BUILDING APMv3 MAP VISUALIZATION
Using Nagios Host Data

Fontys Graduation Project Report
Hans Sebastian Tiono
Eindhoven, May 2015. hans_sebtino@yahoo.com
GRADUATION REPORT  
FONTYS UNIVERSITY OF APPLIED SCIENCES  
HBO-ICT: English Stream

<table>
<thead>
<tr>
<th>Data student:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Family name, initials:</td>
<td>Tione, HST</td>
</tr>
<tr>
<td>Student number:</td>
<td>2583399</td>
</tr>
<tr>
<td>assignment period: (from – till)</td>
<td>February 2nd – June 30th 2015</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data company:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name company/institution:</td>
<td>Acknowledge Benelux BV</td>
</tr>
<tr>
<td>Department:</td>
<td>Acknowledge Proactive Monitoring (APM)</td>
</tr>
<tr>
<td>Address:</td>
<td>Burgemeester Mollaan 80, 5582 CK Waalre</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Company tutor:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Family name, initials:</td>
<td>Peperkamp, TP</td>
</tr>
<tr>
<td>Position:</td>
<td>Head of APM</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>University tutor:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Family name, initials:</td>
<td>Oosterkamp, JO</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Final report:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Title:</td>
<td>Building APMv3 Map Visualization Using Nagios Host Data</td>
</tr>
<tr>
<td>Date:</td>
<td>June 9th, 2015</td>
</tr>
</tbody>
</table>

Approved and signed by the company tutor:

Date : 7 June 2015  
Signature : [Signature]

[Signature]
Preface

This document is the final report of Fontys graduation project which was started from February 1st until June 30th 2015 and took place at Acknowledge Benelux BV, a company that provides business, service, and ICT solutions to its clients. The title of the report is “Building APMv3 Map Visualization Using Nagios Host Data” and DOT framework was used within the research parts of the report. As a brief impression, I learned so many technical things about website development through this graduation project and I am encouraged to learn more. I have chosen to have a graduation project at Acknowledge because the assignment they offered is interesting for me and I felt this company could be a good place to develop myself.

As the graduation project and report have been finished on time, first of all I want to thank Lord Jesus for I believe He has given me wisdom and strength to get through this graduation project. I realize that a doctor can give limited medications, but only God who can give good health, as well as for me a teacher can give limited knowledge but only God who can give wisdom and intelligence.

I would also like to express my gratitude to my company tutor, Mr. Tom Peperkamp because he is very good and kind to me. He helped me evaluating my work patiently, giving many positive advices, and even helping me to write a good writing with correct English grammar. I would also like to thank my university tutor, Mr. Jelle Oosterkamp for his guidance and suggestions for me, so that I could carry out this graduation project. I want to also use this opportunity to express my special gratitude to my family in Indonesia, my mother, my father, and both my sisters for their moral supports to me. I am glad to know that they love me and always care for me. Also, I want to thank my colleagues from APM, Teun, Ruben, Marno, and Jimmy for helping me a lot, my church friends for their spiritual supports, my friends in Indonesia for sharing their kindness and joy, and my friends in Fontys, especially Danny Benlin Oswan who is one of my best colleagues both in Fontys and Indonesia.

I hope that this graduation project will give a satisfying and useful result to the company. Last but not least, I hope this graduation report could be beneficial for other students who are interested in having a research in the same field.

Waalre, June 9th 2015

Hans Sebastian Tiono

The Author
Table of Contents

Summary ...............................................................................................................................5
Glossary ..............................................................................................................................6
Chapter 1: Introduction .....................................................................................................7
Chapter 2: About the company .........................................................................................8
  2.1. Company Description .............................................................................................. 8
  2.2. Company Organization ......................................................................................... 8
  2.3. Mission, Vision, and Values ................................................................................... 9
  2.4. Acknowledge Proactive Monitoring (APM) ............................................................ 9
Chapter 3: Assignment overview ....................................................................................10
  3.1. Background ........................................................................................................... 10
  3.2. Current Problem .................................................................................................. 12
  3.3. Goal ..................................................................................................................... 14
  3.4. Assignment Description ....................................................................................... 14
  3.5. Final Product ........................................................................................................ 14
  3.6. Functional Requirements .................................................................................... 14
  3.7. Technical Requirements ..................................................................................... 14
  3.8. Project Deliverables and Non-Deliverables .......................................................... 14
  3.9. Project Phasing .................................................................................................... 15
  3.10. Project Constraints ............................................................................................. 16
  3.11. Research ............................................................................................................ 16
Chapter 4: Project Initiation .............................................................................................17
  4.1. Introduction .......................................................................................................... 17
  4.2. Research Result ................................................................................................... 17
  4.3. Conclusion and Decision ..................................................................................... 17
Chapter 5: Project Design ................................................................................................18
  5.1. Introduction .......................................................................................................... 18
  5.2. Entity Relationship Diagram (ERD) .................................................................... 18
  5.3. Class UML Diagram ............................................................................................ 18
  5.4. Application Interface ............................................................................................ 19
Chapter 6: Project Construction .......................................................................................20
  6.1. Introduction .......................................................................................................... 20
  6.2. Frameworks or Plugins used ............................................................................... 20
  6.3. Building Icon on APMv3 Map Visualization ....................................................... 20
  6.4. Building Category on APMv3 Map Visualization ............................................... 21
  6.5. Building Map on APMv3 Map Visualization ....................................................... 22
  6.6. Building Asynchronous Backend Connection on APMv3 Map Visualization ....... 24
  6.7. Building Activities Tracker on APMv3 Map Visualization ................................... 27
Chapter 7: Map Editor and Map Viewer ..........................................................................29
  7.1. Introduction .......................................................................................................... 29
  7.2. Map Editor .......................................................................................................... 29
  7.3. Map Viewer ......................................................................................................... 33
Chapter 8: Project Functional Testing .............................................................................35
Chapter 9: Conclusions and Recommendations ........................................37
Evaluation ........................................................................................................38
References/Literature List ..............................................................................39
Appendix A: Research Report .........................................................................I
Appendix B: Design Documents ......................................................................XV
Appendix C: Test Report ................................................................................XXVI
Appendix D: User Manual ...............................................................................XXXII
Appendix E: Project Plan ................................................................................XLIX
Appendix F: Graduation Project Survey Form ..............................................LVIII
Summary

The graduation project took place at Acknowledge Benelux BV, a company which consists of three business lines: Business Solutions, Service Solutions and ICT Solutions. The assignment was given by APM as one of the company’s sub-departments. APM has built a network monitoring application called APMv2 and they are currently building a new version called APMv3. Both APMv2 and APMv3 use Nagios, a well-known open source monitoring tool.

One of the main features in APMv2 is Map Visualization which mainly uses a standard visualization plugin from Nagios called NagVis. However, APM team thinks that it is not really efficient to use NagVis in terms of creating and managing its maps. NagVis needs third party diagram builder software like Microsoft Visio to create its network background. The form of all used backgrounds is image-based file (for example .jpg) which takes time to edit, arrange and store the files. Network infrastructure in a company could possibly change, resulting to the map changes as well. Whenever any change in network happened, APM team had to re-edit the map background in Visio to adjust the current network condition. In fact, in the last four months (from January until April 2015) APM had to edit 223 map backgrounds and it took more or less 10 minutes per map. So, APM had to spend extra 37 hours in total to edit all the maps. The cost price of one employee is €55 per hour and the company had to spend €2035 only to edit maps in Visio for only 4 months. That is why NagVis as the current Map Visualization in APMv2 is not so efficient and takes a lot of time and money.

The general scope of the assignment is to research and build Map Visualization feature in APMv3 to be more efficient than the one in APMv2. There are three possible ways to accomplish the assignment: first is to keep using NagVis with some improvement or optimization to be more efficient, second is to use another available visualization tool other than NagVis, and third is to build the visualization tool itself from scratch. In order to decide which alternative was the best, some researches about NagVis and other possible visualization tools were done and it turned out that the third alternative, building the tool from scratch is the best and the only possible solution. It gives a lot of advantages, as the created tool itself is very flexible to APMv3 application. It was made from zero and it could easily adjust or adapt to what the application needed. The final Map Visualization could be considered as a proof of concept of APMv3 map visualization.

The system development for the project is incremental using the waterfall approach. It means that there will be small waterfall processes in every increment. The reason of using the incremental approach is because the project can be divided into several parts or progresses. In short, the increments are: first, the project initiation, second, the project design, third, the map host construction, fourth, the map host relation construction, and the last one, project functional test. The finished application will be installed in APMv3 as its Map Visualization feature. It has a separate frontend (written in HTML and JavaScript) and a separate backend (written in PHP). All the data exchanges between the frontend and the backend will use AJAX method. The application would be better if there are authentication/authorization features and thorough security implementation within the application as two further recommendations that can be applied later.
Glossary

Acknowledge Proactive Monitoring (APM) – One of the sub-departments in Acknowledge Benelux B.V.

APM version 2 (APMv2) – The application that is developed and used by APM to do network and IT infrastructure monitoring. This application is the current stable version.

APM version 3 (APMv3) – A newer version of APM application and still under development.

Application Programming Interface (API) – In APM context, this is a self-built web interface that can be illustrated as a bridge between the application and database which converts the data from/into database into a readable and well-known format like JSON.

Asynchronous JavaScript and XML (AJAX) - A group of interrelated Web development techniques used on the client-side to create asynchronous Web applications.

Category – A set or group of icons based on their similarities and functions.

Host – In APM context, it’s an object that is monitored by APM application.

Icon – In APM context, it’s an image representation of the host to be placed on the map.

JavaScript Object Notation (JSON) – A lightweight data-interchange format.

Map – A network diagram used to view all the hosts inside it along with their status.

Map host – An icon inside a map that will represent a real host and will reflect real host status when the corresponding map is viewed.

Map host relation – Usually also called relation, a connection between two map hosts.

Map Visualization – One of the features in APM application as well as a term to describe the process of converting hosts data into an easy and understandable network diagram

Nagios – A well-known open source monitoring tool.

Nagios Visualization (NagVis) – Visualization tool made for Nagios

Visualization – A technique to make raw data to be useful and visual in the form of image, diagram, or graph.
Chapter 1: Introduction

Not many people know what network monitoring and its importance are for a company. Nowadays, having a network monitoring is becoming an important thing for a growing or big company which should also have a network infrastructure inside it. Network monitoring is the use of a system that constantly monitors a computer network for slow or failing components and that notifies the network administrator (via email, SMS or other alarms) in case of outages.

APM is one of the sub-departments in Acknowledge as well as the place where the author works as an intern. The main purpose of APM is to develop a system to monitor its clients’ network and IT infrastructures. So, whenever a problem occurs within their IT infrastructure, they will notice it by looking at their real-time host statuses (up/down/unreachable/pending). The team uses an in-house developed application called APMv2 and there will be a new version called APMv3. One of the main features in APMv3 is the Map Visualization. It is a feature to visualize the IT infrastructure of APMv3 in a form of a network diagram. By displaying a visualized map, both APM team and its clients could easily understand the whole IT infrastructure and help them to fix any problem that may occur.

APMv2 uses a plugin called NagVis for the Map Visualization feature. However, APM team thinks that it is not really efficient to use NagVis in terms of creating and managing its maps. The author’s main task is to research and build a more efficient Map Visualization feature that later will be used on APMv3.

This report contains nine chapters: chapter 2 is about the company, chapter 3 basically explains what the assignment is, chapter 4 explains how the assignment started, chapter 5 provides the designs of the application, chapter 6 contains how the application was built, chapter 7 explains Map Editor and Map Viewer as two most important pages in the application, chapter 8 is the application functional test, and chapter 9 contains conclusions and recommendations.
Chapter 2: About the company

2.1. Company Description

Company Name: Acknowledge Benelux BV
Location: Burgemeester Mollaan 80, 5582 CK Waalre, The Netherlands

Figure 2.1. Company logo

Acknowledge is a company which provides business and IT solution to its clients. The company has currently 230 workers, who work in various departments.

Acknowledge was founded in 1994 and was started as an IT supplier which provided box products, software licenses and small solutions. Throughout the years the company expanded its services towards its current portfolio and size.

2.2. Company Organization

In general, Acknowledge has three business lines: Business Solutions, Service Solutions and ICT Solutions. There are several sub-departments in each department. Below is the organization chart of the company.

Figure 2.2. Company organization chart in Dutch

Here is a brief explanation about the three main operations of Acknowledge:

2.2.1. Business Solution

Business Solution department helps clients in their business-related problems. The values that are offered to clients are E-Business (helping clients to build their web shops, websites, or web portals), Risk Management (including information security, auditing, risk analysis, and data analysis), Business Consultancy, Logistics Processes (SAP), and Business Development.
2.2.2. **Service Solution**
Service Solution is the business operation which handles client data and IT system to make sure that the clients’ system becomes safe, stable, and fast. More importantly, the system should be kept and maintained the way it is. The values that are offered in this operations can be divided into three, Service Solutions (including Proactive Monitoring, Service Management, and Application Management), Mobility (making sure that client has a system that is mobile-friendly and accessible all over platform), and Cloud Management (availability, capacity, security and continuity). It was mentioned about APM in Figure 2.2 (marked with yellow text color) which is managed by APM team.

2.2.3. **ICT Solution**
ICT Solution is a department which helps clients decide the best and suitable IT solution for their business. ICT Solution does client consultation and gives advice so clients can have a good decision. The values of this department are client advice and project implementation which can be for Hardware/Software, IT infrastructure/implementation, and Connection/Communication.

2.3. **Mission, Vision, and Values**
The vision of Acknowledge is *Passie voor uw resultaat* which can be translated as *Passion for your* (the clients) *business outcome*. Acknowledge has three missions: First, to help clients achieving all their business goals, second, to take care of the total client experience by connecting processes and services through the clients’ organization, and third, to use effective IT as a basis for the clients’ growing organization. There are several values in Acknowledge which are written on Figure 2.3.

