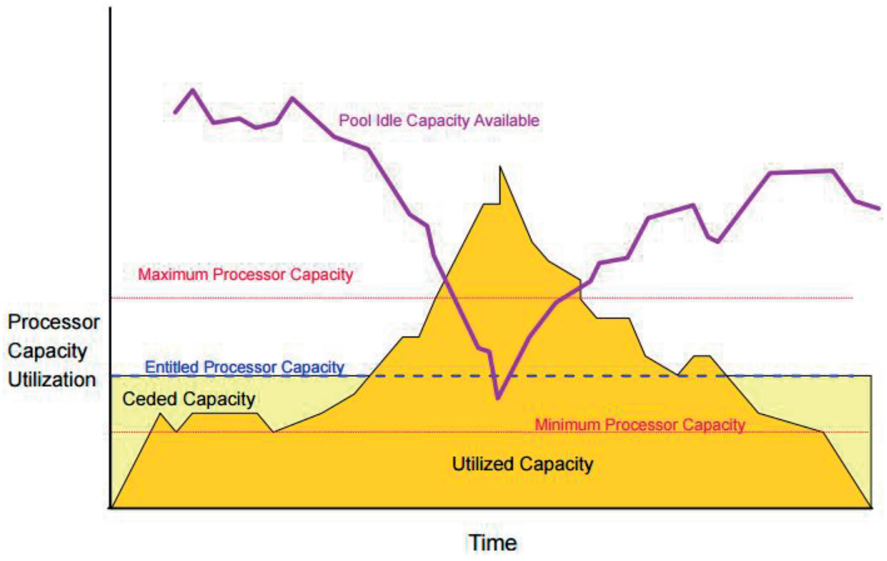
The entitled capacity is the amount of guaranteed whole or fractional number of CPU cores an Lpar has been allowed to consume from the physical server.

An Lpar which is considered **capped** will only be able to consume the CPU capacity which has been assigned explicitly as its entitled capacity, meaning it can not borrow and consume resources from the shared-processor pool. The capped Lpar can however donate its spare CPU cycles which it is not utilizing at the moment to the shared-processor pool. The Lpar does also have the highest utilization priority of those cycles which means it is allowed to take them back whenever it wants. Graph 2-2 illustrates an example of a capped Lpar. The orange area at the bottom labeled "Utilized capacity" represents the actual CPU utilization by the Lpar and because a capped configuration is being used, it cannot surpass the entitled capacity represented by a blue line.

If an Lpar requires more capacity than it has explicitly been assigned it can enter an **uncapped** mode, which lets the Lpar utilize more CPU cores than it was originally entitled to. An uncapped Lpar borrows CPU capacity from a shared-processor pool where it can take spare cycles from the physical server which has not been allocated to other partitions. When the utilized capacity surpasses the entitled capacity in graph 2-3 the remaining shared-processor pool is reduced (purple line). [8][9]



Graph 2-2: Capped Lpar example



Graph 2-3: Uncapped Lpar example

There are performance drawbacks to the uncapped mode when borrowing resources from the shared-processor pool compared to when only using the entitled capacity. An Lpar with a capped mode enabled will always receive processor cycles from the same area on the CPU. While in an uncapped mode the partition will take unused cycles from different CPU areas which leads to longer access times. There is also a waiting queue to the shared-processor pool based on a number of factors, one of which is partition priority.

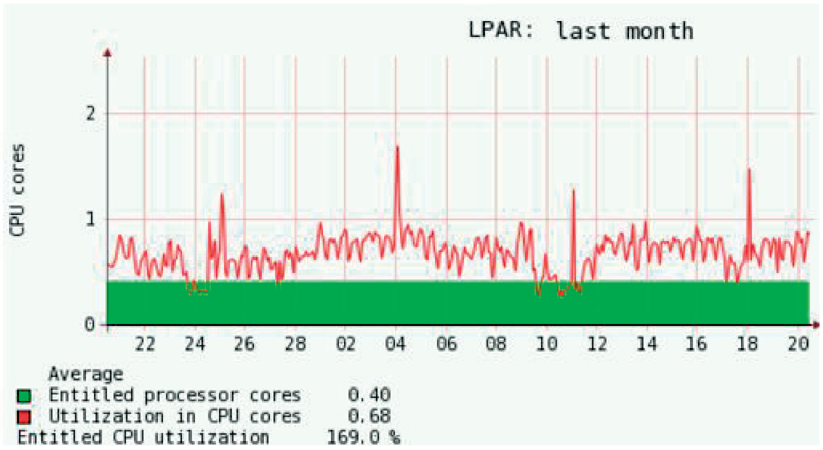
Partitions have different weight (priorities) based on the services that are running. Weight of a partition is represented by an integer (default 128) ranging between 0 and 255 where higher weight equals a higher priority. For example a company's production server may have a weight of 255, while the pre-production testing server has the default 128 weight. This means that the production server will be given roughly 65% of the shared-processor pool while the pre-production testing server receives 35% (255/(255+128)*100). [10]

2.5 Monitoring and tracking

This section describes two of the existing tools that are used for tracking and displaying server status information that was used for data gathering.

Hardware Management Console (HMC)
The HMC is a single-purpose closed appliance developed by IBM and is used to manage POWER servers and their Lpars.[11] It can be accessed both locally and remotely via SSH to gather performance data such as current entitled capacity, how many cores there are left in the shared-processor pool, current used memory and server availability.

Lpar2RRD
Lpar2RRD is an open source server monitoring tool used to display statistical data in relation to CPU utilization and memory for servers and their Lpars. It fetches data directly from the HMC.



```
   Average
🟩 Entitled processor cores      0.40
🟥 Utilization in CPU cores      0.68
Entitled CPU utilization       169.0 %
```

Graph 2-4: Lpar2RRD example screenshot from public Live Demo [12]

This tool is useful to visualize how much entitled capacity Lpar has, what the CPU utilization looked like within certain time periods. An example of this is shown in graph 2-4. It displays the CPU utilization for an Lpar during the last month. In this example, the Lpar has 0.4 entitled capacity (shown as the green rectangle). Anytime the CPU utilization (red graph) goes over the entitled capacity it means that it is borrowing processing power from the shared-processor pool.

3 Related work

This chapter describes the related work of performance thresholds and server monitoring tools.

3.1 Thresholds

There is currently a lack of scientific basis regarding how to determine performance thresholds on servers, especially in the POWER series. A common practice today is to set a threshold based on experience in relation to previous incidents and what is considered to be normal utilization.

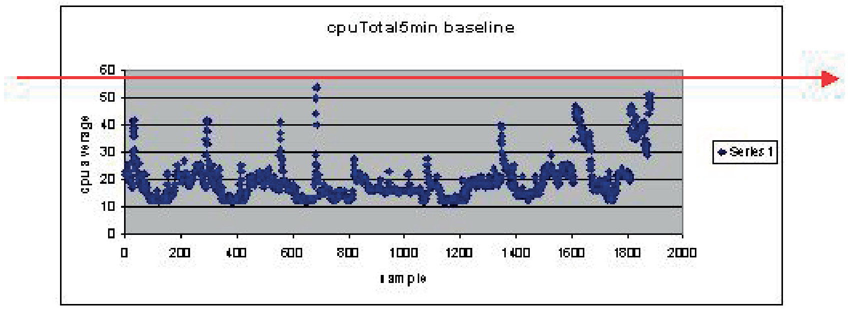
Cisco makes an attempt at defining performance thresholds with a scientific basis. Their paper [13] focuses mainly on network analysis, but they also discuss a strategy to set thresholds based on CPU utilization which they call the "Ready, Set, Go threshold methodology". The methodology uses three thresholds in succession:

- Ready - used to predict which servers might need to be maintained in the future
- Set - used as an indicator, and alerts you to start planning for a repair, reconfiguration or upgrade
- Go - considered a fault condition and requires some action to repair it

It should be noted that Cisco's paper is mostly focused on their own proprietary hardware, and what they mention will not fully apply to POWER servers. However, their methodology for determining thresholds is of interest because implementing three different urgency levels is something that could be used to further expand the implemented dashboard prototype in the future. For example different threshold levels could be represented with different colors in the dashboard.

This methodology is however not appropriate to follow in this thesis due to its primarily focus is on showing whether a system is in a stable or in a critical state (i.e. one threshold per element).

When it comes to defining actual thresholds they look at multiple samples over a certain period of time (graph 3-1).



