



TAMPEREEN TEKNILLINEN YLIOPISTO  
TAMPERE UNIVERSITY OF TECHNOLOGY

NAZIA HASAN  
CONCEPT DEVELOPMENT AND USER EXPERIENCE MEAS-  
UREMENT OF DIPOR DASHBOARD FOR MONITORING STATUS  
FOR DIGITAL SERVICE DEVELOPMENT PROJECTS

Master of Science Thesis

Examiner: Dr. Heli Väättäjä  
Supervisor: Dr. Heli Väättäjä  
Examiner and topic approved by the  
Faculty Council of the Faculty of  
Computing and Electrical Engineer-  
ing  
on June 8th, 2016

## ABSTRACT

**NAZIA HASAN:** Concept Development and User Experience Measurement of Dipor Dashboard for Monitoring Status for Digital Service Development Projects  
Tampere University of technology

Master of Science Thesis, 123 pages, 43 Appendix pages

June 2016

Master's Degree Program in Information Technology

Major: User Experience

Examiner: Doctor Heli Väättäjä

**Keywords:** Dashboard, Digital Service, Service Development Monitoring, User Needs, Agile Development Methodology, GitHub

In Finland, tax payers' money is used by public sector organizations to implement open source digital services to solve problem situations raised by common citizens. However, traditional long development cycle often results with solutions that don't address intended users' needs. To remedy this, Digipalvelutehdas community introduced a process that would require a 3-month long development phase to produce a testable proof of concept for any digital service. To permit further implementation, the development progress needed to be closely monitored. Digipalvelutehdas brought up the idea of Dipor Dashboard for monitoring service development and Sampo Software Oy was assigned to develop a testable interface. The thesis work focused on defining requirements for this dashboard out of needs from its intended users and proposed a concept using low fidelity design sketches. The goal was to determine how attributes from GitHub repositories could be visualized in a Dashboard view to project development progress. The usability and feasibility of the system developed by Sampo Software is also evaluated. User experiment study of the system had been done in a limited scope.

The thesis report begins with background work. A competitor analysis for existing market solutions is then provided. A discussion is made on existing work over Dashboard design, Agile development visualization and using GitHub attributes to build Agile workflows. First phase of empirical work involved interviewing the customer and intended users to develop concept for the dashboard using iterative design and evaluation of low fidelity prototypes. Usability evaluation of the implemented system was done in second phase with two heuristic evaluations and five usability tests. A four week long user study was initiated with two participants in the third phase, which was continued for two weeks due to unavoidable circumstances.

The implemented Dipor Dashboard focused more on organization hierarchy than on Digital services. The supposed dashboard view turned out to be an integration view for comparing different repository works intended for the same service. Used charts had issues in look & feel, functionality and data representation. Development progress wasn't visualized following Agile methodology. Major design and functionality rework would be needed to make the system more usable. Although being a better option, the developed concept needs more research on appropriate visualizations and common data framework to integrate systems other than GitHub repositories.

## PREFACE

My study period in Tampere University of Technology was a completely new experience for me, considering I come from outside Finland. So naturally, undertaking my thesis work in an international education system was filled with challenges. The start was rocky considering I was in a fix to follow a direction. But in the long run, things became easier and the work was rewarding.

My sincere thanks go to my supervisor, Dr. Heli Väättäjä. You encouraged me to setup my own focus and at the same time kept checks on me so that I don't get lost in too many options. Working for a company thesis might often feel demoralizing and frustrating, especially if there is mismatch between visions and objectives. You boosted me up when I was feeling down and told me to be confident with my works and findings. I am grateful to you for your advices and suggestions to make the thesis project easier for me.

I'd also like to give thanks to Sampo Software Oy for considering me eligible to be part of their project work. Special thanks to Jarkko Moilanen, customer of Dipor Dashboard project, for your helping hands whenever I needed. I am grateful to my friend, Ekaterina Olshannikova. You have been a great companion with full of supports and confidence on my capabilities. Thank you Rashidul Sadi, my beloved, for being my rock and tolerating my tantrums on the difficult days.