![Figure 2.3. Acknowledge Values](image)

2.4. **Acknowledge Proactive Monitoring (APM)**

![Figure 2.4. Acknowledge Proactive Monitoring logo](image)

APM is one of the sub-department in Acknowledge which focuses on developing a system to monitor its client’s computer systems, network, and infrastructure environment (usually in general they are called *hosts*). The client’s hosts have status such as *up/down/unreacheable/pending*, and they are visible in real time. So, whenever a host has a problem, both the client and Managed Services department will notice from the monitoring result. Then, they can be alerted that some required actions should be performed to fix that problem. The corrective actions will be done either by APM or the client himself, depending on the contract which has been signed between APM and the client. APM team doesn’t use any local repository to store their works and code, but they use Git instead as their separate storage, so it supports the flexibility of their work.

The author works in this sub-department as an intern. There are 5 people who work in this sub-department as employees and two people as interns (including the author). Tom Peperkamp is the head of APM as well as the company tutor of the author.
Chapter 3: Assignment overview

3.1. Background

Until now, APM team uses in-house developed application called APMv2 and it is the current stable version. The team is developing a new version of the application called APMv3.

3.1.1. APMv2 Application

APMv2 is functioning properly and has been broadly used to monitor clients’ hosts. The application is web-based and it is written mostly in Python and partially in PHP. Figure 3.1 shows APMv2 main interface which contains general overview of all APM’s maintained hosts.

![Figure 3.1. APMv2 home page as well as the status overview page](image)

There are several terms in APMv2 such as host and service. Both of them are the main objects that are monitored by APMv2. Host in APMv2 basically refers to the computer, server or device which can be in physical or virtual form with a specific IP address. While service is a process running on a host. A host can have more than one service or possibly has no services at all, depending on the host type. There are some particular types of host which does not have any services.

Each host has a specific status (up/down/unreachable/pending). Figure 3.1 is APMv2 status overview page which has two pie charts indicating the hosts (left pie chart), and services (right pie chart) status overviews. Every color has a meaning, green is up, red is down, yellow is unreachable, and blue is pending. APM users can simply click on the status to see the hosts or services with the specified status.

One of the main functionalities in APMv2 is the Map Visualization. In this case, map refers to the network logical or physical diagram. Map visualization visualizes the client IT environments according to APM standard. One of the reason to use map visualization is to help the users better understand the connectivity between each host resulting in doing a faster impact analysis of the monitoring result. By visualizing the standardized map, users can notice easily how the IT infrastructure environment was constructed.
APMv2 application uses a tool called Nagios for the whole monitoring functionality. For the Map Visualization functionality, APMv2 uses a tool called NagVis. More information and explanation about Nagios and NagVis can be found in Appendix A.

3.1.2. **APMv3 Application**

APMv3 has the same functions and still uses Nagios as its monitoring tool but there will be some improvements on the display interface, Map Visualization feature, Reporting Service feature, and data storing and retrieval using API. The main purposes of building APMv3 are to improve major functionalities, to make a robust and scalable software, and to increase user experience through a well-built user interface. The database used in APMv3 is MySQL and the programming languages used in APMv3 are Python, HTML, and JavaScript.
APMv3 will use an API called APM API which is a door to some supportive backend data in APMv3 application. The API also helps APM standardizing the data flow (the data that goes out from database to user and the other way around), storing some application protocols, and helping the user authentication and/or authorization processes. There is also a new term used in APMv3 called Elastic Search which can hold some particular data and save it to a cache, to be used later whenever the users want to query those specific data.

3.2. Current Problem
Due to a version change from APMv2 to APMv3, APM needs to rebuild the system all over again and it takes a lot of works. One of the major works needed to be done is the Map
Visualization functionality in APMv3. Previously in APMv2, NagVis is used but it is not very efficient. APM wants to have a better and more efficient Map Visualization for the APMv3 application which retrieves data only via APM API. Simple illustrations below on how APM users create a new map (Figure 3.4) and maintain it (Figure 3.5) will show why the way NagVis works is considered as not very efficient:

![Figure 3.4. Creating a new map in APMv2 Map Visualization](image)

1. First, they have to make an image based file (.jpg/.png) network diagram in third party software like Microsoft Visio or any diagram builder.
2. Second, they have to upload the image to NagVis application
3. Third, they need to add host/service symbol on each relevant host in the network diagram one by one. NagVis is just simply adding small icons which indicate that they are hosts and/or services and the icon will serve the status of corresponding host or service.

The map that has been created and added some host icons will be active. NagVis will automatically relate the host icon to a specified real host status. So, the status on the map (up, down, etc) will follow the real status. However, the map will be difficult to be maintained because whenever a change or modification occurs in the map (shown in Figure 3.5):

![Figure 3.5. Modifying a current map in APMv2 Map Visualization](image)

1. They have to edit the map in third party diagram builder (e.g. Microsoft Visio) again.
2. Then, they have to re-upload the image and set it as the background. In Figure 3.5, the host icons remain the same as the first version of the map because NagVis just saves their x and y positions.
3. Then, the users have to re-adjust the host icons.

The map in Figure 3.5 is simple and it might be still durable, but if a map has for instance 50 or 100 hosts, it would be a very exhausting work.
In fact another problem is, in the last four months (from January until April 2015) APM had to edit 223 maps. The estimated time savings with a more efficient tool are 10 minutes per map. So, APM had to spend extra 37 hours in total to edit all the maps. The cost price of one employee is €55 per hour and the company had to spend €2035 only to edit maps in Visio for only 4 months. That’s why the current Map Visualization is not so efficient in terms of time and money spent.

3.3. Goal

The project was initialized with a purpose to improve the current Map Visualization on APMv2 to be applied later in the APMv3. The company hopes AMPv3 has a more efficient Map Visualization and could exchange every data via API.

3.4. Assignment Description

The project in overall is to research and build a new way in visualizing the monitoring results for APMv3 map visualization. There are several possibilities to accomplish the project requirements. First, NagVis might still be used but it will be modified to become a more like object oriented visualization tool. Second, using other completely different visualization tools or facilities other than NagVis. Third, building a working visualization tool from scratch to be applied in APMv3.

3.5. Final Product

The final product is a tool to create and maintain map visualizations for APMv3, showing the current hosts’ status in a form of a webpage which will be named “APM Map Visualizer". The host data are based on Nagios data that are retrieved from APM API and stored as JSON files, and the map data will use a local database which are retrieved via temporary PHP API. The expected interface and functionality should be as user-friendly as possible. The final product can overcome the current map visualization’s problem which requires APM users to recreate the image-based map whenever a change occurs in the map.

3.6. Functional Requirements

Here are the functional requirements of the program:
- Add/edit/delete icons
- Add/edit/delete/view maps including their hosts and relations
- Add/edit/delete icon categories
- View activities

3.7. Technical Requirements

APMv3 Map Visualizer is responsive because the map size follows the screen resolution. However, the application is designed in a computer environment which has screen resolution of 1680x1050px, so it is highly recommended to use this application on 1680x1050px screen resolution.

The application has been tested on several browsers and it works well on Google Chrome, Mozilla Firefox, Opera, and Safari. It still works on Internet Explorer but there is a strange behavior when it comes to asynchronous connection of the application, so it’s not recommended to use it on Internet Explorer. Chrome is the most recommended browser to be used since the application was developed using it.

3.8. Project Deliverables and Non-Deliverables

All deliverables that can be mentioned and described for the whole project is listed below in Figure 3.6.
Log book is one of the deliverables in the project. It contains all of the activities that have been done during the graduation project along with its time duration. The item that will not be delivered or non-deliverable is meeting minutes with university tutor or company tutor.

### 3.9. Project Phasing

The system development for the project can be considered as incremental using the waterfall approach. It means that there will be small waterfall processes in every increment. The reason of using the incremental approach is because the project can be divided into several parts or progress. Below is the project phasing that has been planned.

<table>
<thead>
<tr>
<th>Increment</th>
<th>Work / Activity</th>
<th>Completion</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Doing a library research about NagVis and other monitoring tools besides NagVis. After that, making a decision whether to stay using NagVis with an object oriented optimization or to find another possible Map Visualization tool besides NagVis. If it seems to be nearly impossible in editing NagVis file or there is no compatible tool found to use for Nagios data, the last decision is to create the tool itself using plugins available. The decision should be based on deep researches and findings.</td>
<td>Completed</td>
</tr>
<tr>
<td>2</td>
<td>Designing APMv3 map visualization.</td>
<td>Completed</td>
</tr>
<tr>
<td>3</td>
<td>Building and implementing APMv3 Map Visualization including map host and map icon.</td>
<td>Completed</td>
</tr>
<tr>
<td>4</td>
<td>Building relation for the map host created in the Increment 3</td>
<td>Completed</td>
</tr>
<tr>
<td>5</td>
<td>Writing and Executing functional test plans</td>
<td>Completed</td>
</tr>
</tbody>
</table>

Table 3.2. Project phasing

![Project phasing diagram](image)
3.10. Project Constraints

Both the final application (the comments or variables and functions naming) and all of the documents were written in English. The application itself was written in HTML and JavaScript for its frontend and PHP for its backend. The database is MySQL, using PHP Data Objects (PDO) as the database access and retrieval method.

3.11. Research

Some of the research questions were changed compared to the previous research questions in the project plan to adjust the current situation. The main research question is: “What is the most efficient and user-friendly way to build Map Visualization that suits APMv3?”. There are several sub-questions which can be found in Appendix A.
Chapter 4: Project Initiation

4.1. Introduction
This chapter was written to describe the very first steps of the project. According to the first increment of the project phasing, a big decision had to be made, either to stay using NagVis with an object oriented improvement, to find another possible tool besides NagVis, or to build a working visualization tool from scratch to be applied in APMv3.

4.2. Research Result
After some research, there are three possible options to be used to accomplish the project goal. All the three possible options can be considered as a research and will be written in detail in the research report (Appendix A). This point covers a brief summary of the research.

4.2.1 Improving NagVis
Improving NagVis is possible, but nearly impossible for those who did not contribute in making it or people outside the developer area because of the complexity of its code.

4.2.2 Using Another Available Visualization Tool
There is another Map Visualization tool for monitoring purpose called Zabbix. However, since the Map Visualization of Zabbix is Zabbix’s proprietary, then it was not possible to apply it for APMv3 which has a developed Nagios environment inside it.

4.2.3 Creating Self-Made Visualization Tool
The only possibility would be to create a self-made map visualization feature for APMv3. The criteria to determine that this possibility is the best one are the reasonability, flexibility, sustainability, and suitability to APMv3. The advantages of creating a self-made map visualization tool are:
- The tool is fully flexible, no need to follow some tool regulations or functions and 100% unique so it has a sale or market value
- The tool can be adjusted to be completely suitable to APMv3 and can exactly fulfill what APMv3 needs and all kind of features can be easily added
- No need to study an available visualization tool and the way it works
- At the end, the tool creator can master his tool, which means he will understand the tool program as a whole
The disadvantages of creating a self-made map visualization tool are:
- Additional research has to be done regarding what frameworks should be used and how to build the tool,
- It will be a lot works since the tool is built from zero
- There will be some risks that the tool doesn’t work properly or there are some bugs, because a new freshly-built tool can’t be compared to a mature visualization tool that has been through a lot of testing like NagVis. A careful application development is needed.

4.3. Conclusion and Decision
Thus, it had been decided to create map visualization from scratch. The advantages can outweigh the disadvantages because the final application will satisfy the most desirable purpose of APMv3 map visualization: to retrieve data only via API, as one of its main advantage which is the full flexibility of the application. Other alternatives might be nearly impossible to accomplish that and needs several backends other than the API.
Chapter 5: Project Design

5.1. Introduction

This chapter is the second increment of the project phasing, designing APMv3 map visualization. Before actually making the Map Visualization, it is very important to make its designs first. The chosen decision in chapter 4 was to build the Map Visualization feature for APMv3 from scratch, so all the designs would start over.

There are three planned designs: Entity Relationship Diagram (ERD), class UML diagram, and the application interface. NagVis doesn’t use database to store data, but it uses config files instead. The similarity between NagVis config files and the ERD can be seen from the data structure of maps and maps_detail table in the ERD. The content of this chapter refers to the Design Document (Appendix C).

5.2. Entity Relationship Diagram (ERD)

For APM Map Visualizer, the ERD contains all the data structures that are stored in the database. This ERD will explain the relation between one table to another.

![Entity Relationship Diagram of APM Map Visualizer](image)

Figure 5.1: Entity Relationship Diagram of APM Map Visualizer

5.3. Class UML Diagram

This Class Diagram of APMv3 Map Visualizer shown in Figure 5.2 helps to better understand the structure of the whole application, the components in the application and the relation between each component.
5.4. Application Interface

The application interface was built using Bootstrap 3 CSS framework. Figure 5.3 below shows a preview of the application interface, specifically on map editor page.
Chapter 6: Project Construction

6.1. Introduction
This chapter covers the third and fourth increments and will explain the project creation. The project is a web-based application consisting of seven frontend pages and five backend files or endpoints.

6.2. Frameworks or Plugins used
There are some frameworks used as plugins in the application. Those plugins are some JavaScript frameworks (jQuery, jQuery UI, jCanvas, Twitter Typeahead, PNotify) and a CSS framework (Bootstrap 3). Bootstrap 3 and jQuery are already used in APMv3. The description of all the plugins can be found in Appendix A.

6.3. Building Icon on APMv3 Map Visualization
The first thing to do is to build the icon management of the application. This includes add/edit/delete icons functionalities. The icon will be later used on a map as an image representation of a host. A single icon is designed to be in 100x100 pixel size, but later, it can be resized to be bigger or smaller while it is used on the map.

6.3.1. Icon Image
Each map host must have exactly one icon image, so it is mandatory to link an icon with an image URL in the database. Figure 6.1 shows a clear example of how the host icon looks alike. Almost all the images are taken from Microsoft Visio.