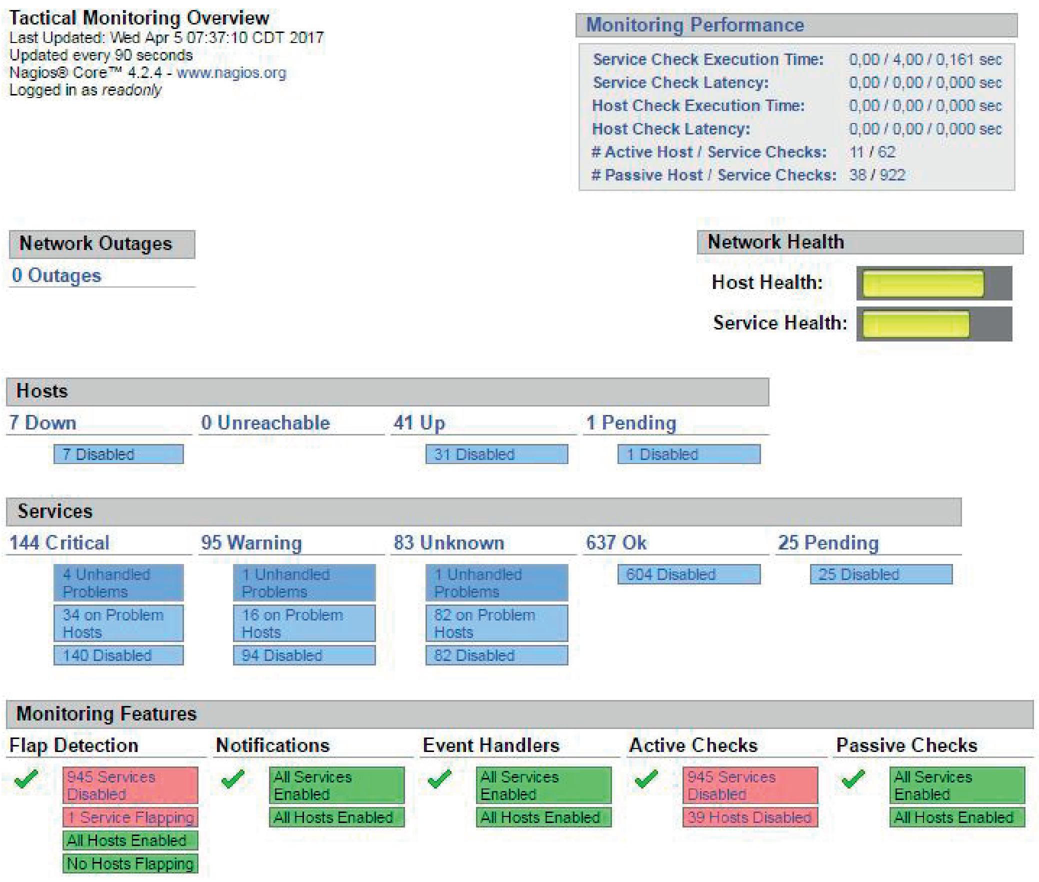
Graph 3-1: Cisco's example of CPU averages over 5 min [13]

Then based on the highest CPU averages during normal usage a baseline is determined (~58% in this case).
This way of defining thresholds does not suit the purposes of this thesis mainly because in a POWER server with multiple Lpars an Lpar can easily spike up for example 500% CPU utilization under a short time (borrowing from the shared-processor pool) when a batch job is being executed, meaning that high high CPU utilization under short periods of time does not necessarily mean that the Lpar is in a critical state.

3.2 Dashboard

There are several already made server monitoring dashboard solutions. Nagios [14] was chosen to be analyzed because it is one of the most popular server monitoring tools with AIX compatibility [15].

Nagios provides a very detail-rich image (figure 3-2) of the servers it is monitoring. However, the interface is not very intuitive and does not instantly display what the issues are with the servers.


Tactical Monitoring Overview
Last Updated: Wed Apr 5 07:37:10 CDT 2017
Updated every 90 seconds
Nagios® Core™ 4.2.4 - www.nagios.org
Logged in as *readonly*

Monitoring Performance

Service Check Execution Time:	0,00 / 4,00 / 0,161 sec
Service Check Latency:	0,00 / 0,00 / 0,000 sec
Host Check Execution Time:	0,00 / 0,00 / 0,000 sec
Host Check Latency:	0,00 / 0,00 / 0,000 sec
# Active Host / Service Checks:	11 / 62
# Passive Host / Service Checks:	38 / 922

Network Outages
0 Outages

Network Health
Host Health:
Service Health:

Hosts

7 Down	0 Unreachable	41 Up	1 Pending
7 Disabled		31 Disabled	1 Disabled

Services

144 Critical	95 Warning	83 Unknown	637 Ok	25 Pending
4 Unhandled Problems	1 Unhandled Problems	1 Unhandled Problems	604 Disabled	25 Disabled
34 on Problem Hosts	16 on Problem Hosts	82 on Problem Hosts		
140 Disabled	94 Disabled	82 Disabled		

Monitoring Features

Flap Detection	Notifications	Event Handlers	Active Checks	Passive Checks
✓ 945 Services Disabled	✓ All Services Enabled	✓ All Services Enabled	✓ 945 Services Disabled	✓ All Services Enabled
1 Service Flapping	All Hosts Enabled	All Hosts Enabled	39 Hosts Disabled	All Hosts Enabled
All Hosts Enabled				
No Hosts Flapping				

Figure 3-2: Nagios screenshot from live demo [http://nagioscore.demos.nagios.com/]

Because Nagios is open-source it relies heavily on plugins, there is a possibility to use existing plugins or to write your own. The plugin *check_Lparstat* [16] which in theory should add support for tracking Lpar entitled capacity (chapter 2.4) is broken and correcting/implementing a similar plugin would have taken a lot of extra time to create because of the in-depth knowledge into the monitoring service it would require. In addition to having strip away unnecessary features to avoid having to maintain a bloated interface with unused resource-heavy services running in the background, as well as customizing it to a non-interactive summary dashboard.

After evaluating Nagios it became clear that this solution would not fit the needs of this thesis work since it does not allow the user to see an overview of all the servers and the related problems to the servers without having to manually navigate into menus, this lead to the conclusion of creating a tailor-made and non-interactive dashboard from scratch.

4 Method

This chapter describes the methods which were used to define performance thresholds and also how the design of the dashboard was produced. *Threshold definition* (4.1) includes the analysis which was done to determine the proposed thresholds. *Dashboard design* (4.2) proposes three different mockup designs for the prototype.

4.1 Thresholds definition

The following section describes the data gathering process consisting of interviews with the end-users and an *evaluation of previous incidents* (4.1.2) with was was used to figure out the underlying problem related to thresholds that this thesis was meant to solve. *Proposed thresholds* (4.1.3) proposes two different performance thresholds in relation to CPU utilization.

4.1.1 Data gathering

To get a better understanding of what the AIX team at IKEA IT actually wanted to get out of this thesis work five informal conversational interviews with members of the team were performed in the hope of being able to define the project scope and priorities. The AIX team consisted of infrastructure designers, product and system specialists.

This lead to prioritizing starting out with something basic - tracking which servers were currently online and offline, as this would have made it easier to get a better idea of how the currently used tools to fetch data from the servers worked.

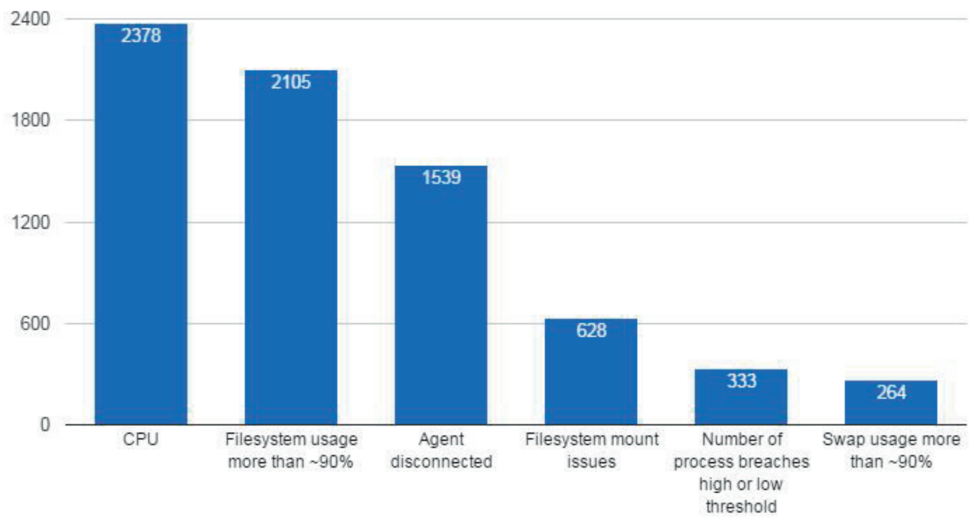
It was also suggested that focus should be put on being able to track when the CPU utilization had been over the entitled capacity for extended time periods since currently the only way to track this was to use the performance monitor tool Lpar2RRD and individually go into individually each Lpars overview page and check the graphs generated (see graph 2-4 for example).

Ideally focusing on tracking multiple different performance data metrics would have been desirable but because dealing with limited resources prioritizing the most impactful ones was important.

4.1.2 Evaluating previous incidents

To get a better idea behind the most common incidents related to the POWER servers an evaluation of previous incidents was made in the hopes of finding out what thresholds should be focused and prioritized.
These incidents are created either manually by a system administrator or automatically by a system process whenever unusual activity is detected.

Analyzing the collected data from the past 12 months gave the results shown in graph 4-1.

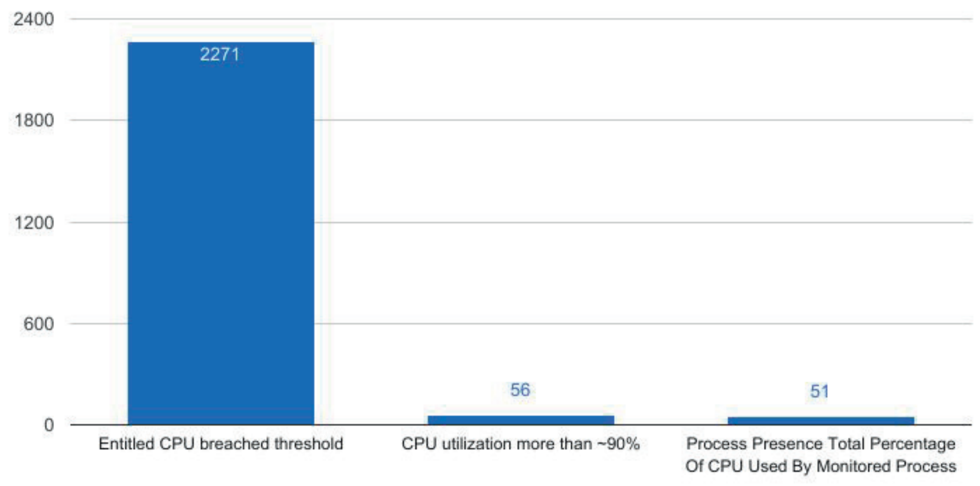


Graph 4-1: Analysis of most common incidents from april 2016 - april 2017

Since CPU related incidents were the most common type of incident during the past 12 months it was decided to particularly focus on this type of incidents.