![Figure 6.1. Examples of APMv3 Map Visualizer icon image](image)

All the configuration of icons can be found in Icon Manager page, which has the functionalities of adding, editing, and deleting icons. This page is dealing with image upload feature, in order to add an icon or edit an existing icon. Therefore, it uses HTML `input type="file"` within a form to upload the image. The uploaded file extensions must be either .gif, .jpeg, .jpg, or .png, otherwise the application backend would not accept the file and it will return an error message saying that the specified file type is not acceptable. The preferable file type is PNG because it supports transparency and has a better image color quality than .gif. PNG was created as an improved, non-patented replacement for “.gif”, and is the most used lossless image compression format on the Internet. The application limits the uploaded file size to be 3MB maximum size per image.

6.3.2. Icon Last File Naming
The icon images are named consecutively in a folder. The file names of the icons for example are 1.png, 2.png, 3.png, 4.png. If the user adds a new icon, it will be named 5.png. The last number of the icon, which is 5 is stored in a separate table called `last_filenaming`. Every time APM user adds a new icon, this table will be modified with the latest file name of the icon.

6.3.3. System Icon
A single icon could be either a system icon or just a normal icon. System icon is an icon that can neither be created, deleted nor edited by anyone except the system administrator (the current
application still doesn’t have any authorization, so it just can’t be deleted). The purpose of the system icon is to represent the general standard icons in APMv3 Map Visualizer. The system icons have existed since the application was built. Any new icon that is created will be just a normal or custom icon icon, which can be later edited or deleted. The application will distinguish if an icon is a system icon or not by looking upon its Boolean table attribute called is_system.

6.4. Building Category on APMv3 Map Visualization

Each icon is categorized in a category which means every icon must have exactly one category. All the configuration of the categories in APM Map Visualization can be found in Category Manager page, which has the functionalities of adding, editing, and deleting categories.

![Infrastructures category Applications category](image)

Figure 6.2. An example image explaining what category is.

6.4.1. List of all categories and icons

There are some initial categories and icons (Table 6.1) that are considered necessary.

<table>
<thead>
<tr>
<th>Category</th>
<th>Icon name</th>
<th>Icon</th>
<th>System icon</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Infra-structure</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(system category)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iconless Host</td>
<td></td>
<td>![Icon1]</td>
<td>Yes</td>
</tr>
<tr>
<td>Computer</td>
<td></td>
<td>![Icon2]</td>
<td>Yes</td>
</tr>
<tr>
<td>Server</td>
<td></td>
<td>![Icon3]</td>
<td>No</td>
</tr>
<tr>
<td>Router</td>
<td></td>
<td>![Icon4]</td>
<td>Yes</td>
</tr>
<tr>
<td>Switch</td>
<td></td>
<td>![Icon5]</td>
<td>Yes</td>
</tr>
<tr>
<td>Firewall</td>
<td></td>
<td>![Icon6]</td>
<td>Yes</td>
</tr>
<tr>
<td>Storage</td>
<td></td>
<td>![Icon7]</td>
<td>Yes</td>
</tr>
<tr>
<td>Air Conditioner</td>
<td></td>
<td>![Icon8]</td>
<td>Yes</td>
</tr>
<tr>
<td>UPS</td>
<td></td>
<td>![Icon9]</td>
<td>Yes</td>
</tr>
<tr>
<td>Telephone</td>
<td></td>
<td>![Icon10]</td>
<td>No</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Category</th>
<th>Icon name</th>
<th>Icon</th>
<th>System icon</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Platform</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(system category)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Windows Host</td>
<td></td>
<td>![Icon11]</td>
<td>Yes</td>
</tr>
<tr>
<td>Linux Host</td>
<td></td>
<td>![Icon12]</td>
<td>Yes</td>
</tr>
<tr>
<td>Database</td>
<td></td>
<td>![Icon13]</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Applications</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(system category)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Website</td>
<td></td>
<td>![Icon14]</td>
<td>Yes</td>
</tr>
<tr>
<td>Active Directory</td>
<td></td>
<td>![Icon15]</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Clients</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(system category)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sensor</td>
<td></td>
<td>![Icon16]</td>
<td>Yes</td>
</tr>
<tr>
<td>Workplace</td>
<td></td>
<td>![Icon17]</td>
<td>Yes</td>
</tr>
<tr>
<td>Printer</td>
<td></td>
<td>![Icon18]</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 6.1. Initial categories and icons
6.4.2. Storing Category Data

All the category data are stored in a table called `categories`. There is a mandatory attribute called `category_id` in `icons` table which is a foreign key to `id` attribute in `category` table. That is how the icons and categories are related each other.

6.4.3. System Category

System category has the same concept as the system icon. It is a category that can’t be edited or deleted except by the system administrator.

6.5. Building Map on APMv3 Map Visualization

Maps are what the application needs after the icons and categories have been created and filled with some records. Map is an empty place or field where the map host along with their relation(s) to other map host(s) can be placed and later it will become a meaningful network information of a particular IT infrastructure. Figure 6.3 will show what map is.

![Creating/Editing a map to be a useful information](image)

Figure 6.3. A preview of how a map looks like

6.5.1. Storing Map Data

All the map data are stored in a table called `maps`. However only the overall map’s information that will be saved in this table (only save a map in empty states). The map creation and modification date are stored consecutively in `created_at` and `updated_at` attributes.

6.5.2. Map Size

Each map has a specific size depending on how big the map needs to be. The `width` and `height` attributes in `map` table will store the map’s size in pixel (px) units. The minimum width and height of the map are set to be 100px and there is no restriction for the maximum map size. The default map size when it is first created is adjusted to the computer’s resolution size.

6.5.3. Map Background

A map can have a background image or not. The background is set on the map by inserting it as a CSS `background-image` of the map HTML element without any repeat. The image URL is stored in the `urlBackground` attribute within the `map` table. The background images are named consecutively (like the icon images) and `last_filenaming` table is used to serve this purpose.

There is no size restriction of the background and the background will remain as its original size but it is recommended to use image smaller than 3MB, so the image upload would be fast. If the background is too small or big comparing to the map size, the background will not be stretched on the map.
6.5.4. Map Host

A map can have zero or many map hosts. A map which does not have any host is an empty map or a new map that has just been created. All the map host data are saved in a table called `map_details`.

The table has these attributes, along with their explanations: `id` that becomes the unique key of the corresponding map host, `map_id` that stores which map the host belongs to and at the same time acts as a foreign key to `maps` table, `icon_id` that stores which icon image the host uses and acts as a foreign key to `icons` table, `x` and `y` that stores the horizontal and vertical position relatively to the map, `host_id` that stores the host name from APM API, `label` that stores information displayed below the map host, `width` and `height` that stores the map host size, and `redirect_id` that indicates to which maps the host will be redirected if it is clicked.

6.5.5. Map Host Relation

A map host can have no relation or many relations to another map host. If there is a relation between two hosts in a map, then there will be a straight line drawn between those two hosts using JCanvas, a JavaScript plugin for HTML5 canvas. The relations data are stored in `maps_relations` table. Figure 6.5 will clearly illustrate the whole components of a map.
6.6. Building Asynchronous Backend Connection on APMv3 Map Visualization

Ideally, after the application has finished, all of the data connections between the application and the database have to be through APM API. However, the APM API is still being built now and it’s not possible to be used. At first, AMPv3 Map Visualizer used HTML, JavaScript, and PHP altogether, resulting in a very complex web application and later it would be hard to integrate it with APM API because it doesn’t use PHP. So, the author had decided to take all the PHP code to be a separated temporary API (backend) and the remaining code would be only HTML and JavaScript (frontend). The final application would be much easier to be migrated to the finished real API as the main advantage of building the temporary API. Figure 6.6 shows three possibilities in building the application backend.

![Figure 6.6. Three possible backend communications](image)

The temporary API will process all the server-side data and all of the application webpages are the frontends which will display the client-side information as interactively as possible. The only data ready to be used from APM API are the hosts data, so the application will use a JSON external file which is originated from APM API. After the APM API is ready to be used, the temporary API and the external JSON file will be migrated fully to APM API.

The frontend communicates with the backend using AJAX (Asynchronous JavaScript and XML). AJAX is asynchronous data communication protocol. With Ajax, web applications can send data to and retrieve from a server asynchronously (in the background) without interfering with the display and behavior of the existing page.

6.6.1. RESTful Architecture Approach

The AJAX connections will be built according to RESTful architecture which means the connections use a set of HTTP methods: GET for selecting data, POST for inserting data, PUT for updating data, and DELETE for deleting data.

6.6.2. Application Endpoints

Endpoint is the URL needed to be specified within the AJAX call in order to do a correct intended action. To give illustration, the endpoint to select maps is different with the endpoint
to select icons. The temporary API needs to use fifteen endpoints with different HTTP methods from the total of eight tables. Below are the list of all the endpoints along with their explanation.

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Endpoint</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>/visualization/activities/</td>
<td>Get the list of all activities in JSON</td>
</tr>
<tr>
<td>GET</td>
<td>/visualization/categories/</td>
<td>Get the list of all categories in JSON</td>
</tr>
<tr>
<td>GET</td>
<td>/visualization/icons/</td>
<td>Get the list of all icons in JSON</td>
</tr>
<tr>
<td>GET</td>
<td>/visualization/maps/</td>
<td>Get the list of all maps (without map hosts and relations) in JSON</td>
</tr>
<tr>
<td>GET</td>
<td>/visualization/maps/{id}</td>
<td>Get a specific map along with its map hosts and relations in JSON</td>
</tr>
<tr>
<td>POST</td>
<td>/visualization/categories/</td>
<td>Insert a new custom category</td>
</tr>
<tr>
<td>POST</td>
<td>/visualization/icons/</td>
<td>Insert a new custom icon</td>
</tr>
<tr>
<td>POST</td>
<td>/visualization/maps/</td>
<td>Insert a new map</td>
</tr>
<tr>
<td>POST</td>
<td>/visualization/images/</td>
<td>Upload a new image for a map background or an icon image depending on the value of the $_POST['imgtype'] variable</td>
</tr>
<tr>
<td>PUT</td>
<td>/visualization/categories/</td>
<td>Edit a specific category</td>
</tr>
<tr>
<td>PUT</td>
<td>/visualization/icons/</td>
<td>Edit a specific icon</td>
</tr>
<tr>
<td>PUT</td>
<td>/visualization/maps/</td>
<td>Edit a specific map along with its map hosts and relations</td>
</tr>
<tr>
<td>DELETE</td>
<td>/visualization/categories/</td>
<td>Delete a category</td>
</tr>
<tr>
<td>DELETE</td>
<td>/visualization/icons/</td>
<td>Delete an icon</td>
</tr>
<tr>
<td>DELETE</td>
<td>/visualization/maps/</td>
<td>Delete a map along with its map hosts and relations</td>
</tr>
</tbody>
</table>

Table 6.2. List of APMv3 Map Visualizer endpoints

6.6.3. Data Structure in JSON

All AJAX connections will return a JSON formatted data, even if the connections were meant to insert, update or delete from a table. The select operation will obviously return the specified table contents in JSON, but the insert, update and delete operations will return a notification in JSON whether the operation was successful or not. Figure 6.7 shows how the JSON data structures in map, category, and icon select operations look like. Same thing like Selecting Categories and Selecting Icons also applies on Selecting Activities.
The JSON data returned if a select operation is performed. They will only return JSON data like this:

```json
{
  'notif': 'success',
  'notif_title': 'Successful in Deleting a Category',
  'notif_message': 'A category has been deleted!' 
}
```

indicating that the delete category operation has succeeded. If the delete operation was failed for some reasons, then the notification would be like for example:

```json
{
  'notif': 'error',
  'notif_title': 'Error in Deleting a Category',
  'notif_message': 'There is still child icons in this parent category. (Delete the icons first)' 
}
```

The application frontend will receive all kinds of notification message and will display them in the client-side interface using PNotify.
6.6.4. Data Retrieval Method

How the map data are retrieved in the frontend page is by using both jQuery.getJSON() and jQuery.ajax() request methods. Both of them are AJAX request methods, but in APMv3 Map Visualizer they are both used because they have their own advantages and disadvantages. The jQuery.ajax() can be written like this:

```javascript
$.ajax({
  datatype: "json",
  method: the_method, // the_method is either GET, POST, PUT, or DELETE
  url: the_endpoint, // the_endpoint depends on what is needed to be retrieved
  success: function( the_result ){
    // display the results stored in the_result, i.e. map results
  },
  error: function( the_error ){
    // display the error stored in the_error variable
  }
});
```

and the jQuery.getJSON() is even simpler because it can be written like this:

```javascript
$.getJSON( the_endpoint, function( the_result ) {
  // only GET method that can be used
  // display the results OR store them in a variable
});
```

It can be clearly seen from the comparison of both the request methods above that the jQuery.getJSON() does not need many lines of codes. Another advantage of using jQuery.getJSON() is that you can actually store the result in a global JavaScript variable to be used later on. The jQuery.ajax() method returns a result which can only be used inside the function of the success handler due to the asynchronous nature of AJAX. So the result can only be displayed and later if we need to display we have to recall the jQuery.ajax() request again, causing more page loadings and a slow application. That is one disadvantage of using jQuery.ajax() request method.

However, there are at least two advantages of using the jQuery.ajax() method. First, unlike jQuery.getJSON() that can only use GET HTTP method, jQuery.ajax() can use all four HTTP methods (GET, POST, PUT, and DELETE). So, jQuery.ajax() will support the RESTful architecture which is already being built in APM API. It is possible to use GET HTTP method to serve POST, PUT, and DELETE purposes, but the application then will not have RESTful architecture anymore. The second advantage of using the jQuery.ajax() request method is that the error can be returned in a very simple way using the error setting from the jQuery.ajax() itself which is written above, `error: function( the_error ){}`.

Therefore in APMv3 Map Visualizer, all insert, update, and delete operations use jQuery.ajax() to maintain the application to be RESTful and the select operation usually uses jQuery.getJSON() because the returned results are usually needed to be used again later.

6.7. Building Activities Tracker on APMv3 Map Visualization

The application has a feature to log every single action that is performed. The logging function is in the backend and the activity data are stored in `activities` table. The table contains what the activity was, where and when it was done. The activity tracker is displayed in a separate webpage and user can sort the activities according to their table header (ascending or descending) using jQuery table sorter. Figure 6.9 shows how the activity tracker looks like.
Figure 6.9. Activity Tracker in APMv3 Map Visualizer.
Chapter 7: Map Editor and Map Viewer

7.1. Introduction

Map editor and map viewer are the two most important pages in APMv3 Map Visualizer. As the names already imply, map editor page is where a user can edit a map and map viewer page is where a user can view a map.

7.2. Map Editor

Map editor has a page layout such as Figure 7.1 below.

This page uses Bootstrap 3 JavaScript (like popover for the map options, tooltip for icon explanation, collapse for icon selection in side bar, and modal for displaying map menu) to support its functionality and user-friendliness. Below are some components of map editor.

7.2.1. Map Host

When map host is clicked, it will show a popover (Figure 7.2) to adjust or change its host data.

The Relate Host triggers a black line used to relate the specified map host to another map host. Delete Host button will delete the map host, Change Host Icon is used to change the map host icon to default.

April 2015, Fontys Graduation project, Hans Sebastian T.
host icon (e.g. printer icon), and Set Host Icon to Default is used to change the map host icon according to the default icon of the host and only appears active if the map host is not an iconless host. The up arrow button (↑) is used to set the host label exactly the same with the host name (copy the host name and paste it to host label). Every map host in a map has these data as its element attribute:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Usage</th>
<th>Table used</th>
</tr>
</thead>
<tbody>
<tr>
<td>data-x</td>
<td>Stores the horizontal position (x) of the map host relative to the map</td>
<td>map_details with x attribute</td>
</tr>
<tr>
<td>data-y</td>
<td>Stores the vertical position (y) of the map host relative to the map</td>
<td>map_details with y attribute</td>
</tr>
<tr>
<td>data-iconid</td>
<td>Stores the icon that is used for the map host</td>
<td>map_details with icon_id attribute as a foreign key of icons</td>
</tr>
<tr>
<td>data-hostid</td>
<td>Stores the attached host (entered from Host Name input)</td>
<td>map_details as a foreign key of the hosts.json file</td>
</tr>
<tr>
<td>data-redirectid</td>
<td>(optional) Stores a map id. If this host is clicked in the Map Viewer, it will navigate to the specified map</td>
<td>map_details with redirect_id attribute as a foreign key of maps</td>
</tr>
<tr>
<td>data-hostlabel</td>
<td>Stores the host label. The label is printed below the host itself</td>
<td>map_details with label attribute</td>
</tr>
<tr>
<td>data-defaulticon</td>
<td>Stores the default icon id of the map host. If there is no host attached, the default value is “no”</td>
<td>-</td>
</tr>
<tr>
<td>data-rawid</td>
<td>Stores the map host id (from map_details table)</td>
<td>map_details</td>
</tr>
<tr>
<td>data-tempid</td>
<td>Stores an incremental temporary id to be used later for saving purpose</td>
<td>-</td>
</tr>
<tr>
<td>width (saved in CSS)</td>
<td>Stores the width of the map host</td>
<td>map_details with width attribute</td>
</tr>
<tr>
<td>height (saved in CSS)</td>
<td>Stores the height of the map host</td>
<td>map_details with height attribute</td>
</tr>
</tbody>
</table>

Table 7.1. Map host data

Figure 7.3 shows some actions that can be done to a map host.

![Map Host](image)

Figure 7.3. Some possible actions for a map host

7.2.2. Side Bar

The side bar is placed at the left side of the Map Editor and is auto-hidden which means it can be hidden if it is not used. There are four things in the side bar: the navigation links, Map Menu button, View Map button, and the collapsible icon list sorted according to their category.
Bootstrap collapse is used to make the icon list collapsible and jQuery UI draggable is used to make the icon draggable and droppable.

7.2.3. **Top Bar**

Top bar contains two important text input, Icon Search input which enables users to search icons according to their name or description and Host Search input which enables users to search hosts according to their host name or host tags. Each host has its default icon, so the icon of a host that will appear as a search result is its default icon. Both the host and icon search features use Twitter Typeahead for their custom autocomplete.
In order to place one of the result to be used on the map as a map host, the users just need to drag and drop on the desired icon (Figure 7.7), just like what they have to do to add icon from the side bar icon list.

**7.2.4. Main Bar**

Main bar is the map itself. It contains all of the map hosts along with their relations. It can be resized using jQuery UI resizable. The map size will later follow the size of the main bar.

**7.2.5. Keyboard Press List**

Below is the list of all the keyboard presses used in the map editor. The keyboard shortcut is handled in jQuery by retrieving the `event.keyCode` as the keys pressed whenever the users press any keyboard key.
Table 7.2. Keyboard Press List

<table>
<thead>
<tr>
<th>Keys</th>
<th>Condition</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTRL+DEL</td>
<td>-</td>
<td>Toggle deletion state of map host relation(s). During the deletion state, all of the map host relations will turn red and when one of them is clicked, it will be deleted.</td>
</tr>
<tr>
<td>DEL</td>
<td>A map host has to be selected</td>
<td>Delete the selected map host</td>
</tr>
<tr>
<td>CTRL+S</td>
<td>-</td>
<td>Save the map. By default, a browser will show a save webpage window, but it is avoided by using event.preventDefault() JavaScript code.</td>
</tr>
<tr>
<td>SHIFT</td>
<td>A map host has to be moved</td>
<td>Straight-align the map host vertically or horizontally depending on the position of the mouse cursor</td>
</tr>
<tr>
<td>↑ (up)</td>
<td>A map host has to be moved</td>
<td>Move the map host up</td>
</tr>
<tr>
<td>→ (right)</td>
<td>A map host has to be selected</td>
<td>Move the map host to the right</td>
</tr>
<tr>
<td>↓ (down)</td>
<td>A map host has to be selected</td>
<td>Move the map host down</td>
</tr>
<tr>
<td>← (left)</td>
<td>A map host has to be selected</td>
<td>Move the map host to the left</td>
</tr>
</tbody>
</table>

7.3. Map Viewer

Map viewer shows the whole map along with its map hosts, map host relations, and host status of every map host.

It can be seen from Figure 7.8 that there is something called iconless host. Iconless host is just a normal map host, but it is invisible on the map viewer, so only the status and the map host label that will appear on the map. This map host is usually used for a physical map which uses background image. Figure 7.9 is an example of physical map.
The background will show the real photo of the hosts and it is definitely not a good idea to place a computer icon over the background. It is much better to display only the host statuses using iconless hosts attached to every hosts on the background photo. There are five possible host statuses. All of the statuses are retrieved from hostresources.json which came from APM API. Figure 7.10 shows how each of the host status looks like.

The unknown status indicates that the map host doesn’t have any host attached and the pending status shows when the host was first created (the status does not last long).
Chapter 8: Project Functional Testing

Functional test has been done with three test items: Map management, Category and Icon Management, and Activity Viewer. There were four people from APM team performing this test. The reason to choose APM team is because they know APM better than others. All the necessary solutions have been implemented. Figure 8.1 shows the test summaries.

Map Management

<table>
<thead>
<tr>
<th>Test Cases</th>
<th>Expected Result</th>
<th>Actual Result / Comments</th>
<th>Checked</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add a new map</td>
<td>A new map is created with a successful notification and the page redirects to Map editor to directly edit the map</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Edit a map:</td>
<td></td>
<td>No map name found. Solution: Print the map name. There is no confirmation to save the map after leaving/closing the page. Solution: give confirmation</td>
<td>v</td>
</tr>
<tr>
<td>Resize the map</td>
<td>The map size gets smaller/bigger</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Search and add an icon “computer”</td>
<td>Computer map host is added on the map</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Search and add a host “api003.apm.test”</td>
<td>A map host is added on the map with host api003.apm.test attached in it</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Open side bar</td>
<td>Map side bar is opened</td>
<td>Quite hard to find. Solution: User Manual. Should dissapear when clicked on map hosts as well. Solution: apply that</td>
<td>v</td>
</tr>
<tr>
<td>Add an icon from side bar</td>
<td>A map host is added on the map</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Resize map host</td>
<td>Map host gets smaller/bigger</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Relate the map host to another map host</td>
<td>Map host is related each other</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Delete the map host</td>
<td>Map host is deleted</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Change map host details</td>
<td>Map host information is updated</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Show map menu</td>
<td>Map menu is shown</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Edit the map information</td>
<td>Map name/description is changed</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Upload a background</td>
<td>A new background is uploaded and set as a map background</td>
<td>Unable to save new image &gt; 3MB. Solution: gives size limit on map background</td>
<td>v</td>
</tr>
<tr>
<td>Remove a background</td>
<td>Map background is removed</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Save map</td>
<td>Map is saved</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Build a sample map according to the sample given and save the map</td>
<td>A sample map is built</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Use keyboard shortcuts:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Move Map host (←→↑↓)</td>
<td>Map host is moved</td>
<td>Not visible which icon is selected. Solution: make a brightness differences on selected map host</td>
<td>v</td>
</tr>
<tr>
<td>Delete relation (CTRL+DEL)</td>
<td>Relation(s) are deleted</td>
<td>Hard to click. Solution: make relation line thicker. No confirmation on deletion.</td>
<td>v</td>
</tr>
</tbody>
</table>
### Category and Icon Management

<table>
<thead>
<tr>
<th>Test Cases</th>
<th>Expected Result</th>
<th>Actual Result</th>
<th>Checked</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add a new category &amp; Edit the category</td>
<td>A new category is added</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Add a new icon and attach it to the new category</td>
<td>A new icon is added</td>
<td>&quot;Stored in&quot; notification is not needed. Solution: Delete it. No image resolution check? Resolution check will be done on the APM API as a later backend</td>
<td>v</td>
</tr>
<tr>
<td>Edit the created icon</td>
<td>The map size gets smaller/bigger</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Delete the created icon</td>
<td>Computer map host is added on the map</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>View the icon in the map editor</td>
<td>A map host is added on the map with host api003.apm.test attached in it</td>
<td>ok</td>
<td>v</td>
</tr>
</tbody>
</table>

Average of total time spent: 5 minutes

### Activity Viewer

<table>
<thead>
<tr>
<th>Test Cases</th>
<th>Expected Result</th>
<th>Actual Result</th>
<th>Checked</th>
</tr>
</thead>
<tbody>
<tr>
<td>View the activity window</td>
<td>Activity window is opened</td>
<td>Seeing the ID's instead of name. Name will not be displayed because ID is the only unique attribute. Name could possibly be retrieved but it's not possible for items that has been deleted.</td>
<td>v</td>
</tr>
<tr>
<td>Sort the activities by one of the table attributes</td>
<td>Activities are sorted</td>
<td>Activities are sorted, but the ID is considered as string, so it will be sorted like this: ID: 1, ID: 11, ID: 2, ID: 21; ID: 3,... Solution: add the table name as the ID separator</td>
<td>v</td>
</tr>
</tbody>
</table>

Average of total time spent: 2 minutes

Figure 8.1. Functional test summaries
Chapter 9: Conclusions and Recommendations

All mandatory requirements of the application have been fulfilled. An optional requirement to give authentication and authorization features within the final application is not done because the process had to follow APMv3 user data (retrieved from APM API) and the API is still being built right now, so it’s not possible to complete this requirement. Thus, the use case diagram is also discarded because it explains the authentication and authorization part.

APMv3 that is still developed will later use the application that the author has developed called APMv3 Map Visualizer for its map visualization feature. Through some researches, the application was decided to be built from scratch. It is a web-based application, which has separate frontend (written in HTML and JavaScript) and separate RESTful backend (written in PHP). All necessary data within this application are exchanged asynchronously (in and out) using AJAX method.

APMv3 Map Visualizer can overcome the current map visualization’s problem which requires APM users to recreate the image-based map whenever a change occurs in the map because the application can create the map itself directly on the webpage and it does not need any third party software like Visio to design the map. Compared to NagVis, which is used in APMv2, the final application has two major advantages such as, it is fully flexible to be integrated with APM API (NagVis needs a huge customizations to integrate it with APM API) and it uses separate frontend and API will be the only backend, which means no need to install any webservice (NagVis needs to install PHP).

Further recommendation would be completing the application with authentication and authorization features after it has been integrated with APM API and implementing security within the application.
**Evaluation**

At first, I was not a very good frontend programmer and knowing that I had to deal such an assignment which requires a good client-side (JavaScript) programming skill, I was challenged to learn more and give extra efforts on it. JavaScript was not really my expert skill and I was better at PHP (backend). As a result, I learned a lot of JavaScript techniques that I haven’t known before and I was introduced many useful JavaScript plugins. Now I’m thinking reversely that frontend development might be my future career because it combines programming with art. I also feel like I’m blessed with a good design/art skill, therefore frontend development might be very suitable for me. I learned that a good software requires a programmer to have a knowledge in art as well, so the software will be as user-friendly as possible.

I experienced some real challenges that could possibly happen in IT career. One of them is a project change. There have been some changes during the project, so some items in the project plan had to be modified to adjust the later situation. One of the biggest change is the programming language I used. At first, the application used HTML, JavaScript and PHP altogether as a whole single application. JavaScript served the application’s functionality and interactivity as a client-side programming language and PHP served the server-side matters including SQL, session management, and form validations or submissions. The result was good and it was as expected because all the functionalities worked well. After that, something came in mind as my company tutor also suggested, that the application could have been improved to be easily adjusted and integrated into APMv3 application by using separate frontend and backend. APMv3 application only uses HTML and JavaScript and in order to connect to the database or retrieve any data, the frontend sends AJAX requests to the API and it will return the requested items in JSON. So, the Map Visualization could have been better if it had HTML and JavaScript as a separate frontend and PHP as a separate backend because then it would be suitable with current APMv3 application and APM API. This was a major change and quite a big challenge for me. Then, I had decided to change over the programming language and rewrite the application to have a separate frontend and backend.