It should be noted that other incidents (e.g. related to the filesystems) are also relevant but were chosen to be disregarded due to lack of resources.

When analysing all incidents from the past 12 months containing "CPU" as keyword, see graph 4-2, breaches of entitled CPU utilization were the most usual type of CPU related incident and therefore it was chosen as the main focus of this thesis work.



Graph 4-2: Analyzing CPU related incidents

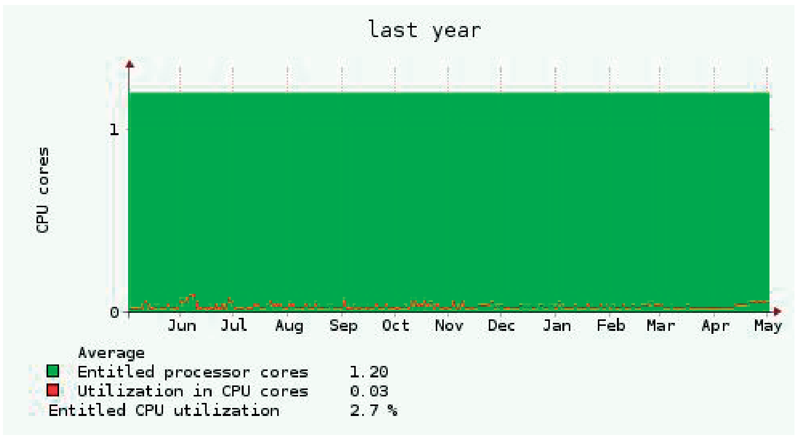
4.1.3 Proposed thresholds

This section proposes two different threshold models, one for high utilization and one for low utilization. The methods are based on the aforementioned data gathering and evaluation.

4.1.3.1 Low utilization

When an Lpar has been allocated more entitled capacity than it requires it will result in underutilized resources, this contradicts the whole purpose of virtualization which is to optimize the utilization of resources. Therefore it was decided to have a threshold that would warn whenever an Lpar has gone underutilized for a long period.[17]

Making an exact determination of what a "long period" is in this case is not feasible. There are many variables and outside factors that play part. In some cases seven days would be good, other times three months would be recommended. This depends on what the Lpar is supposed to be used for, which applications are going to be used on top of the Lpar and what business model it uses. These are inaccessible factors which cannot be analyzed throughout this thesis, therefore a value of 30 days was determined through general performance analysis of available Lpars in the monitoring tool Lpar2RRD and from expert consulting from the AIX team.



Graph 4-3: CPU Utilization for an Lpar for the period May 2016 - May 2017 (from Lpar2RRD)

Graph 4-3 gives an example of an Lpar with entitled capacity of 1.2 cores yet the Lpar only used about 0.03 cores in average. If the entitled capacity had been lowered to a more appropriate value in consideration to the average CPU utilization after the first month then a considerable amount of unused assigned cores would have been turned off or assigned to other Lpars running more critical services, thus providing economic benefits.

4.1.3.2 High utilization

This threshold is breached when the current utilization is higher than the proposed utilization value (see table 4-6) and the time required for triggering has been reached. Breaching this threshold is a clear indication that the Lpar is being overloaded. The reason behind this could for example be malfunctioning of services (i.e. bugs causing unexpected behaviour) or that the entitled capacity has been wrongly configured.

It should be noted that prolonged utilization of the shared-processor pool is not desirable because eventually if all Lpars are temporarily borrowing resources from the shared-processor pool the CPU will run out of resources to distribute, it is however acceptable for the shared-processor pool to be used for short amounts of time during utilization peaks.

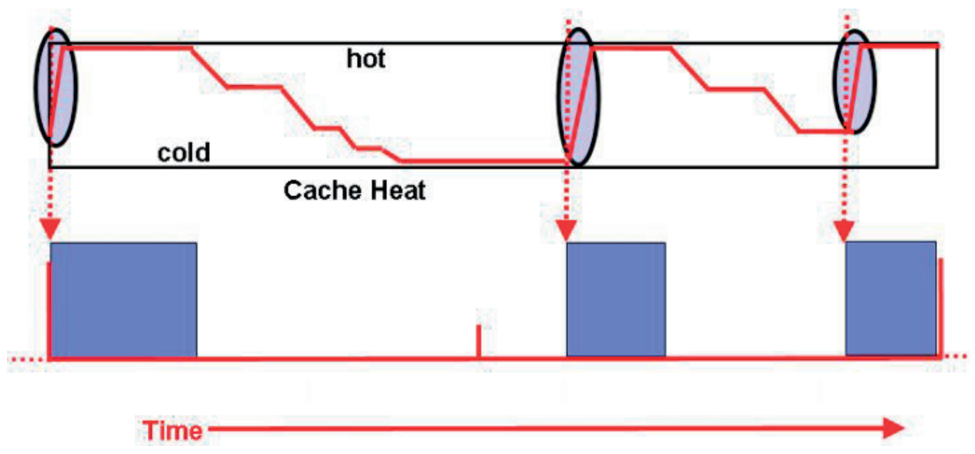


Figure 4-4: Example of cache reload causing time loss

Since the Lpars time-share the physical processor of the machine, there are disadvantages in the way the cache memory in the CPU is handled when an uncapped configuration is being used. Lpars in a machine cannot access parts of cache memory that have been recently written to by other Lpars and whenever an Lpar gets interrupted by a different Lpar its CPU cache gets overwritten meaning that whenever it is able to execute again it has to fetch its required data to the cache once again (see figure 4-4).[18] Whenever the current Lpar (blue rectangle) utilizes more than its entitled capacity, it has to take resources from the shared-processor pool. This is when it can get interrupted by other Lpars with a higher weight value. The longer an Lpar gets interrupted for, the more contents of the cache will become overwritten over time. If the delay is long enough such that the original cache gets mostly overwritten (cold), then whenever the current Lpar is allowed to run again it will have to fetch whatever data is missing to the cache once again. This time penalty is presented with blue ellipse in the figure.

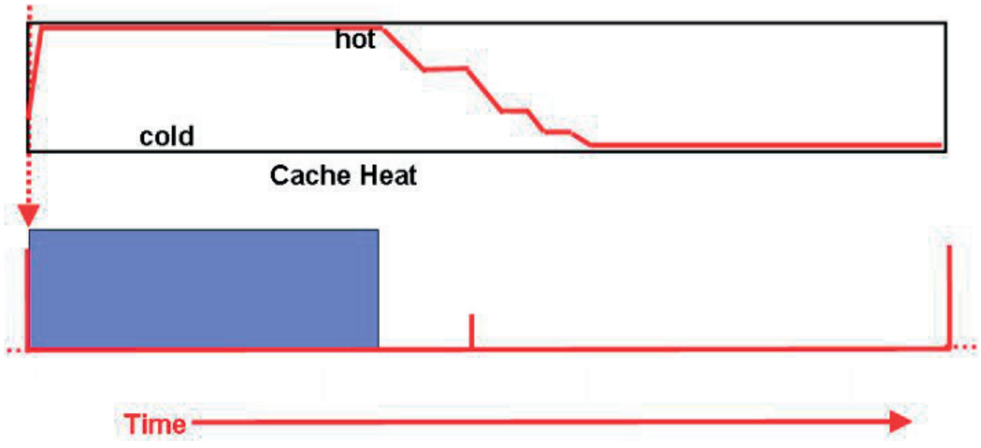


Figure 4-5: Example of cache reload solution

If the current Lpar instead has enough entitled capacity to ensure it will be able to run without having to access the shared-processor pool then it will not get interrupted by other Lpars thus reducing the amount of times the cache has to be fetched. An example can be seen in figure 4-5, the current Lpar in this case has a higher entitled capacity and can therefore not be interrupted by other Lpars thus reducing the time penalty.

The are occurrences where breaching a threshold may be legitimate. One occurrence would be during peak hours, when the Lpar is undergoing heavy workload during the day. An example of such peak hours would be the online hours of an economy service for the IKEA store within EU. A service as such will only be used during office hours, when there are people accessing the stores. This would lead to higher CPU utilization during that time.

To define the thresholds that were used, a starting point of CPU utilization of over 400% after 8 hours was chosen. This static value has been tweaked and established after many years of practical usage by the AIX team on the accessible Lpars used in this thesis, therefore it was chosen as the starting point. What was done to further improve this threshold was to make the required time for triggering dependent on the current utilization.