I could say the internship experience has met my personal expectations and I would recommend this company for graduate students to have an internship here.
References/Literature List


http://en.wikipedia.org/wiki/Configuration_file


Appendix A: Research Report
Research Report: Table of Contents

Introduction ........................................................................................................................................ III

Research 1: Project Initiation Research ......................................................................................... IV
1.1. Introduction ............................................................................................................................... IV
1.2. The Possibility to Improve NagVis ............................................................................................ IV
1.3. The Possibility to Use Another Visualization Tool ............................................................... IX
1.4. Creating a Self-Made Map Visualization .................................................................................. X

Research 2: General Information ................................................................................................... XI
2.1. Introduction ............................................................................................................................... XI
2.2. Nagios ..................................................................................................................................... XI
2.3. Nagios Visualization (NagVis) ............................................................................................... XI

Research 3: Plugins or Frameworks Used ....................................................................................... XII
3.1. Introduction ............................................................................................................................... XII
3.2. jQuery ...................................................................................................................................... XII
3.3. jQuery UI ................................................................................................................................. XII
3.4. Bootstrap 3 .............................................................................................................................. XII
3.5. jCanvas .................................................................................................................................... XIII
3.6. Twitter Typeahead .................................................................................................................. XIII
3.7. PNotify .................................................................................................................................... XIII
3.8. Font Awesome ...................................................................................................................... XIII

Research Report: References .......................................................................................................... XIV
**Introduction**

This research report contains all the researches that have been done in detail during the whole project. The research can be considered as applied research which use existing knowledge to solve problems in practice. The research is based on DOT framework which has five research strategies: Field, Library, Workshop, Laboratory and Showroom. The author was introduced and taught the DOT framework by Fontys University of Applied Science.

The main research question is: “What is the most efficient and user-friendly way to build Map Visualization that suits APMv3?”. Below are several sub-questions for that main research question.

**Project initiation:**
- How does NagVis work? – Library
- Are there any other available map visualization tools? – Library
- What is APMv2 and APMv3? – Field
- What is the APM API? – Field

**Building an efficient and user-friendly Map Visualization:**
- What is the most efficient and user-friendly way to build a visualization tool for APMv3? What are the criteria to choose the best one? – Field
- How are map hosts for map visualization tool designed and developed? – Workshop, Field
- How are map relations for map visualization tool designed and developed? – Workshop, Filed
- What are the plugins that can be suitably used for APMv3 map visualization? – Library

**Formulating and executing test plans:**
- What are the test items? Who are the test subjects? – Laboratory
- How are the test items executed? – Laboratory
- What are the advantages of the final product compared to other map visualization products? – Showroom

All the Library and Showroom research were performed by searching from the internet, the Laboratory research was performed by doing a functional test of the application (the test was performed by three people in APM team because they know APM background/environment better than other sub-departments), the Field research was performed by doing several conversations with colleagues and company tutor and also progress meeting with company tutor, and the Workshop research was basically done by coding the application.
Research 1: Project Initiation Research

1.1. Introduction

This research is written to give a broad overview of the researches that have been performed regarding the Project Initiation.

Previously, a big decision has to be made, either to stay using NagVis with an object oriented improvement, to find another possible tool besides NagVis, or to build a working visualization tool from scratch to be applied in APMv3.

1.2. The Possibility to Improve NagVis

NagVis is used in APMv2 and it gives many advantages. It works really well with Nagios, since it is Nagios proprietary. It is free and open source, therefore everyone can actually change, optimize, or edit it. Even to reduce some of its functionality, they are all possible. The author has downloaded the source code from NagVis official website, http://www.nagvis.org/downloads. Source code investigation has been performed in order to know for sure the possibility of improving NagVis to become more like object oriented tool.

1.2.1. NagVis Folders and Files Structure

The overall file amount of the whole NagVis project is 633 files in 78 folders. Each folder contains some subfolders and the subfolders are highly possible to contain other subfolders. The maximum folder’s depth level in NagVis project is 7, which can be found in NagVis Project Folder/frontend/nagvis-js/ext/dwoo-1.1.0/plugins/builtin/blocks/section.php (besides section.php, there are also 20 other files in this folders). Those files are too deep and might not be needed to improve NagVis, but it is in NagVis project and it would be used somehow by NagVis.

Figure 1.1. A preview of one of NagVis file
1.2.2. NagVis Backend Classes

NagVis has frontend and backend classes that are used as the main core of the application. There are 19 frontend classes and 85 backend classes listed in Figure 1.2 below. The backend classes are NagVis important parts. Each class contains several functions and the amount of function is various. It can be up to 15 functions in one class or possibly only 2 functions in one class.

1.2.3. NagVis Frontend Classes

If the author wants to improve NagVis, the frontend classes need to be modified. For instance, if a new frontend feature like host drag and drop (which the company wishes to have) wants to be added, the author has to deal with NagVis frontend classes.

NagVis uses PHP language a lot and even its front end codes are written in PHP, neither in HTML, nor in JavaScript. It uses PHP variables to store all of the front-end codes and NagVis prints (echo) those variables in certain places when they are needed. Figure 1.3 shows how the front end codes are stored in PHP variable in a specific PHP function.

In Figure 1.3, the frontend codes are written in an implicit PHP object variable such as, $this->type. It means, in order to understand the meaning, purpose, and usability of the function, the reader must understand the whole workflow of NagVis application which is nearly impossible for someone who does not write the function himself or does not contribute in writing other related functions in NagVis frontend codes.
1.2.4. **NagVis Commentary**

One important aspect to understand a code written by other programmers is the code comments. Code comments are the only easily readable parts of all codes in common. In some NagVis functions, there are comments available, but in some other functions there are no comments, like shown in Figure 1.4 (found in NagVis Project Folder/frontend/nagvis-js/js/nagvis.js). Those make the functions even more difficult to understand.

![Figure 1.4. A preview of NagVis function with and without comments](image)

1.2.5. **NagVis Data Storing**

After looking through how NagVis retrieves and saves its data, NagVis actually does not use any database. However, it still needs to save its data and the data saving method turns out to be using a config file (.cfg).

Config files configure the initial settings for some computer programs. They are used for user applications, server processes and operating system settings. The files are often written in ASCII (rarely UTF-8) and line-oriented, with lines terminated by a newline or carriage return/line feed pair, depending on the operating system. They may be considered a simple database.[5]
APM team retrieves data via API and the API queries the data from separate MySQL database. So, APMv3 uses MySQL database in separate server and the Map Visualization of APMv3 should stores its map data in the database, instead of saving it in local config file (.cfg) like what NagVis has right now. If the author wants to change the way NagVis stores its map data from config file to a remote database, it would be a major change and it can’t be 100% successful guaranteed.

1.2.6. NagVis Iconset

Iconset is a set of image icons showing the status of a host or service in NagVis. NagVis has icon status in many formats like shown below in Figure 1.5.

![NagVis Iconset Lists](image)

Figure 1.5. A preview of NagVis iconset lists

It is desired in APMv3 to have not only just an icon, but also a whole host (a computer image with an icon), so that the host does not have to be in image based format and just in the form of NagVis map background. Figure 1.6 shows exactly what the company desires in APMv3. The computer actually exists as a host and draggable, which means that the computer can be moved all around the map. That makes APM user can edit the map easily when a change occurs. The computer should not be just previewed as a map background and then stacked an icon on top of it.

![APM Map Visualizer](image)

Figure 1.6. A preview of APMv3 desired host

NagVis actually can accomplish that by making a set of icon (an iconset) with the computer in as the host image and save it one by one per icon status like shown in Figure 1.7. They can be called a custom iconset because the icons were made by the user himself. There are some NagVis standard iconsets and if the user does not want to use them, they can change the iconset to their custom self-made iconset.
The author had tried creating one custom iconset and it worked perfectly like what the company desires. Unfortunately, to make an iconset, he had to create exactly 22 images showing each status of the host. If he would make another iconset (e.g. router or switch iconset), he had to create exactly other 22 images showing each status of the host. This is really not effective and takes a lot of storage as 22 images have to be existed to represent only one host. If the company for example needs 50 hosts, then they will end up having 1100 .PNG images in their server.

### 1.2.7. NagVis Language Selection

NagVis is a multilingual application, which means it is actually designed to serve many language selections. The language selections that are offered by NagVis are English (en_US), Germany (de_DE), French (fr_FR) and Brazilian Portuguese (pt_BR). This is one of the features and advantages of using this tool, but then the source code becomes more complex as the functions to pick language are also added within the application. This confuses anyone who does not understand the source code as a whole and does not know the difference between language codes and functional codes.

### 1.2.8. NagVis for Enterprise

NagVis is a free software, but it turns out that the NagVis developers offer everyone who wants to extend or costumize it, which should cost money. There is a page in NagVis official website, containing where to contact the developers to ask for a feature extension and why NagVis users should trust them for feature extension ([http://www.nagvis.org/enterprise-support](http://www.nagvis.org/enterprise-support))
In Figure 1.8, it can be implicitly seen that NagVis application which is offered free has only standard basic features, which can fit to all general purpose. If anyone wants to customize it, they should contact the developers instead of customizing it on their own because they will find difficulties in understanding the code.

1.2.9. Conclusion

Although NagVis is freely editable, making some modifications in its code is not as easy as what the author thought. A very good understanding of its major functions is needed in order to add or edit its features. If someone wants to change NagVis code, they have to understand the application as a whole, not only understanding one function, but also understanding all the related functions.

In short, NagVis contains over complicated functions and it is nearly impossible to modify NagVis codes, thus the author has decided not to use NagVis anymore for the Map Visualization in APMv3.

1.3. The Possibility to Use Another Visualization Tool

There is another monitoring tool available called Zabbix and it even has a built-in Map Visualization functionality. So, Nagios needs to use NagVis as its plugin, but Zabbix does not need to use any plugin, because Zabbix itself already has Map Visualization functionality. There is a review states that Zabbix Map Visualization functionality is better than NagVis. The reviewer is a Nagios user who had migrated to Zabbix because of the map visualization in Zabbix is much better. He said, “We're currently monitoring over 80 hosts and 200 services with Nagios and it's been working great. But, I'm migrating to Zabbix.” [7]. Then he explained, “Another strength on Zabbix is the network map feature – it's a nightmare to create a useful map of more than 20 hosts in Nagios.” [7] The map visualization can be seen in Figure 1.9.
However, APMv3 uses Nagios and it has its own data structure. Zabbix Map could not be just taken out and used for Nagios. Both Nagios and Zabbix have their own environment, even though they are both monitoring tools. The choice is to use completely Zabbix and leave Nagios or to use Nagios without Zabbix Map Visualization functionality. Nagios had been a main and central tool for APMv3, therefore, Zabbix Map could not be used even if it is well-built and considered as a good Map Visualization tool.

1.4. Creating a Self-Made Map Visualization

The author had to consider to choose this option even if he has a principle: “Don’t create a tool yourself, use the one that is available.” Creating such a new tool requires a lot of works and if there is a tool freely provided, one should not create himself. However, considering the limited time to study NagVis, creating a self-made map visualization appeared to be the best alternative to be used for APMv3. The advantages of creating a self-made map visualization tool are:
- The tool is fully flexible, no need to follow some tool regulations or functions,
- The tool can be adjusted to be completely suitable to APMv3 and can exactly fulfill what APMv3 needs,
- All kind of features can be easily added,
- No need to study an available visualization tool and the way it works,
- At the end, the tool creator can master his tool, which means he will understand the tool program as a whole.

The disadvantages of creating a self-made map visualization tool are:
- A lot of researches has to be done regarding what frameworks should be used and how to build the tool,
- It will be a lot works since the tool is built from zero,
- There will be some risks that the tool is not working properly or there are some bugs, because this new freshly-built tool can’t be compared to a mature visualization tool that has been through a lot of testing and mature like NagVis.

Figure 1.9. A preview of Zabbix Map Visualization functionality. [6]
Research 2: General Information

2.1. Introduction

This research was performed in order to find out some necessary information that would support the project initiation phase.

2.2. Nagios

Nagios is a well-known open source monitoring tool that has been used by many people since March 14, 1999 [3] (Nagios initial release date). The main objects to be monitored by Nagios are computer system, network and IT infrastructure. Nagios offers monitoring and alerting services for servers, switches, applications, and services. It alerts the users when things go wrong and alerts them a second time when the problem has been resolved. Nagios, originally created under the name NetSaint, was written and is currently maintained by Ethan Galstad along with a group of developers who are actively maintaining both the official and unofficial plugins. [3]

![Nagios logo](image)

Figure 2.1. Nagios logo

Nagios can be considered as a powerful monitoring tool, not only because it is a well-developed and mature system itself, but also it has all of these capabilities [2]:
- Monitoring the entire IT infrastructure of its user
- Spotting problems before they occur
- Letting the users know immediately when problems arise
- Sharing availability data with stakeholders
- Detecting security breaches
- Helping its users to plan and budget for IT upgrades
- Helping its users to reduce downtime and business losses

2.3. Nagios Visualization (NagVis)

![NagVis logo](image)

Figure 2.2. NagVis logo

NagVis is a visualization tool made for Nagios. Its data structure, functions, and codes are fully suitable with Nagios, so the users of this tool need to install Nagios in order to use this visualization tool. As a Nagios plugin, Surely NagVis was released after Nagios, exactly in April 30th, 20014. NagVis is mostly written in PHP (60.4%), the rests are in HTML (16.2%), JavaScript (18.9%), and Shell script and CSS (4.6%) [4]. Right now, the current stable version of NagVis is version 1.8.
Research 3: Plugins or Frameworks Used

3.1. Introduction
This chapter is written to know the possibility of using some external plugins. The application will use some JavaScript plugins which help the frontend to be as user-friendly as possible. Most plugins used are JavaScript plugins, only the Bootstrap 3 is both JavaScript and CSS plugins.

3.2. jQuery

![jQuery logo](image)

Figure 3.1. jQuery logo [8]

jQuery is a cross-platform JavaScript library designed to simplify the client-side scripting of HTML. jQuery is the most popular JavaScript library in use today. jQuery is free, open-source software licensed under the MIT License. [8]

In APMv3 Map Visualizer, jQuery will be used for almost all JavaScript operation because it can simplify most JavaScript codes.

3.3. jQuery UI

![jQuery UI logo](image)

Figure 3.2. jQuery UI logo [9]

jQuery UI is a collection of GUI widgets, animated visual effects, and themes implemented with Query (a JavaScript library), Cascading Style Sheets, and HTML. Both jQuery and jQuery UI are free and open-source software distributed by the jQuery Foundation under the MIT License; jQuery UI was first published in September 2007. [9]

jQuery UI offers three main facilities called Interactions, Widgets, and Effects. Each facility has many sub-facilities or features. Interactions contain Draggable, Droppable, Resizable, Selectable, and Sortable. For APMv3, the used interactions are Draggable, Droppable, and Resizable. Widgets, Effects, and Utilities are not used for APMv3.

3.4. Bootstrap 3

![Bootstrap logo](image)

Figure 3.3. Bootstrap logo [16]

Bootstrap is CSS as well as JavaScript framework that can help programmers creating a good frontend interface. It contains HTML- and CSS-based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions. The bootstrap framework aims to ease web development. [15]
3.5. jCanvas

jCanvas is a jQuery plugin that makes the HTML5 canvas easy to work with. jCanvas is a JavaScript library, written using jQuery and for jQuery, that wraps around the HTML5 canvas API, adding new features and capabilities, many of which are customizable. Capabilities include layers, events, drag-and-drop, animation, and much more. The result is a flexible API wrapped up in a sugary, jQuery-esque syntax that brings power and ease to the HTML5 canvas. [11]

3.6. Twitter Typeahead

Twitter Typeahead is a plugin to create an input autocomplete that would be used for host search and icon search in the Map Editor. The typeahead.js library consists of 2 components: the suggestion engine, Bloodhound, and the UI view, Typeahead. Bloodhound is robust, flexible, and offers advanced functionalities such as prefetching, intelligent caching, fast lookups, and backfilling with remote data. [17] The Bloodhound is not used in APMv3 Map Visualizer, instead the application uses a user-defined variable containing a function. The name of the variable was named as SubstringMatcher.

3.7. PNotify

PNotify is a JavaScript notification plugin, developed by SciActive. Formerly known as Pines Notify. It is designed to provide an unparalleled level of flexibility, while still being very easy to implement and use. PNotify provides desktop notifications based on the Web Notifications Draft. If desktop notifications are not available or not allowed, PNotify will fall back to displaying the notice as a regular, in-browser notice. [18]

3.8. Font Awesome

![Font Awesome logo](image)  
Figure 3.4. Font Awesome logo [19]

Font Awesome is a plugin that is used to generate custom icons in a form of a font. In APMv3 Map Visualizer, Font Awesome is used for the animated loading spinner, which is one of the Font Awesome icon.
Research Report: References

[12] Host icon images, images are taken from Microsoft Visio
Appendix B: Design Documents
1. **Entity Relationship Diagram (ERD)**

Entity Relationship Diagram can be defined as a data model for describing the data or information aspects of a business domain or its process requirements, in an abstract way that lends itself to ultimately being implemented in a database such as a relational database. [1] The ERD type that is used in Figure 1.1. below is a Crow’s Feet ERD.

For APM Map Visualizer, the ERD contains all the data structures that are stored in a MySQL database. This ERD will explain the relation between one table to another.

Figure 1.1: Entity Relationship Diagram of APM Map Visualizer

**Tables Description:**

- **maps**
  
  This table stores maps data. However only the overall map’s information that will be saved in this table (only save the map in empty states, without any hosts or we can say that the map in this table doesn’t contain anything). It only saves the size, the background, map description, when the map was created/updated, and who created the map. The hosts (computers, servers, devices) will be saved in another table called maps_details.
- **maps_details**
  This is the table that stores the maps’ hosts, along with their x and y position in their map, which icon that is used, and which host name (real host) that is used. This table can be categorized as the child table of the maps.

- **maps_relations**
  This table stores the relations of two hosts. If there is a relation between two hosts in a map, then there will be a straight line drawn between those two hosts. The hosts refer to the maps_details table, so this table can be categorized as the child table of the maps_details.

- **icons**
  This table stores the icon’s image that is being used for the hosts. Each host must have exactly one icon image, so all the hosts in a map are querying icon information from this table. Figure 1.2. shows a clear example of how the icon looks alike.

![Figure 1.2. Examples of icon image that are used in APM Map Visualizer. [2]](image)

- **categories**
  Each hosts icon is categorized in a category. This table stores the category information of the host icons.

![Figure 1.3. An example image explaining what category is. [2]](image)

- **hosts**
  Hosts table contains the information of real host that are being used, either they are owned by Acknowledge or by clients.

- **users**
  The users table contains the information of APM Map Visualizer’s users. This table is used for authentication and authorization purposes.

- **activities**
  This is the log table, which means this table stores all the activities done by certain user in a certain period of time.

- **last_filenaming**
  In Figure 1.1., this table has no relation at all with other tables. This table does not contain any map records related information. This table basically just stores the last host icon’s file name in the image directory. The file naming uses auto increment.
approach, which means the file will be named such as 1.png, 2.png, 3.png, etc. This table stores the latest number that is being used for the file naming.

2. Class UML Diagram
This Class Diagram of APM Map Visualizer shown in Figure 2.1. helps to better understand the structure of the whole application, the components in the application and the relation between each component. In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system’s classes, their attributes, operations (or methods), and the relationships among objects. [3]

![APM Map Visualizer Class UML Diagram](image)

Figure 2.1. Class UML Diagram of APM Map Visualizer

3. Use Case Diagram
Use case diagram of APM Map Visualizer gives broader overview of all the functionalities in the application and the users that are given the authorization in using the application. The users are divided into several type of users with their own role(s).

One thing to remember is that APM Map Visualizer is a sub-application of APMv3, so the users and the user roles of APM Map Visualizer have to follow the current users from APMv3.

In software and systems engineering, a use case is a list of steps, typically defining interactions between a role and a system, to achieve a goal. The actor can be a human, an external system, or time. [4]
4. **Application Interface**

This point will show the display interface of APM Map Visualizer. This should be the desired interface, but later the display interface can change depending on the future situations and needs.

---

**Main Menu:**

Main menu describes all the links to another page. This main menu is for the System Administrator. A normal user should not have a main menu of Map Visualizer, but they will have their own main menu in APMv3 application.
Maps Manager:

This is the page where system and map administrator can manage maps (create, edit and delete maps).

<table>
<thead>
<tr>
<th>ID</th>
<th>Map Name</th>
<th>Size</th>
<th>Background</th>
<th>Last Update</th>
<th>Description</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Server Room</td>
<td>1316x1024</td>
<td></td>
<td>2015-02-27 11:00:40</td>
<td>This map is the physical network of the server room</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Acknowledge Network map</td>
<td>1638x652</td>
<td></td>
<td>2015-02-27 12:14:20</td>
<td>This is acknowledge internal network</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Pentagonal</td>
<td>732x756</td>
<td></td>
<td>2015-02-03 11:46:34</td>
<td>Pentagonal map</td>
<td></td>
</tr>
</tbody>
</table>
The map editor page is the main and central page of APM Map Visualizer where a map is modified. This page is the only page that directly interacts with hosts. This page uses drag and drop functionalities to ensure the user-friendliness when someone wants to edit a map.
Figure 4.5. Viewing a map with their hosts’ status

This is the view map page where normal user should have access to view it. This map shows the status of the hosts and the status data are retrieved directly from APM API.

Icon Manager:

This page contains all of available icons. Only system administrator and map administrator that can open this page. System administrator(s) is able to manage all of the icons. Meanwhile, map administrator(s) can only manage certain icons which are made by them.

Figure 4.6. The main page of icon manager
**Category Manager:**

This page maintains all the host categories. All hosts must have exactly one category. One category can have no host or many hosts. Figure 4.8. shows the main page of category manager.
Figure 4.9. Main page of category manager

Figure 4.10. Adding a category in category manager

Figure 4.11. Editing a category in category manager
Design Document: References

[2] Host icon images, images are taken from Microsoft Visio
Appendix C: Test Report
Introduction

This document is the test report and contains APMv3 Map Visualizer testing. The type of the application testing is functional test and the purpose of the test is to ensure all the functionalities and features within the application work well as what are expected before. The test will be performed by APM team, consisting of four people. The reason of choosing APM team as the test subjects is because they have a very good knowledge of APMv2 and APMv3 and they know what are best for APMv3 map visualization.
Functional Test Plan

Map management:

- **Expected time estimation**: 10 minutes
- **Pre-condition**: Main menu is opened and webservice is already on.

**Test plans:**
- Add a new map
- Edit a map:
  - Resize the map
  - Search and add an icon “computer”
  - Search and add a host “api003.apm.test”
  - Open side bar:
    - Add an icon from side bar
  - Resize map host
  - Relate the map host to another map host
  - Delete the map host
  - Change map host details
  - Show map menu:
    - Edit the map information
    - Upload a background
    - Remove a background
  - Save map
- Build a sample map and save it:
  - Use keyboard shortcut to: Move map host (← → ↑ ↓), Save map (CTRL+S), Delete relation(s) (CTRL+DEL)
  - Delete the map

---

Category & Icon management:

- **Expected time estimation**: 5 minutes
- **Pre-condition**: Main menu is opened and webservice is already on.

**Test plans:**
- Add a new category & Edit the category
- Add a new icon
- Edit the icon

XXVIII
- Delete icon
- View the icon in the map editor

**Activity management:**

*Expected time estimation*: 2 minutes

*Pre-condition*: Main menu is opened and webservice is already on.

*Test plans*:
- View the activity window
- Sort the activity according to one of the table attribute (for example according to the date, etc).
## Functional Test Report

### Map Management

<table>
<thead>
<tr>
<th>Test Cases</th>
<th>Expected Result</th>
<th>Actual Result / Comments</th>
<th>Checked</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add a new map</td>
<td>A new map is created with a successful notification and the page redirects to Map editor to directly edit the map</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Edit a map:</td>
<td></td>
<td>No map name found. <strong>Solution:</strong> Print the map name. There is no confirmation to save the map after leaving/closing the page. <strong>Solution:</strong> give confirmation</td>
<td>v</td>
</tr>
<tr>
<td>Resize the map</td>
<td>The map size gets smaller/bigger</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Search and add an icon “computer”</td>
<td>Computer map host is added on the map</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Search and add a host “api003.apm.test”</td>
<td>A map host is added on the map with host api003.apm.test attached in it</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Open side bar</td>
<td>Map side bar is opened</td>
<td>Quite hard to find. <strong>Solution:</strong> User Manual. Should dissapear when clicked on map hosts as well. <strong>Solution:</strong> apply that</td>
<td>v</td>
</tr>
<tr>
<td>Add an icon from side bar</td>
<td>A map host is added on the map</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Resize map host</td>
<td>Map host gets smaller/bigger</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Relate the map host to another map host</td>
<td>Map host is related each other</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Delete the map host</td>
<td>Map host is deleted</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Change map host details</td>
<td>Map host information is updated</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Show map menu</td>
<td>Map menu is shown</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Edit the map information</td>
<td>Map name/description is changed</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Upload a background</td>
<td>A new background is uploaded and set as a map background</td>
<td>Unable to save new image &gt; 3MB. <strong>Solution:</strong> gives size limit on map background</td>
<td>v</td>
</tr>
<tr>
<td>Remove a background</td>
<td>Map background is removed</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Save map</td>
<td>Map is saved</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Build a sample map according to the sample given and save the map</td>
<td>A sample map is built</td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Use keyboard shortcuts:</td>
<td></td>
<td>ok</td>
<td>v</td>
</tr>
<tr>
<td>Move Map host (← → ↑ ↓)</td>
<td>Map host is moved</td>
<td>Not visible which icon is selected. <strong>Solution:</strong> make a brightness differences on selected map host</td>
<td>v</td>
</tr>
<tr>
<td>Delete relation (CTRL+DEL)</td>
<td>Relation(s) are deleted</td>
<td>Hard to click. <strong>Solution:</strong> make relation line thicker. No confirmation on deletion.</td>
<td>v</td>
</tr>
</tbody>
</table>
### Save map (CTRL+S)
- **Map is saved**
  - **Expected Result:** ok
  - **Checked:** ✓

### View the sample map
- **The map is viewed and shows the correct information**
  - **Expected Result:** ok
  - **Checked:** ✓

### Delete the map
- **The map is deleted**
  - **Expected Result:** A small bug, didn’t delete the map until 2nd trial. **Solution:** recheck the code
  - **Checked:** ✓

**Average of total time spent:** 15-20 minutes

### Category and Icon Management

<table>
<thead>
<tr>
<th>Test Cases</th>
<th>Expected Result</th>
<th>Actual Result</th>
<th>Checked</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add a new category &amp; Edit the category</td>
<td>A new category is added</td>
<td>&quot;Stored in&quot; notification is not needed. <strong>Solution:</strong> Delete it. No image resolution check? Resolution check will be done on the APM API as a later backend</td>
<td>✓</td>
</tr>
<tr>
<td>Edit the created icon</td>
<td>The map size gets smaller/bigger</td>
<td>ok</td>
<td>✓</td>
</tr>
<tr>
<td>Delete the created icon</td>
<td>Computer map host is added on the map</td>
<td>ok</td>
<td>✓</td>
</tr>
<tr>
<td>View the icon in the map editor</td>
<td>A map host is added on the map with host api003.apm.test attached in it</td>
<td>ok</td>
<td>✓</td>
</tr>
</tbody>
</table>

**Average of total time spent:** 5 minutes

### Activity Viewer

<table>
<thead>
<tr>
<th>Test Cases</th>
<th>Expected Result</th>
<th>Actual Result</th>
<th>Checked</th>
</tr>
</thead>
<tbody>
<tr>
<td>View the activity window</td>
<td>Activity window is opened</td>
<td>Seeing the ID's instead of name. **Name will not be displayed because ID is the only unique attribute. Name could possibly be retrieved but it’s not possible for items that has been deleted.</td>
<td>✓</td>
</tr>
<tr>
<td>Sort the activities by one of the table attributes</td>
<td>Activities are sorted</td>
<td>Activities are sorted, but the ID is considered as string, so it will be sorted like this: ID: 1, ID: 11, ID: 2, ID: 21, ID: 3,... <strong>Solution:</strong> add the table name as the ID separator</td>
<td>✓</td>
</tr>
</tbody>
</table>

**Average of total time spent:** 2 minutes
Appendix D: User Manual
APMv3 MAP VISUALIZER
USER MANUAL

English Version 1.0
# Table of Contents

**Introduction** .................................................................................................................. XXXV

**Maps Manager** .................................................................................................................. XXXV

Map Editor .......................................................................................................................... XXXVII
Map Host .......................................................................................................................... XL
Keyboard Shortcut .......................................................................................................... XLII
Map Viewer ..................................................................................................................... XLIII

**Categories Manager** ...................................................................................................... XLV

**Icons Manager** ............................................................................................................. XLVI

**Activities Tracker** .......................................................................................................... XLVIII
Introduction

Welcome to APMv3 Map Visualizer! APMv3 Map Visualizer is one of the features in APMv3 application. This user manual will provide you basic sufficient things you need to know to use APMv3 Map Visualizer.