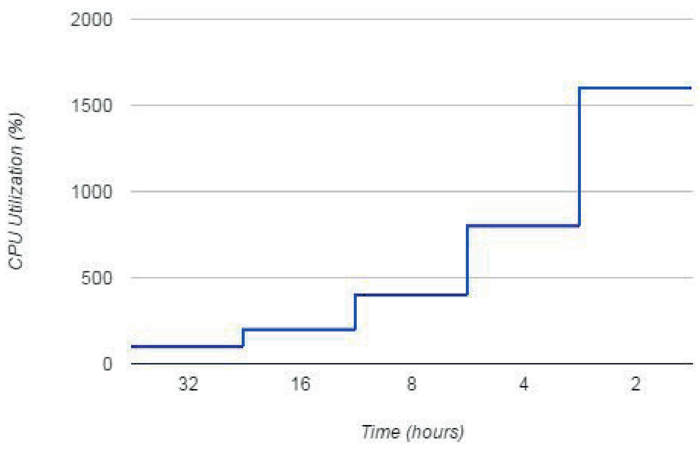
Required time to trigger (hours)	Utilization threshold (%)
32	100
16	200
8	400
4	800
2	1600

Table 4-6: The chosen threshold values

As seen in table 4-6 and graph 4-8, whenever the starting threshold for utilization is doubled the required time for triggering is halved (see formula 4-7) meaning that the utilization is proportionate to the inverse of *Time*. Where *Time* is the amount of hours required before triggering the threshold.

$$\boxed{Utilization \propto \frac{1}{Time}}$$

Formula 4-7: The CPU utilization is proportionate to the inverse of Time required for triggering



Graph 4-8: Threshold plateaus

Actions are required from a system administrator if an Lpar's CPU utilization has breached one of the thresholds in this case. An action to take might be starting an investigation of the individual Lpar. It is important to see how this particular Lpar correlates to the host machine and its total CPU pool. If an error is found during the investigation necessary steps has to be made regarding workload management e.g. resource reallocation or reconfiguring of entitlement. As a last resort the least critical Lpar has to be stopped or migrated to another host machine.

4.2 Dashboard design

This section describes the different mockup designs which were created before developing the dashboard. Mockups were carried out to detect design problems early on. These mockups were used to figure out what what the end user would have liked to see in the final prototype.

To develop mockups early in the process is a way to avoid extensive changes of the layout after the implementation. This removes the risk of having to remove or deprecate already implemented functions. [16]

Mockup 1

The first mockup (see appendix I) was a simplistic concept of a dashboard. It was divided into three different pages that the user would have to navigate through. The first page shows all the data centers in the form of a table. The data centers are colored in either green or red depending on if they are in a critical state (red) or running in normal condition (green). The amount of critical servers out of total amount available are also noted on this page.

The second page is reached by navigation with a click on a data center. It then shows the amount of servers (Lpars) that are critical with a number in the top right corner, and also with red and green circles, each circle representing one Lpar and its state.

The third and last page is reached by clicking on one of the circles in the previous page. This last page presents status information of the Lpar. Status information regarding the servers condition, and an explanation as to why it is marked red (critical condition) if that is the case.

We discarded this mockup because it required user-interaction to display any useful information to the user. It also only showed the Lpars and not the physical servers. The good thing with only showing Lpars was that it required less navigation (clicks). However one page (second page) could only show 28 Lpars with this mockup — which is not a very good overview image, considering one data center can contain hundreds of Lpars.

Mockup 2

The second mockup (see appendix II) gives a more modern impression. The first page shows a world map with green and red circles displaying the data centers spread across the world and their current state. Clicking on one of the circles representing a data center takes you to the second page of this mockup. The left of this page shows the country of where the data center is located at, and with a circle indicating its geographical position. The right side represents the available Lpars in that particular data center and its condition.

Clicking on one of those Lpars on the second page would take you to a third page with more detailed information regarding the Lpars state, but this mockup was discarded before reaching that stage. The second mockup tried to give a more modern look and feel by using a world map. It was however not particularly useful due to the fact that it also required user-interaction and could only visualize a few Lpars per data center on the same page. There was also drawbacks to the first page, due to it being geographical, if one smaller country would have two data centers with relative close proximity to each others, it would be hard to make out the data centers.

Mockup 3

The third mockup (see appendix III) is a static one-page dashboard which does not require any user-interaction unlike the other two mockups. It is split into three panels, the top-left panel list the current active server problems with description regarding the data center, machine name, Lpar and the reason for triggering a threshold. The bottom-left panel gives overview status data of all the data centers, colored in either green - normal condition - or red if the data center contains one or more critical servers. The right panel displays a view of all the available Lpars. The largest circle represent a data center, the middle-sized circle represent one physical server and the colored circles represent one Lpar. Green or red based on the server's state, the red circle is also slightly larger to get noticed easier. If a circle is red, it will also be displayed in the top-left panel under "Server problems".

The third mockup is a combination of the other two, it includes both an overview table and a circle-representation of the Lpars. It does not require any user-interaction to receive the necessary data and can therefore be displayed on an always-on monitor without the need for user input. It does however have problems related to scalability, more data centers and Lpars means the circles will have to get smaller. It may cause issues when the dashboard has to display more than a thousand Lpars.

5 Result

This chapter presents the results of evaluating the defined thresholds and the implemented dashboard.

5.1 Threshold evaluation

This section evaluates the thresholds proposed in chapter 4 for low respectively high utilization.

5.1.1 Low utilization

For this evaluation four different Lpars running on the same POWER server were chosen. The evaluation consisted of looking at the CPU utilization graphs in Lpar2RRD, as shown in appendix VI, using our proposed threshold for low utilization.

The evaluation found that in the cases of Lpar 1 through 3 the low utilization threshold would be triggered after the first 30 days. The threshold would then create an incident that would be presented on the dashboard. If we assume that the system administrator lowers the EC of the Lpar after noticing it is not being used enough, the entitled capacity could for example then have been lowered by 0.3 per Lpar for 335 days.

The POWER server used during the evaluation has an approximate cost of 200 SEK per core/day. This entails a total saving of (3* 0.3 * 200 * 335) 60300 SEK if the entitled capacity had been lowered after the first 30 days.

The opposite is observed in the 4th Lpar. It had a higher utilization than its entitled capacity during a long interval. If our threshold for high utilization had been used instead, the system administrator would have been able to quickly notice this increase in utilization and appropriately adjust the entitled capacity for this Lpar. After consulting with a system administrator it was recommended that this Lpar should have had its entitled capacity increased by 0.4 cores. It would thus result in a cost of (0.4 * 200 * 365) 29200 SEK over the past year.

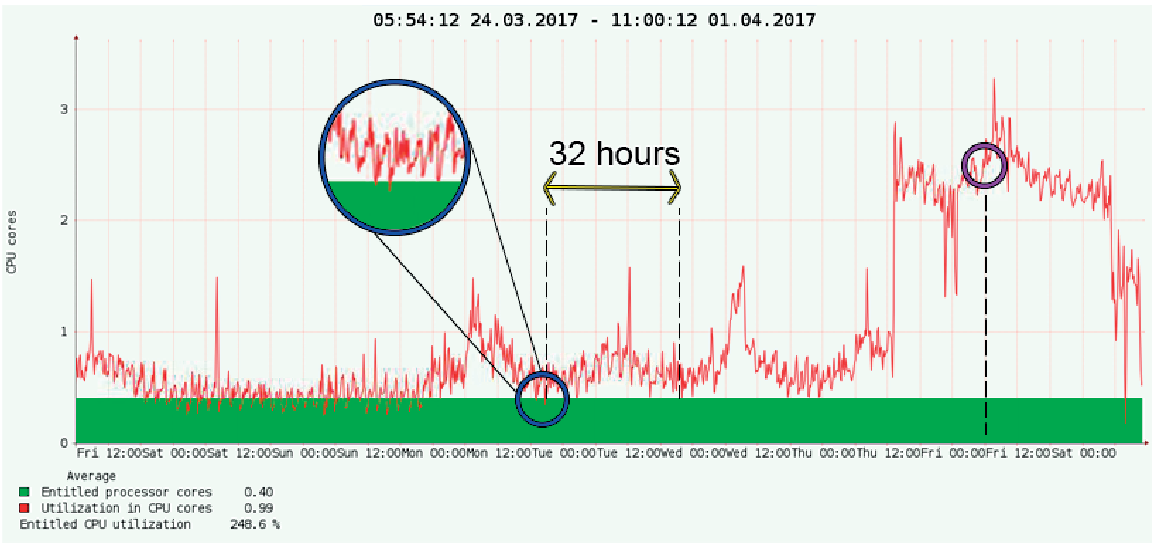
Using our example we could disable 0.9 cores (Lpar 1 to 3) and allocate 0.4 cores (Lpar 4). This results in a saving of (60300 - 29200) 31100 SEK over a span of 365 days on just one POWER server. As well as making Lpar 4 run faster due to a higher entitlement (less borrowing from the shared-processor pool).

5.1.2 High utilization

Three different *Entitled Capacity* related incidents were randomly chosen from the internal incident management system iDesk between 2016-04-01 and 2017-04-01 to evaluate this thesis' high utilization thresholds. The incidents chosen are seen in table 5-1. The column *Entitled CPU* denotes the current CPU utilization when the incident was triggered. *Threshold set at* is the threshold set for the particular Lpar by the earlier model, and *Triggered at* is the time when the incident was triggered.

Graph	Entitled CPU	Threshold set at	Triggered at
Graph 5-2	548%	400%	2017-03-30 11:36 PM
Graph 5-3	430%	400%	2017-02-23 03:16 AM
Graph 5-4	996%	400%	2017-03-02 05:59 PM

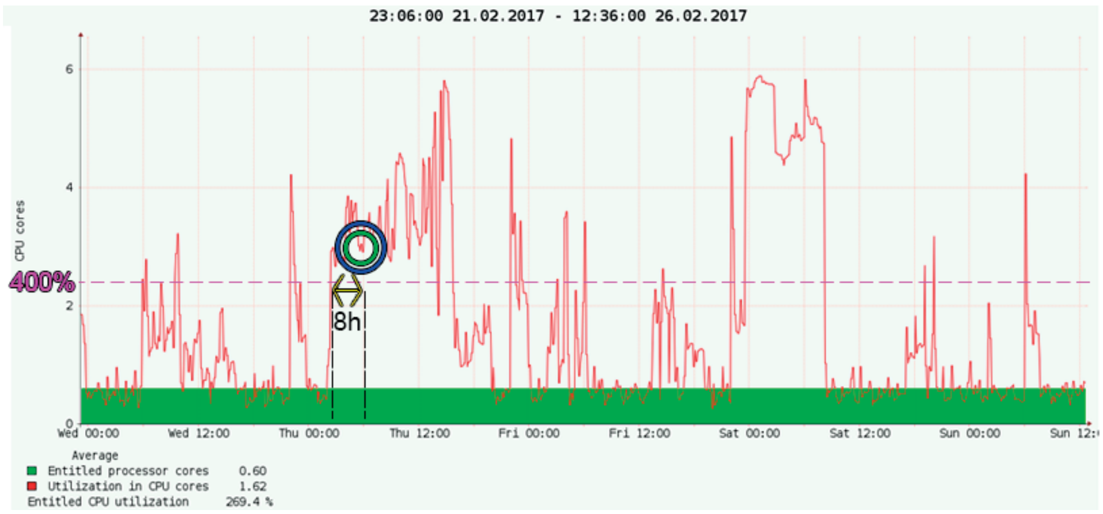
Table 5-1: Three randomly chosen incidents



```
        Average
🟩 Entitled processor cores     0.40
🟥 Utilization in CPU cores     0.99
Entitled CPU utilization     248.6 %
```

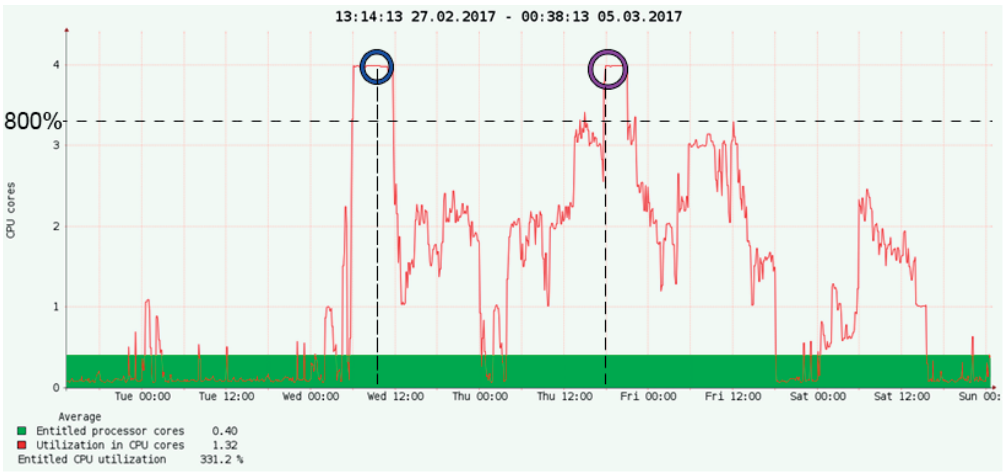
Graph 5-2: Analysis of an Lpar from Lpar2RRD which caused an incident and comparing this thesis' thresholds with the previous threshold model

Graph 5-2 illustrates the first randomly chosen incident. The purple circle around 00:00 on Friday represents the triggered incident from iDesk, this is when the system administrators were notified about the issue. However as presented in the graph, there is instability occurring before the iDesk incident was triggered. As seen in the blue circle, 12.00 AM on Tuesday is the last time the Lpar has less than 100% CPU utilization. This is when the dashboard will begin monitoring the Lpar, when the defined threshold (>100% during 32 hours) is breached. An incident is triggered at around 03:00 PM on Wednesday. This means that when using the threshold defined in this thesis, the dashboard would notify the system administrator after noticing the first signs of instability. The Lpar instability could then be analyzed in real-time and have the larger spike occurring at 06.00 AM minimized or altogether stopped from happening.



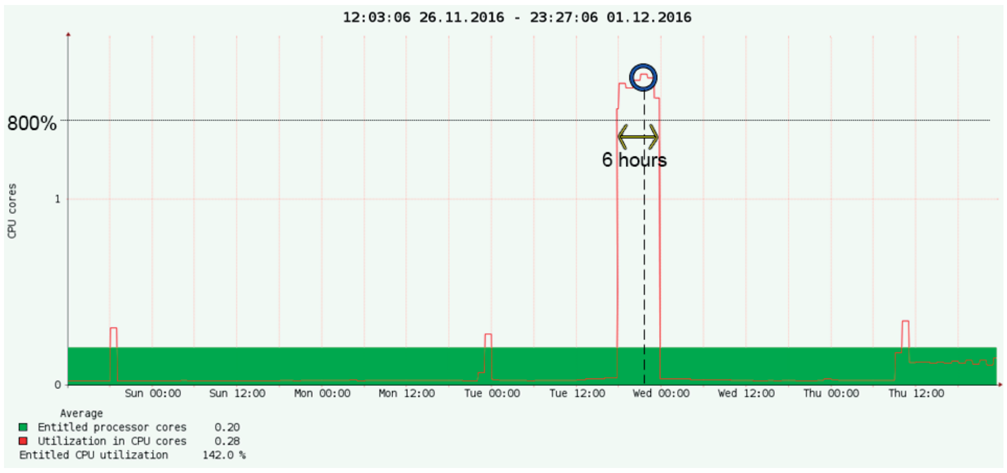
Graph 5-3: Analysis of an Lpar from Lpar2RRD which caused an incident and comparing this thesis' thresholds with the previous threshold model

Graph 5-3 illustrates the second randomly chosen incident. The incident is generated on Thursday at 03:16 AM. In this case both our defined threshold (marked with purple circle) and the generated threshold (marked with a purple circle) would simultaneously trigger after surpassing 400% utilization after 8 hours.



Graph 5-4: Analysis of an Lpar from Lpar2RRD which caused an incident and comparing this thesis' thresholds with the previous threshold model

As seen in graph 5-4 the incident is generated on Thursday at 05:59 PM (marked with purple circle), in this case there is a similar spike the day before but this did not trigger an incident because in iDesk an incident is generated when the average utilization for the past eight hours has been over 400%. Since a set of multiple threshold intervals were defined during this thesis, the threshold model will also look at above 800% CPU utilization during four hours which is what occurs in the first spike. This results in triggering only four hours after the instability and spike occurs. The thresholds of this thesis would therefore trigger at (as seen in the blue circle) around Wednesday 10.00 AM instead of what was actually reported through iDesk (purple circle, at Thursday 05:59 PM). The system administrators would thus be notified one and a half day in advance when using the thresholds of this thesis.



Graph 5-5: A false positive where this thesis' threshold would trigger

Graph 5-5 shows an instance where the one of the thresholds of this thesis would be triggered. It was not reported as an incident in iDesk because the major spike in the graph did not last for more than eight hours. The spike lasted for six hours and was above 800% throughout this period. This means that this thesis' threshold for above 800% utilization during four hours would be triggered. This is however a false positive as it is only a spike which corrects itself after six hours. In this instant, a system administrator would be notified without there being a long-term issue.

5.2 Dashboard evaluation

This section presents the results of a usability test performed to consolidate how satisfied the AIX team were with the graphical interface.

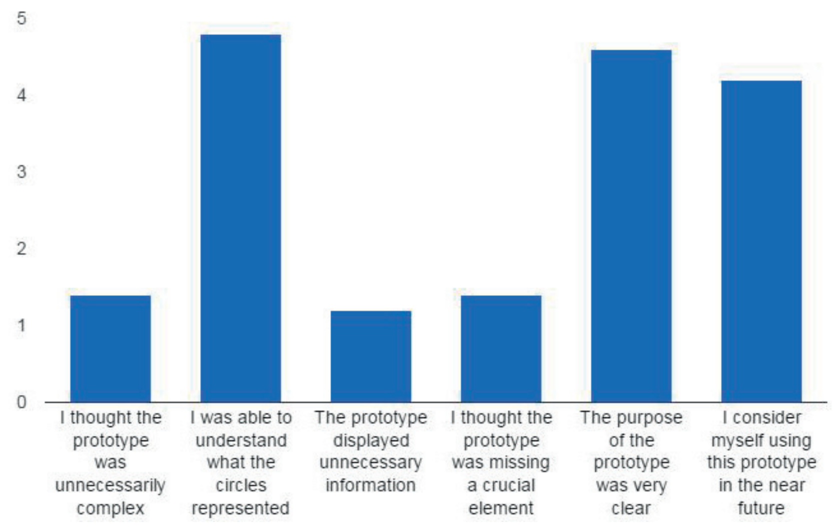
5.2.1 Usability test

Usability tests are a form of quantitative research that is used for detecting evident problems with a system. It is not a type of test that can be used for comparing two systems but it will make evident any obvious flaws with the dashboard prototype.

The questionnaire seen in *appendix V* was sent to the AIX team and used as the base for the usability test.

The results of the usability test performed with the members of the AIX team is illustrated in graph 5-1. A value of 0 in the y-axis represents **strongly disagree** while a value of 5 in the y-axis represents **strongly agree**.

The results shown in the graph are the average responses out of the four members of the AIX team who answered the questionnaire.