This is the Main Menu of APMv3 Map Visualizer. Here you can find four buttons that go to four main functionalities in APMv3 Map Visualizer: Maps Manager, Categories Manager, Icons Manager, and Activities Tracker.

Maps Manager

In APMv3 Map Visualizer, the term Maps refers to the network diagram that shows all the map hosts that are placed on the map along with their statuses.

This is a preview of how a map looks like. We will explore all the features in Maps Manager one by one, but first, we have to understand these terms first in order to easily understand the other later explanations: map host and map host relation.

- **Map host**: The host icon that is placed on the map
- **Map host relation**: A relationship between one map host to another map host (the black
Okay now, we are going back to the topic, Maps Manager.

This is the Maps Manager page. Here you can create a new map, edit your maps, and delete your maps.

- To **create** a map, press the *Create a New Map* tab (marked with *).
- To **edit** one of your map, you can select which map you want to edit and press the edit button (marked with *).
- To **delete** your map, press the delete button (marked with *).

When you want create a new map, and you press the create a new map (*), you will be redirected to Create a new Map tab and you have to fill in your map name (mandatory) and map description.
After filling out the information, you will be redirected to Map editor to edit your map directly. This map editor can also be accessed for your other maps by pressing the Edit button (*) in your map you want to edit.

Map Editor

Below is the interface of map editor page. Map editor is the only page where you can edit your map. When you press Logo you will be redirected back again to the main menu.

Let’s start from Top Bar. The top bar contains two important input fields.

- The one on the left is called **Icon Search Bar**. Here you can search all available icon images in APMv3 Map Visualizer by typing its name or description as the keyword.
- The one beside it is called **Host Search Bar**. Here you can search all available APM hosts.

It’s very easy to add an icon into your map. Just type your keyword in the **Icon Search Bar** (for example “Computer”) and drag and drop your desired icon. (drag and drop the image). If you refuse to do drag and drop, you can just simply click your desired icon and it will appear at the top of your map.
Same thing is also applied to Host Search Bar. It’s as easy as just search and drag!

Now, we continue to the Side Bar.

The side bar is toggable and hideable. It means that the side bar will only appear if you make it appear (trigger it) and will disappear when you don’t need it anymore. To make the side bar appear, press the side bar menu just below the Logo. The menu will appear when you hover your mouse over it.
Here is the preview of the Side Bar.

(1) This will navigate you outside the map editor. There will be a confirmation to save the map if you made any changes.

(2) Map Menu contains the information of the map. You can change those information and also upload a map background (max 3MB). You can save the map from this map menu.

(3) Map viewer is a page to view the map (will be explained later)

(4) This is another alternative to add icon into a map. It’s like Icon Search Bar, but you don’t need to search anything here. It will show you all icons grouped by their category and all you need is just to select your desired icon and do drag and drop on the icon.
The **Main Bar** is the biggest space on the map editor. It is basically the map itself which contains all the map hosts along with their relations. Main bar also contain the background image of the map. You can set the background of the map by opening the Map Menu.

### Map Host

If you have a map host on a map, you can change its data by simply click on it. A popover will appear and you can change the map host data.

1. If you click this input, a modal will shows up and you will be able to search all APM hosts. This input is the host name input which set the map host to a specified APM host.
Select a host that you want to attach to the selected map host. If you see from the hosts autocomplete option, there is an image indicating that this host should have that icon normally. But the choice is yours, whether you want to use that icon image or any other icons.

(2) This input is the host label. When you fill any information on this input, it will appear below the map host. Ideally you would put a short and clear information as a understandable explanation of the map host. The up arrow button (↑) beside this input is the host name copier. If you feel like the host name itself is quite clear to explain your map host, then you can just copy the host name using this button.

(3) This input is the option input to select a map name. If you select one of the maps here, later when you view the map and this map host is clicked, you will be redirected to that selected map.

Note: if you select a map here, you will not be able to select a host. So a map host which has a map redirection will have no host.

Now you will be explained about the four buttons on the map host popover:

- Relate Host
- Delete Host
- Change Host Icon
- Set Host Icon to Default

Relate Host
You can relate a map host with another map host(s). Map host(s)? Yes it means that you can relate to one or many other map hosts (0..n). When you press the Relate Host button, a thin black line will appear and it will follow your mouse cursor. Hover your cursor over any other map host and click that map host. That is how you make a relation between a map host with another map host. If you want to cancel relating your map host, press [ESC]

![Relating a map host to another map host](image)

Delete Host
This button is of course to delete a map host. Make sure you don’t need this map host anymore before deleting it. You can also delete a map host by selecting it and press [DEL] on your keyboard.

Change Host Icon
You can change your map host to another icon. By clicking this button, a modal will open and you can easily select which icon you want to change.

Note: All iconless host will not be able to use this button (Iconless host will be explained later)
Set Host Icon to Default
This button can be clicked only if you have attached a host within your map host. This button will change your map host icon to its default icon according to the host name you selected.

Note: All iconless host will not be able to use this button (Iconless host will be explained later)

Keyboard Shortcut
There are several keyboard shortcuts for the map editor:

<table>
<thead>
<tr>
<th>Keys</th>
<th>Condition</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTRL+DEL</td>
<td>-</td>
<td>Toggle deletion state of map host relation(s). During the deletion state, all of the map host relations will turn red and when one of them is clicked, it will be deleted.</td>
</tr>
<tr>
<td>DEL</td>
<td>A map host has to be selected</td>
<td>Delete the selected map host</td>
</tr>
<tr>
<td>CTRL+S</td>
<td>-</td>
<td>Save the map</td>
</tr>
<tr>
<td>SHIFT</td>
<td>A map host has to be moved</td>
<td>Straight-align the map host vertically or horizontally depending on the position of the mouse cursor</td>
</tr>
<tr>
<td>↑ (up)</td>
<td>A map host has to be selected</td>
<td>Move the map host up</td>
</tr>
<tr>
<td>→ (right)</td>
<td>A map host has to be selected</td>
<td>Move the map host to the right</td>
</tr>
<tr>
<td>↓ (down)</td>
<td>A map host has to be selected</td>
<td>Move the map host down</td>
</tr>
<tr>
<td>← (left)</td>
<td>A map host has to be selected</td>
<td>Move the map host to the left</td>
</tr>
</tbody>
</table>
Especially for the [CTRL+DEL], you use that to delete a relation between two map hosts.

After [CTRL+DEL] has been pressed, you enter the Delete Relations Mode. Click on the relation(s) / red lines you wish to delete. After you are done deleting the relation(s) press [CTRL+DEL] again to escape the Delete Relations Mode.

Map Viewer

After you have done editing the map, (don’t forget to save it) you can view it from the Map editor Side Bar or Maps Manager and see its real host status:

To see the services inside the map host, hover over it and you can see the services inside that host (if the host has services). You can also see all available services along with the map information below the map:
Previously, there is a term called Iconless Host.

**Iconless Host:**
A special icon that only appears on map editor and will be hidden on map viewer. Only its host status that replace it instead.

So in map editor you will see this:

![Iconless Host Icon](image)

And in map viewer you will only see this instead, the **host status**, replacing the iconless host:

- ![Up](image)
- ![Down](image)
- ![Unreachable](image)
- ![Pending](image)
Let’s go to Categories Manager. You can access it from the main menu by clicking on the Categories manager option.

**Category**: A set/group of icons based on their similarities and functions.

**System Category**: A category that can’t be edited or deleted except by the system administrator. Can also be called as initial category

If you open the categories manager, the display will be like this:

You can add a new category (*), edit your non-system category (*), delete your non-system category (*). Every new category you add will be a non-system category.
Icons Manager

Icon: The host image you use to represent a map host in a map.
System Icon: An icon that can’t be edited or deleted except by the system administrator.

If you open the icons manager, the display will be like this:

You can add a new icon (★), edit your non-system icon (★), delete your non-system icon (★).
Every new icon you add will be a non-system icon (like the image above. When the edit and delete buttons are not disabled, then it’s a non-system icon).
Below is the list of the initial categories with their icons:

<table>
<thead>
<tr>
<th>Category</th>
<th>Icon name</th>
<th>Icon</th>
<th>System icon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Infra-structures</td>
<td>Iconless Host</td>
<td><img src="image" alt="Icon" /></td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Computer</td>
<td><img src="image" alt="Icon" /></td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Server</td>
<td><img src="image" alt="Icon" /></td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Router</td>
<td><img src="image" alt="Icon" /></td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Switch</td>
<td><img src="image" alt="Icon" /></td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Firewall</td>
<td><img src="image" alt="Icon" /></td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Storage</td>
<td><img src="image" alt="Icon" /></td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Air Conditioner</td>
<td><img src="image" alt="Icon" /></td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>UPS</td>
<td><img src="image" alt="Icon" /></td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Telephone</td>
<td><img src="image" alt="Icon" /></td>
<td>No</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Category</th>
<th>Icon name</th>
<th>Icon</th>
<th>System icon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Platforms</td>
<td>Windows Host</td>
<td><img src="image" alt="Icon" /></td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Linux Host</td>
<td><img src="image" alt="Icon" /></td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Database</td>
<td><img src="image" alt="Icon" /></td>
<td>Yes</td>
</tr>
<tr>
<td>Applications</td>
<td>Website</td>
<td><img src="image" alt="Icon" /></td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Active Directory</td>
<td><img src="image" alt="Icon" /></td>
<td>Yes</td>
</tr>
<tr>
<td>Clients</td>
<td>Sensor</td>
<td><img src="image" alt="Icon" /></td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Workplace</td>
<td><img src="image" alt="Icon" /></td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Printer</td>
<td><img src="image" alt="Icon" /></td>
<td>Yes</td>
</tr>
</tbody>
</table>

If you add a new icon, you also have to upload an image. It is advised to upload a .PNG image with the same width and height. There is no size limit, but normally the application will treat an image well with minimum 100x100 pixel size.
Now if you navigate to main menu and select the Activities Tracker, you will see every action that you have done such as Add/Edit/Delete Map, Add/Edit/Delete Category, and Add/Edit/Delete Icon.
Appendix E: Project Plan
# PROJECT PLAN

**FONTYS UNIVERSITY OF APPLIED SCIENCES**

**HBO-ICT: English Stream**

<table>
<thead>
<tr>
<th><strong>Data student:</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Family name, Initials:</td>
<td>Tiono, HST</td>
</tr>
<tr>
<td>Student number:</td>
<td>2583399</td>
</tr>
<tr>
<td>Project period: (from – till):</td>
<td>February 2nd – June 30th 2015</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Data company:</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name company/institution:</td>
<td>Acknowledge Benelux BV</td>
</tr>
<tr>
<td>Department:</td>
<td>Acknowledge Proactive Monitoring (APM)</td>
</tr>
<tr>
<td>Address:</td>
<td>Burgemeester Mollaan 80, 5582 CK Waalre</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Company tutor:</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Family name, Initials:</td>
<td>Peperkamp, TP</td>
</tr>
<tr>
<td>Position:</td>
<td>Head of APM</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>University tutor:</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Family name, Initials:</td>
<td>Oosterkamp, JO</td>
</tr>
</tbody>
</table>

Approved and signed by the company tutor:

- **Date:** March, 12th 2015
- **Signature:**

Approved and signed by the university tutor:

- **Signature:**

Agreed and signed by the student:

- **Date:** March, 12th 2015
- **Signature:**
Project Definition

2.1. Current Situation / Problem Statement

As have been mentioned above, APM uses a tool called Nagios. In order to make visualizations of the monitoring result (visualizations could be defined as displays or interfaces, in the form of maps or diagrams), APM uses NagVis which is a plugin for Nagios. Figure 2 and 3 show the overview of how NagVis looks like.

How NagVis works

It first loads an image and set it as the map background. The image could be physical image like shown in Figure 3 or logical image (a form of maps and diagrams) like shown in Figure 2. Then NagVis is just simply adding small icons which indicate that it is a host or service. Each icon symbol also indicates the status of the host. For example if the icon is green, then the host status is okay. If the icon is red then the host status is critical. Figure 4 shows exactly the icons in various different host status/condition.
To create the background image, usually, the company uses a modeling or diagram builder applications such as Microsoft Visio. The form of the diagrams are image based files (.jpg/.png) which are not so effective and takes time to edit, arrange and store. That is one problem that APM faces because APM users have to spend some time to create or draw the image every time they want to visualize the monitoring result. It’s even worse if they have to modify some information or adding a new host, then they have to re-edit the image background to match the correct and current situation.

2.2. Project Justification / Project Description

APMv2 (the current APM application) uses an inefficient way of visualizing the monitoring result because:

- It depends on third party diagram builder software such as Microsoft Visio. Without that software, NagVis can still be run but only in the form of small icons which is not user friendly and unclear. The company needs Visio to build the diagram and set it as a background of the map.
- The map is static which is difficult to edit and maintain whenever any changes occur. This costs time (which also costs money) because there are a lot of modifications made.