Graph 5-1: Presentation of the AIX teams answer to the usability questionnaire where 0 indicates "Strongly disagree" and 5 indicates "Strongly agree"

Improvement proposals and feedback by the AIX team:

- Be able to mark both physical servers and Lpars as "in maintenance" when there is work in progress. If an Lpar is being shut down purposely it should not be displayed as "Offline"
- Implement a timestamp to see when an offline Lpar was last online

5.2.2 Dashboard prototype

It was decided to make an implementation based on mockup 3. The result of the finished dashboard prototype can be seen in appendix VI.

6 Conclusion

This chapter concludes the thesis with our findings. It discusses the process of the thesis and the evaluation that was previously made. An important ethical factor with the dashboard is confidentiality and the need for keeping it secure, this is also discussed in this chapter under *ethical aspects.*

6.1 Threshold

This thesis has proposed two different threshold factors: High utilization and low utilization. High utilization is a common threshold factor looked at by most monitoring tools today and also it was the most common type of occurring incident in the analysis from chapter 4.12. Which is why it was chosen to monitor high utilization, however we expanded on the idea and made the reporting more accurate by making the required time for triggering dependent on the current utilization.

The low utilization model proposes a threshold of 30 continuous days where the CPU utilization has not reached 100%. A period of 30 days was determined through general performance analysis of available Lpars in the monitoring tool Lpar2RRD and from consulting with the AIX team.

As demonstrated in the evaluation of our high utilization threshold (chapter 5.1.2), critical incidents can be reported earlier than the previous system. This occurred in two out of four randomly chosen cases, the third case was equal to the previous model and the fourth caused a false positive, thus *in 75% of the randomly tested cases our method provided an equal or better result.*

By monitoring both under and overutilized Lpars we manage to pick up on where we could remove resources and where those resources should be reallocated for better performance. As proven by Griffith[18] there is a possibility to increase the performance by as much as 60% in CPU utilization in Lpars which has a low entitled capacity and thus borrowing most of its resources from the shared-processor pool. By informing the system administrator we can start to balance the resource load between Lpars and thus *increase performance and save on total cost with our solution.*

As shown in graph 5-5 our method may lead to false positives in some cases. It is not known how likely it is for this this type of event to occur. It should however be noted that when a CPU utilization spike as the one shown in the graph occurs it will only be displayed in the dashboard for a short period of time. This is because as soon as it goes below its requirements for triggering, it will be removed from the dashboard.

6.2 Dashboard

Through a usability test we determined that the dashboard was appreciated by the end-users. All of the users who answered the questionnaire were considered to be satisfied with the prototype. Four out of five considered the dashboard to be a tool which they would use in the near future.

6.3 Ethical aspect

Some of the data which is handled within the developed prototype is considered confidential. The prototype stores data such as login credentials to servers of which are crucial to the operations of IKEA IT. The dashboard does also function as a form of blueprint by visualizing the structure of the private business network. It is therefore important store the login credentials securely and not give anyone access to the dashboard outside the private network.

Confidentiality is important because of the outside threat to the private network. Today attacks are a regular and continuous occurrence. If an attacker breaches the secure network and is given access to the dashboard, we do not want the attacker to collect more information regarding the login credentials.

7 Future work

This section discusses the next steps which we recommend taking to make the thresholds more accurate and the dashboard more useful. Throughout the process of developing we have come to learn what features of the dashboard are good and which are bad. Given more time and resources, the dashboard could be greatly improved.

7.1 Thresholds

To expand on this solution we recommend a step towards making the thresholds more dynamic and thus responsive to a changing environment. There are additional factors which affect the stability of servers worth looking into.

The most common factors looked at today are CPU utilization over a set time, meaning that a server will be considered in a critical state if the usage has been above X% for Y minutes. This approach is the practise most used today. It works relatively well and is what we expanded on. However there are numerous other factors which are overlooked for simplicity that play part in the state of a server.

Shared-processor pool resources and availability
When a Lpar exceeds the CPU usage threshold for a longer time-period it may be an indication that there is a problem with the Lpar. It may also pose no immediate danger. Therefore it would not be appropriate to flag the Lpar as critical and as a reason to call a system administrator to look at it as long as there is more resources available to the Lpar in the shared-processor pool. If there are unused resources available in the shared-processor pool for the Lpar to use, then the time before alerting should be prolonged.This would be economically beneficial as it gives the Lpar additional time to correct itself, and the Lpar will not crash due to lack of resources.

It does also go both ways, if the Lpar only uses a fraction of its entitled capacity but the physical server's shared-processor pool is low on resources it must still flag as the machine being in a critical state and a system administrator must take action. If it is not solved, the Lpar may not get the appropriate resources it requires to continue on its activity when required - this may lead to crashes or serious delay. Therefore it is recommended to have a threshold that takes the remaining shared-pool capacity in consideration when determining the state of an Lpar.

Priority
When there is a problem with several Lpar at once, the current weight of the Lpars must be taken into consideration. Every Lpar with a problem should be flagged as critical, but the ones with the largest weight (highest priority) should be marked accordingly on the dashboard.

Workload
Different thresholds for different periods of the day. This relates to making the thresholds more dynamic and learning from experience. Analysis of earlier data is crucial in this case to figure out when the server workload is the heaviest throughout the day, week or month and adjust thresholds and entitled capacity accordingly. This is especially important in test environments where there are batch jobs for example running four hours a day where it has utilization well above its entitlement. This might be an opportunity to optimize performance.

7.2 Dashboard

An additional page devoted to settings and adjustments is recommended to create for the sake of customizability and ease-of-use. The settings page would include an option to add and remove data centers, options to filter out certain thresholds and ability to modify existing thresholds.

Additional improvement of the dashboard would be to implement a minute-based heatmap. The heatmap would be visualized with every Lpar above its entitlement usage during the past minute will be shown as an orange circle. In the case of a sudden overload / server attack targeting a specific machine, all of its Lpar might turn orange within a minute. This would give the system administrators an excellent minute-to-minute overview of how the servers are working.

A proposed idea from the AIX team which we believe would improve the usefulness of the dashboard would be to implement a feature that gives the system administrators a way to disable certain Lpars and machines from being monitored. This would be especially useful during scheduled maintenance. This could be done by adding functionality for user-input and letting the system administrator for example right click on an Lpar and mark it as “In maintenance”, this would turn the Lpar green or remove it from the dashboard altogether for a certain amount of time. Additional text should then be added to show how many Lpars are being monitored compared to the total amount.

8 References

[1] IBM. 2017. IBM AIX. [ONLINE] Available at: http://www-03.ibm.com/systems/power/software/aix/. [Accessed 4 April 2017].

[2] Information Technology Intelligence Consulting (ITIC) Corp, ITIC, 2016. ITIC 2016 Global Server Hardware, Server OS Reliability Report. INFORMATION TECHNOLOGY INTELLIGENCE CONSULTING, February 2016, 34.

[3] Information Technology Intelligence Consulting (ITIC) Corp, ITIC, 2013. Intel Xeon Processor E7 Family Reaches Reliability Parity with RISC/UNIX, Delivers 99.999% Reliability, Availability and Serviceability. INFORMATION TECHNOLOGY INTELLIGENCE CONSULTING , July 2013, 20.

[4] Wikipedia. 2017. IBM POWER microprocessors [ONLINE] Available at: https://en.wikipedia.org/wiki/IBM_POWER_microprocessors [Accessed 5 April 2017]

[5] Intel x86 and IBM Power CPUs: Which, when, why?, Terry Keene [ONLINE] Available at: http://www.computerweekly.com/opinion/Intel-x86-and-IBM-POWER-CPUs-Which-When-Why

[6] Pittman, SP, 2015. Understanding CPU utilization on AIX. AIX performance documentation, [Online]. 1, 1. Available at: https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/Power%20Systems/page/Understanding%20CPU%20utilization%20on%20AIX [Accessed 11 May 2017].

[7] IBM. 2010. Mainframe hardware: Logical partitions (Lpars). [ONLINE] Available at: https://www.ibm.com/support/knowledgecenter/zosbasics/com.ibm.zos.zmainframe/zconc_mfhwLpar.htm. [Accessed 4 April 2017]

[8] Peter Bergner. et al, Peter , 2015. Performance Optimization and Tuning Techniques for IBM Power Systems Processors Including IBM POWER8. 2nd ed. p.60 U.S: IBM.

[9] Adrian Demeter et al, 2004. AIX 5L Differences Guide Version 5.3 Edition. 1st ed. IBM.

[10] Guernion, YG, 2009. The Truth Behind IBM AIX Lpar Performance. ORSYP, Whitepaper, 15

[11] IBM. 2013. Hardware Management Console. [ONLINE] Available at: https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/Power+Systems/page/Hardware+Management+Console. [Accessed 4 April 2017]

[12] Lpar2RRD. 2017. Live Demo [ONLINE] Available at: http://demo.lpar2rrd.com [Accessed 5 April 2017]

[13] Cisco Engineers, Cisco, 2005. Baseline Process Best Practices White Paper. Technology White Paper, 15112, 22.

[14] Nagios. 2017. [ONLINE] Available at: https://www.nagios.org/. [Accessed 30 January 2017].

[15] Monitis 2017. [ONLINE] Available at:
http://www.monitis.com/blog/11-top-server-management-monitoring-software/
[Accessed 8th May 2017]

[16] Nagios Lpar_stat plugin [ONLINE] Available at:
https://exchange.nagios.org/directory/Plugins/System-Metrics/CPU-Usage-and-Load/check_Lparstat-2D1-2E1-2Esh/details [Accessed 24 April 2017]

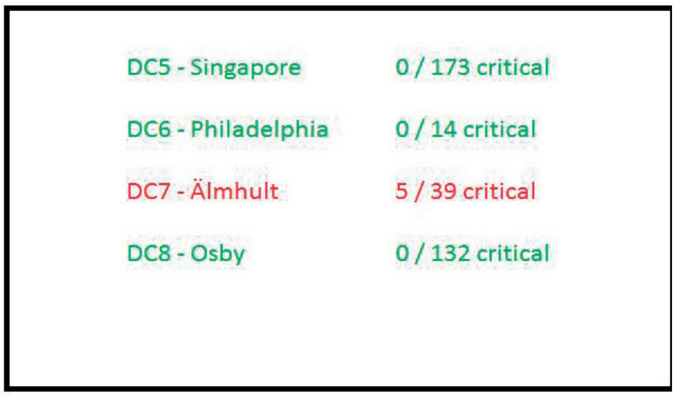
[17] Capacity Planning and Performance Management on IBM PowerVM Virtualized Environment p.23, Neeraj Bhatia [ONLINE] Available at:
https://neerajbhatia.files.wordpress.com/2011/10/powervm_capacity_planning.pdf
[Accessed 2 May 2017]

[18] Low Entitlement has a Bad Side Effect, Nigel Griffiths [ONLINE] Available at:
https://www.ibm.com/developerworks/community/blogs/aixpert/entry/local_nar_far_memory_part_5_low_entitlement_has_a_bad_side_effect4 [Accessed 2 May 2017]

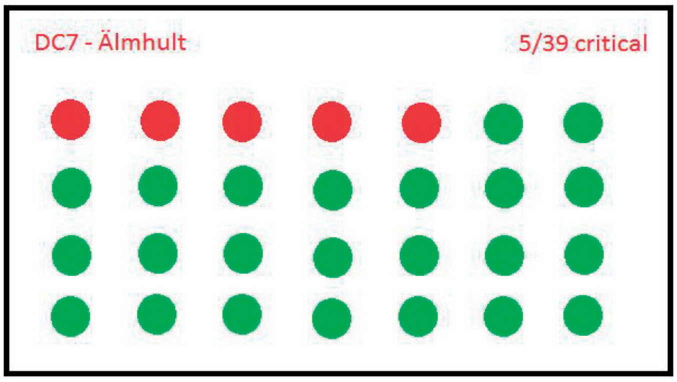
Appendices

This section presents illustrations of mockups (I-III). A questionnaire used for performing the dashboard usability test (IV). Collection of LPAR2RRD screenshots used for the evaluation (V). A screenshot of the final prototype (VI).

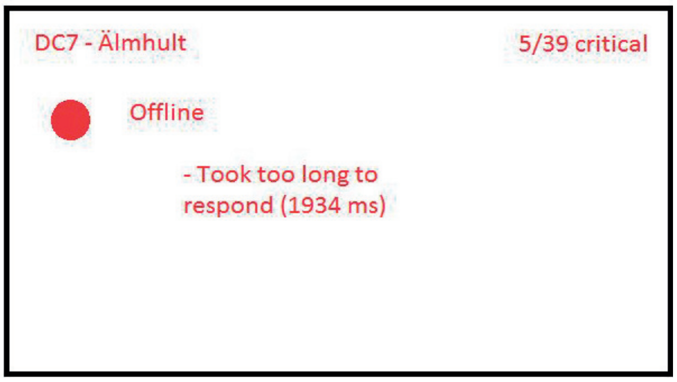
Appendix I



DC5 - Singapore	0 / 173 critical
DC6 - Philadelphia	0 / 14 critical
DC7 - Älmhult	5 / 39 critical
DC8 - Osby	0 / 132 critical



DC7 - Älmhult 5/39 critical

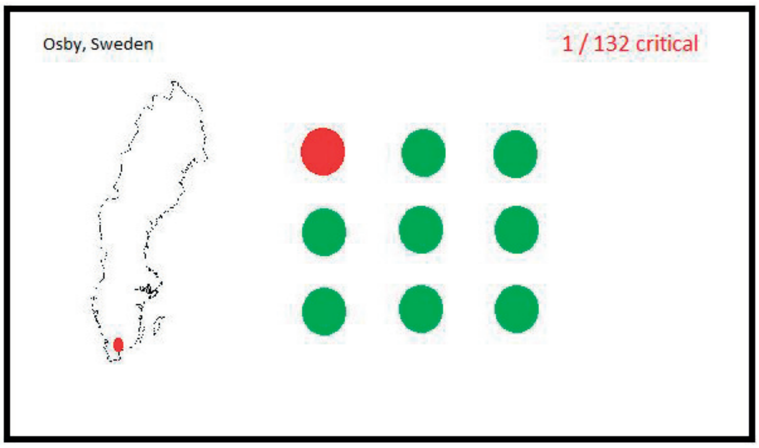


DC7 - Älmhult 5/39 critical

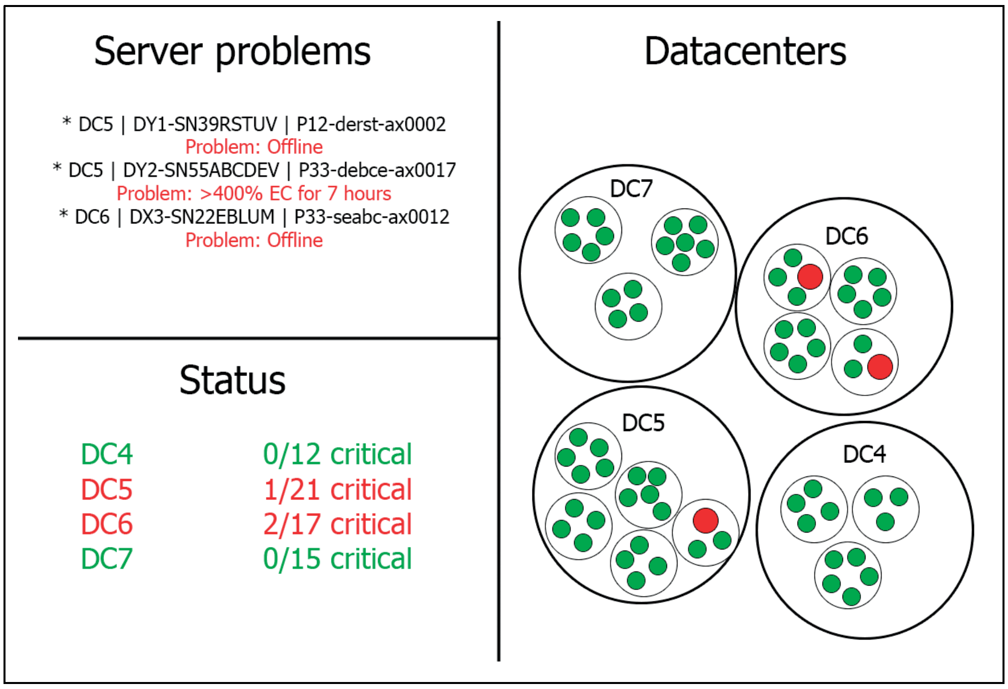
Offline

- Took too long to respond (1934 ms)

Appendix II



Appendix III



Server problems

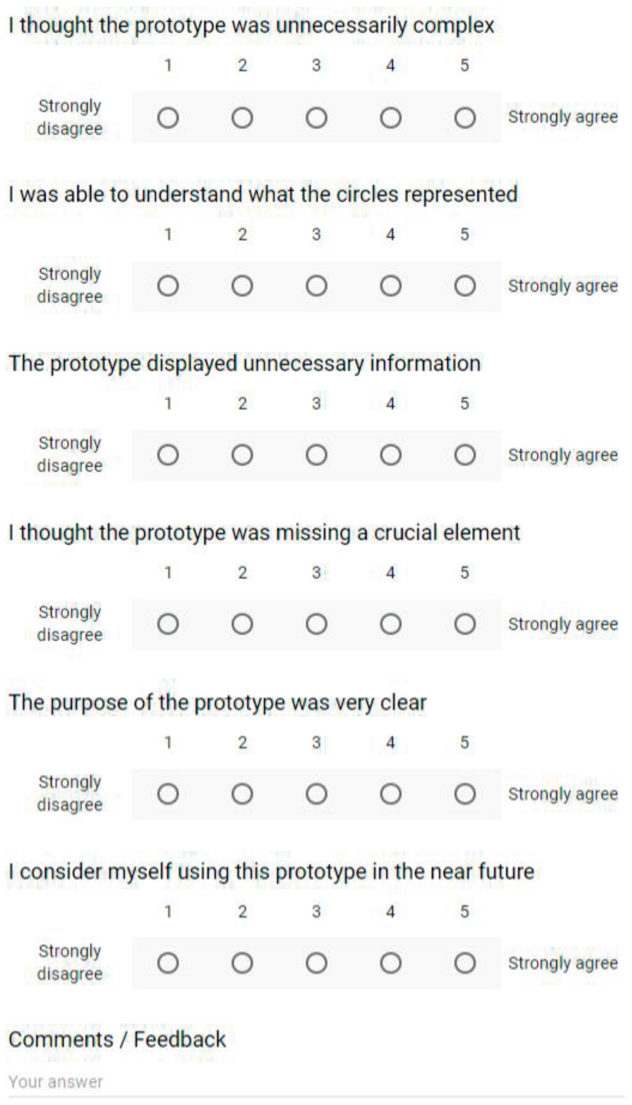
* DC5 | DY1-SN39RSTUV | P12-derst-ax0002
Problem: Offline
* DC5 | DY2-SN55ABCDEV | P33-debce-ax0017
Problem: >400% EC for 7 hours
* DC6 | DX3-SN22EBLUM | P33-seabc-ax0012
Problem: Offline