The project in overall is to research and build a new way in visualizing the monitoring results. There are several possibilities to accomplish the project requirements:

- NagVis might still be used but it will be modified to become a more like object oriented visualization tool, so they do not have to draw NagVis background image like has been done until now. APM has various host icons, for instance a server icon, router icon, firewall icon, etc. The modification should include adding the host icon into the final application. It means that the Nagvis icons shown in Figure 4 must be modified because it is not able to show the host icon, but just showing the host status (whether it is okay, critical, warning, etc).
- Or maybe other solutions are using other completely different visualization tools or facilities (not using NagVis) and there is another possible tool other than NagVis. It will answer the problem statement because what the company desires is to have an object oriented visualization for the work they are doing.

In order to complete this project, research needs to be done. There also has to be some research questions defined in the project plan so it will be easier to accomplish all the goals of the project. Here are some items that can be considered as research questions:

Researching NagVis, other monitoring tool, or plugin to create map visualization
- How does NagVis work?
- Are there any other available map visualization tools?

Designing and implementing the map visualization on APMv3
- What is APMv3?
- What is the best and the most effective way to build a visualization tool for APMv3?
  What are the criteria to choose the best one?
- How are host objects for map visualization tool designed and developed?
- How are host relations for map visualization tool designed and developed?
- What kind of plugin that can be suitably used for the implementation of the host relations?
- What is the APM API?
- What is MK live status?
- How is live status data from the APM API retrieved?

Formulating and executing test plans
- What are the test items? Who are the test subjects?
- How are the test items executed?
- What are the advantages of the final product compared to other map visualization products?

2.3. Final Product

The final product is a tool to create and maintain map visualizations for APMv3, showing the current hosts’ condition and status in a form of a webpage. The tool will be named “APM Map Visualizer”. The map visualization is based on Nagios data that is retrieved via the APM API. The expected interface and functionality should be as user-friendly as possible.

Here are the functionalities of the program:
- Add/edit/delete maps including their hosts and relations (an empty map is required to perform the delete functionality).
- Add/edit/delete icons (there are system icons which can’t be deleted or modified. In order to delete an icon, the icon may not be used in any map).
- Add/edit/delete icon categories (there are system categories which can’t be deleted or modified. an empty category is required to perform the delete functionality)
- View activities
- Authorize every user to specific functionalities/actions

The final product can overcome the current map visualization’s problem which requires APM users to recreate the image-based map whenever a change occurs in the map.

2.4. System Development Methodology Approach

The system development for the project can be considered as incrementals using the waterfall approach. It means that there will be small waterfall processes in every increment. The reason of using the incremental approach is because the project can be divided into several parts or progress. Here are the project increments or phasing that can be planned:

<table>
<thead>
<tr>
<th>Increment</th>
<th>Work / Activity</th>
<th>Completion</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Doing a library research about NagVis and other monitoring tools besides NagVis. Find out NagVis advantages and disadvantages. After that, making a decision whether to stay using NagVis with an object oriented optimization or to find another possible tool besides NagVis to be used for the APMv3 map visualization. If it seems to be nearly impossible in editing NagVis file or there is no compatible tool found to use for Nagios data, then the last decision is to create the tool itself using plugins available. The decision should be based on deep research and findings</td>
<td>Has to be completed</td>
</tr>
</tbody>
</table>
2. Designing APMv3 map visualization and along with the database design. Has to be completed
3. Building object-oriented design and implementation for the APM API 3 map visualization. Has to be completed
4. Building relation for the objects created in the Work/Activity 3 Will be implemented after previous increment
5. Building user authorization/authentication for map editing/viewing/deleting permission Will be implemented after previous increment (optional / additional work)
6. Writing and Executing test plans Has to be completed

Figure 5: Project increments table

2.5. Project Deliverables and Non-Deliverables

All deliverables that can be mentioned and described for the whole project are:

<table>
<thead>
<tr>
<th>Deliverable</th>
<th>Detail</th>
<th>Latest Date</th>
<th>In the form of</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Final version of project plan</td>
<td>March 9th, 2015</td>
<td>soft copy</td>
</tr>
<tr>
<td>2</td>
<td>Final version of design document</td>
<td>March 16th, 2015</td>
<td>soft copy</td>
</tr>
<tr>
<td>3</td>
<td>Final version of test report and manual</td>
<td>June 1st, 2015</td>
<td>soft copy</td>
</tr>
<tr>
<td>4</td>
<td>Final version of the graduation report</td>
<td>June 16th, 2015</td>
<td>soft copy</td>
</tr>
<tr>
<td>5</td>
<td>Final version of the program</td>
<td>June 30th, 2015</td>
<td>soft copy</td>
</tr>
<tr>
<td>6</td>
<td>Log book*</td>
<td>June 30th, 2015, updated daily</td>
<td>spreadsheet (on the cloud)</td>
</tr>
</tbody>
</table>

Figure 6: Project deliverables table

*Log book is one of the deliverables in the project. It contains all of the activities that have been done during the graduation project along with its time duration (it is updated daily). It is accessible on the cloud (using Google Drive) in this link: [https://docs.google.com/spreadsheets/d/1Ir64n5ETt2pblsi40ys6jE5K-cR5gbEbjh4pa4GspQ/](https://docs.google.com/spreadsheets/d/1Ir64n5ETt2pblsi40ys6jE5K-cR5gbEbjh4pa4GspQ/)

The item that will not be delivered (non-deliverables) are:
- Meeting minutes with university tutor or company tutor

2.6. Project Phasing

Below is the project phasing along with its deliverables that can be thought at the moment. The content of this project phasing will refer to point 2.4 and 2.5, System Development Methodology Approach and Project Deliverables and Non-Deliverables. The detail of the works can be seen in the project increments table (Figure 5) and the deadlines or the form of submission can be seen in the project deliverables table (Figure 6).
2.7. Project Risks

Every project in general has risks and in the project plan, every programmer must try to define the project risks as they might be very helpful later. The project risks’ impacts and probabilities can be categorized as very low, low, medium, high, very high. Here are the project risks that can be defined at this moment:

<table>
<thead>
<tr>
<th>Risk</th>
<th>Impact</th>
<th>Probability</th>
<th>Possible Countermeasure</th>
</tr>
</thead>
<tbody>
<tr>
<td>NagVis program code is not editable or contains over complicated functions</td>
<td>High</td>
<td>High</td>
<td>Leave NagVis program in its original state, make another possible solutions.</td>
</tr>
<tr>
<td>Missing the deliverables’ deadline</td>
<td>Medium</td>
<td>Medium</td>
<td>Working overtime to avoid other unfinished deliverables</td>
</tr>
<tr>
<td>NagVis config files can not be migrated to an operational database</td>
<td>Low</td>
<td>Low</td>
<td>Keep using the config files to store maps data</td>
</tr>
<tr>
<td>Misunderstanding or communication problem</td>
<td>Very High</td>
<td>Very Low</td>
<td>Always ask for the correct expected results in every deliverable.</td>
</tr>
</tbody>
</table>

2.8. Project Constraints

Main Language:

Both the final application and all of the documents will be written in English. However, if in the end the company requires a Dutch final application, then only the display interface that has to be changed to Dutch. All the comments, database structure, and variables will remain in English.

Programming Language:

The application will be coded in HTML using JavaScript (possibly jQuery) as the client side programming language, PHP programming language as the server side programming language, and it will also use some CSS plugins like bootstrap.
Database:
The database that will be used is MySQL, using PHP Data Objects (PDO) as the database access and retrieval method.

Working Description
The graduation project will last 101 days in total, excluding weekends and public holidays with working hours of 8 hours per day. There will be a log book to store working details per day which is important in tracking works, finding mistakes, finding code histories, and also as a proof of working in detail.

Communication Plan
4.1. Contacts Details

Intern
<table>
<thead>
<tr>
<th>Name</th>
<th>Hans Sebastian Tiono</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address</td>
<td>Engelsbergenstraat 2B, 5616JC</td>
</tr>
<tr>
<td>Personal Email</td>
<td><a href="mailto:hans.sebtino@yahoo.com">hans.sebtino@yahoo.com</a></td>
</tr>
<tr>
<td>School Email</td>
<td><a href="mailto:h.tiono@student.fontys.nl">h.tiono@student.fontys.nl</a></td>
</tr>
<tr>
<td>Number</td>
<td>0616058688</td>
</tr>
</tbody>
</table>

Company Tutor
<table>
<thead>
<tr>
<th>Name</th>
<th>Tom Peperkamp</th>
</tr>
</thead>
<tbody>
<tr>
<td>Email</td>
<td><a href="mailto:TomPeperkamp@acknowledge.nl">TomPeperkamp@acknowledge.nl</a></td>
</tr>
<tr>
<td>Number</td>
<td>0683176607</td>
</tr>
</tbody>
</table>

University Tutor
<table>
<thead>
<tr>
<th>Name</th>
<th>Jelle Oosterkamp</th>
</tr>
</thead>
<tbody>
<tr>
<td>Email</td>
<td><a href="mailto:j.oosterkamp@fontys.nl">j.oosterkamp@fontys.nl</a></td>
</tr>
<tr>
<td>Business Number</td>
<td>0885071381</td>
</tr>
<tr>
<td>Mobile Number</td>
<td>0620972848</td>
</tr>
</tbody>
</table>

4.2. Communication Plan

<table>
<thead>
<tr>
<th>Between</th>
<th>Purpose</th>
<th>Importance</th>
<th>Via</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intern and University tutor</td>
<td>Consultation of working progress</td>
<td>Medium</td>
<td>Email</td>
</tr>
<tr>
<td></td>
<td></td>
<td>High</td>
<td>Appointment</td>
</tr>
<tr>
<td>Intern and Company tutor</td>
<td>Consultation of working requirements and results</td>
<td>Any</td>
<td>Direct meeting in the office</td>
</tr>
<tr>
<td>Intern, Company Tutor, and University Tutor</td>
<td>Company visit</td>
<td>High</td>
<td>Direct meeting in the company office</td>
</tr>
</tbody>
</table>
Project Plan: References

[1] Image of Figure 2, http://www.nagvis.org/images/screenshots/nagvis_vmware.jpg

[2] Image of Figure 3, http://www.nagvis.org/images/screenshots/c_by_dave_rearden_2.png

[3] Image of Figure 4, http://docs.nagvis.org/1.4/en_US/extending/iconsets.html
Appendix F: Graduation Project Survey Form
Appendix: Graduation Project Survey
HBO-ICT: English Stream

University of Applied Sciences

Specialisation: ICT & Software engineering / Business / Technology, BME, other:

Data student:
Name student: Initials: HST
Name: Hans Sebastian Tjono
First name: Hans
Student number: 2583399
Telephone: 0616086688 / 0617343279
E-mail: hans_sebtino@yahoo.com

Data company:
Company mentor: Initials: -
Name: Tom Peperkamp
Telephone: +31 (0) 40 254 44 30 / +31 (0) 6 83 117 66 07
E-mail: tompeperkamp@acknowledge.nl

Department/position: Acknowledge Proactive Monitoring (APM)

Startdate Graduation project: February 1st, 2015
Duo Graduation project: Yes/ No
If duo name of buddy: -

Accepted by student: date: 15-1-2015 signature:

Hand in date Graduation Project Survey:
Approved by graduation project coordinator: yes/ no date: signature

Remarks: ________________________________

LIX
Description of the graduation project:

1. **Describe the problem analysis:**
   (what is the reason for the internship company to initiate this assignment? What is for the company the added value of this assignment? Can you describe the starting situation and starting points: introduction and problem definition.)

   Acknowledge has three main departments or operations. One of those business lines is Service Solutions. Service Solutions contains managed services and the in-house development of Prioxx and Acknowledge Proactive Monitoring (APM). APM focuses on developing a system to monitor their client’s computer system, network, and infrastructure environment. Up until now, APM has implemented Nagios which is an open source tool to implement such monitoring system above. Usually, the company uses modeling or diagram builder application such as Microsoft Visio to create the visualization diagram for the system infrastructure or networking diagram. The form of the diagrams are image based files (.jpg/.png) which are not so effective and takes time to edit, arrange and store. What the company desires is to have an object oriented visualization of the work that they are doing.

2. **Describe the graduation assignment.**
   (especially the objectives, results to be delivered and final products to be realized. Also indicate what you want to achieve for the internship company. Give a clear description of the graduation assignment).

   The graduation assignment will be to help the company implement a new way of visualizing the working diagrams using object oriented visualization instead of using the current image based files. The final products should be a webpage that is used mainly to show diagrams based on Nagios data that are retrieved from a database. The graduation assignment is also to research the possibility of presenting enriched NagVis data for the diagram visualization.

3. **What is the research component of this assignment?**
   (What are your research topics? If necessary, draw up a research plan).

   There are some things to be considered as the research components:
   1. How to Design and Develop a Visualization Model.
   2. How the structure and functionalities of Nagios and NagVis are.
   3. How to enrich the system with metadata (like icon per host type).
   4. How to retrieve NagVis data via the APM API.

4. **What are the methods and tools?**
   (What operating procedure and means can you made use of during the internship period at the company? What facilities will be made available by the company?)

   Nagios has a specific add-on to handle data visualization problems. The add-on is called NagVis which stands for Nagios Visualization. This is a good add-on to be implemented in order to have an image visualization of Nagios data. The web client retrieves the data via the APM API.

   Acknowledge uses Git as the code repository system, linked to a development, test, acceptance and production environment.

5. **How and by whom will you be guided by the company, which other ‘stakeholders’ are involved?**
   Mr. Tom Peperkamp will be the project tutor for the graduation project and some programmer colleagues might be the partner or the ones who help to introduce and explain the technical part.

6. **What fields of Study play an important factor in realizing the graduation assignment?**
   (for example information analysis, design, realization, monitoring and security).

   - Information analysis: Technical investigation and deep study of Nagios and NagVis.
   - Programming/Realization: Adapting to Linux environment and the implementation Visualization module based on NagVis data.