Status

DC4	0/12 critical
DC5	1/21 critical
DC6	2/17 critical
DC7	0/15 critical

Appendix IV

Usability poll for the AIX Team



I thought the prototype was unnecessarily complex

	1	2	3	4	5	
Strongly disagree	○	○	○	○	○	Strongly agree

I was able to understand what the circles represented

	1	2	3	4	5	
Strongly disagree	○	○	○	○	○	Strongly agree

The prototype displayed unnecessary information

	1	2	3	4	5	
Strongly disagree	○	○	○	○	○	Strongly agree

I thought the prototype was missing a crucial element

	1	2	3	4	5	
Strongly disagree	○	○	○	○	○	Strongly agree

The purpose of the prototype was very clear

	1	2	3	4	5	
Strongly disagree	○	○	○	○	○	Strongly agree

I consider myself using this prototype in the near future

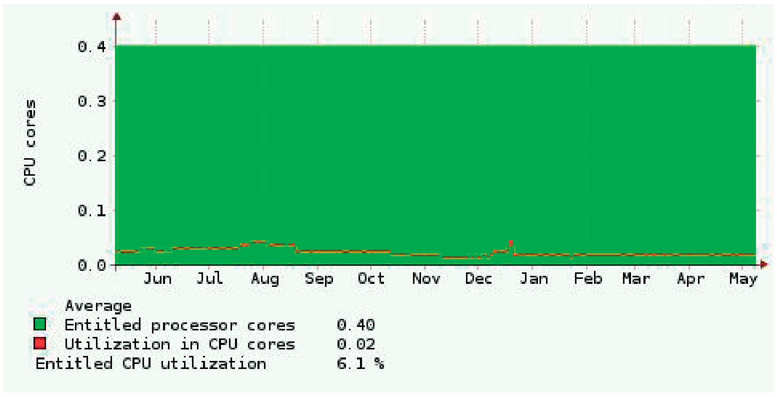
	1	2	3	4	5	
Strongly disagree	○	○	○	○	○	Strongly agree

Comments / Feedback

Your answer

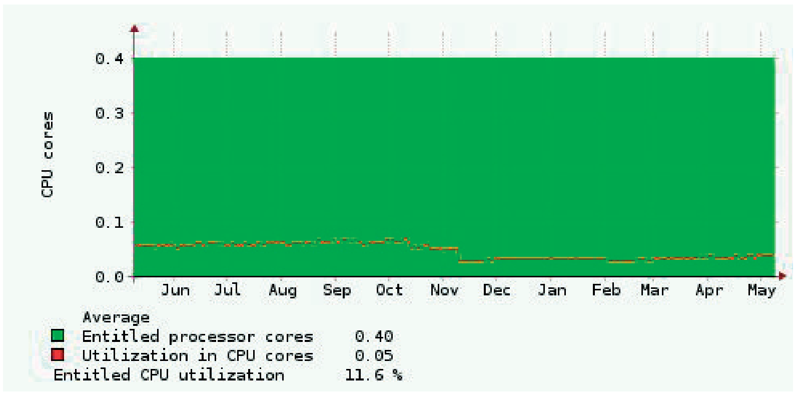
Appendix V

Lpar 1



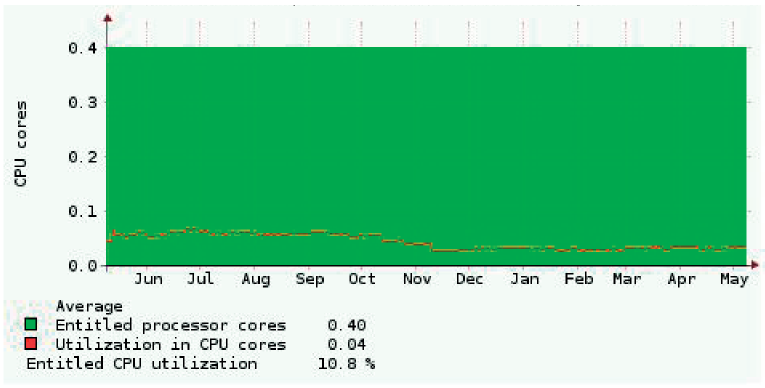
```
   Average
🟩 Entitled processor cores     0.40
🟥 Utilization in CPU cores     0.02
Entitled CPU utilization       6.1 %
```

Lpar 2



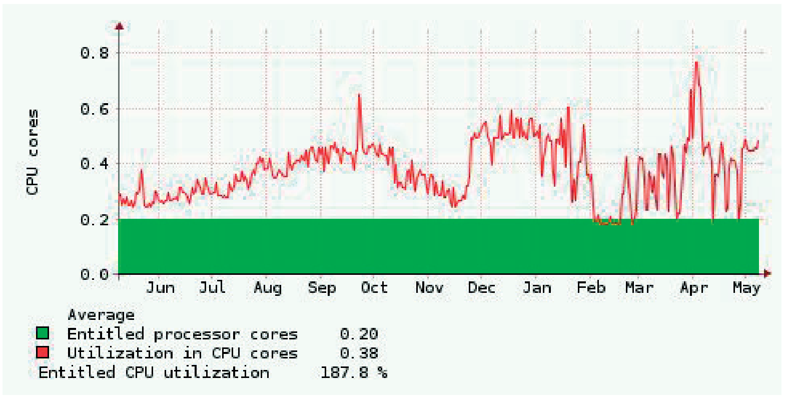
```
   Average
🟩 Entitled processor cores     0.40
🟥 Utilization in CPU cores     0.05
Entitled CPU utilization      11.6 %
```

Lpar 3



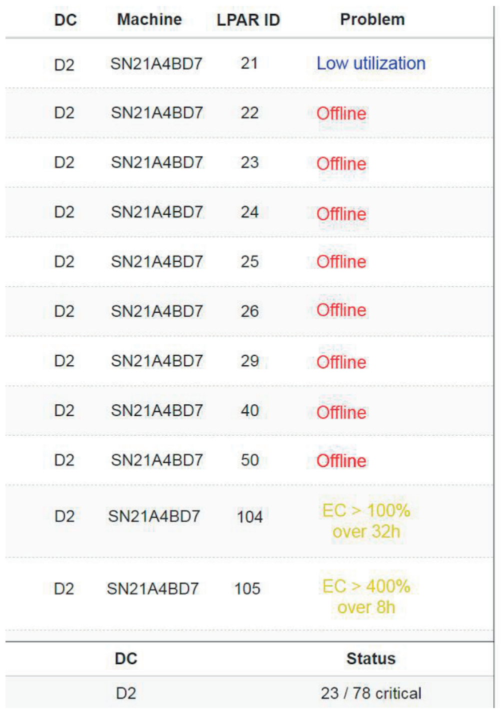
```
     Average
🟩  Entitled processor cores    0.40
🟥  Utilization in CPU cores    0.04
Entitled CPU utilization       10.8 %
```

Lpar 4



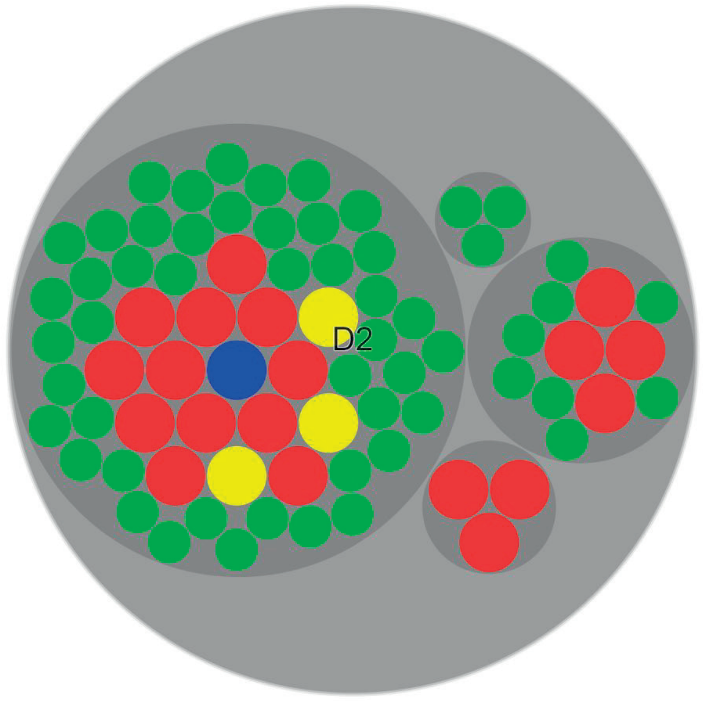
```
     Average
🟩  Entitled processor cores    0.20
🟥  Utilization in CPU cores    0.38
Entitled CPU utilization      187.8 %
```

Appendix VI



DC	Machine	LPAR ID	Problem
D2	SN21A4BD7	21	Low utilization
D2	SN21A4BD7	22	Offline
D2	SN21A4BD7	23	Offline
D2	SN21A4BD7	24	Offline
D2	SN21A4BD7	25	Offline
D2	SN21A4BD7	26	Offline
D2	SN21A4BD7	29	Offline
D2	SN21A4BD7	40	Offline
D2	SN21A4BD7	50	Offline
D2	SN21A4BD7	104	EC > 100% over 32h
D2	SN21A4BD7	105	EC > 400% over 8h

DC	Status
D2	23 / 78 critical



Printed by Tryckeriet i E-huset, Lund 2017

Series of Bachelor's theses
Department of Electrical and Information Technology
LU/LTH-EIT 2017-580

http://www.eit.lth.se