This would be incomplete without acknowledging the moral support and contribution from my parents. You had the faith and confidence that your daughter would make her own place, all by her herself in this foreign land. What I am today is because of your encouragement and prayers. Thanks are never enough to repay your hardship.

I have always been a person of faith. My sincere gratefulness to the Almighty for showing me the ray of hope when things appeared difficult to endure.

In Tampere, 22.11.2017

Nazia Hasan

## CONTENTS

1.	INTRODUCTION .....	1
1.1	Background and Motivation.....	2
1.2	Research Objectives and Methodology.....	5
1.3	Thesis Structure.....	6
2.	COMPETITOR ANALYSIS .....	8
2.1	General Analysis of the Compared Systems.....	12
2.2	GitHub Integration Tools .....	16
2.3	Advantages and Limitations of the Discussed Systems.....	17
2.3.1	Advantages and Limitations of GitHub .....	18
2.3.2	Advantages and Limitations of the Discussed Systems for Development Progress Monitoring .....	18
3.	EXISTING WORKS ON DASHBOARD DESIGN, AGILE DEVELOPMENT VISUALIZATION AND GITHUB REPOSITORY ATTRIBUTES.....	20
3.1	Dashboard Design .....	21
3.2	Agile Development Visualization .....	24
3.3	GitHub Attributes to Monitor Development Progress and Visualize Agile Workflows .....	27
4.	RESEARCH METHODOLOGIES: PHASE 1 – CONCEPT DEVELOPMENT ..	30
4.1	Sub Phase 1: Customer Interview .....	31
4.1.1	Interview Questionnaire.....	32
4.2	Sub-phase 2: User Interviews and Evaluation of Low-Fidelity Prototypes ..	34
4.2.1	Participant Recruitment.....	34
4.2.2	Interview questionnaire.....	36
4.2.3	Paper Prototype Evaluation of Dipor Dashboard.....	38
4.3	Results from the Executed Methodologies.....	39
4.3.1	Construction of Affinity Diagram.....	39
4.3.2	Personas .....	41
4.3.3	User Requirements.....	46
4.3.4	Sketches of the Concept Developed for Dipor Dashboard .....	47
5.	AN OVERVIEW OF THE IMPLEMENTED DIPOR DASHBOARD .....	57
5.1	Home Page of Dipor Dashboard .....	57
5.2	Organization view .....	58
5.3	Service Integration View.....	59
6.	RESEARCH METHODOLOGIES: PHASE 2 – USABILITY EVALUATION OF DIPOR DASHBOARD.....	63
6.1	Heuristic Evaluation.....	64
6.1.1	Chosen Heuristics for Evaluation .....	64
6.1.2	Heuristic Evaluation Procedure .....	69
6.1.3	Identified Usability Problems from Heuristic Evaluation.....	69

6.2	Usability Testing of Dipor Dashboard .....	74
6.2.1	Participants in the Usability Tests.....	74
6.2.2	Technical Aspects about the Conducted tests .....	76
6.2.3	Procedure of the Usability Tests .....	76
6.2.4	Tasks in the Usability Tests .....	77
6.2.5	Results Obtained from the Usability Tests .....	81
6.2.6	Feedback from Satisfaction Questionnaire and Interview .....	84
6.2.7	Problems Found in the Usability tests.....	88
7.	RESEARCH METHODOLOGIES: PHASE 3 – LONG TERM USAGE STUDY	
	97	
7.1	Description of the Method Used .....	97
7.2	Description of the Study Procedure .....	98
7.3	Results from Long Term Usage Studies .....	100
8.	DISCUSSIONS .....	105
8.1	Analysis of Dipor Dashboard with respect to Existing Solutions and Dashboard Design.....	105
8.2	User Experience .....	107
8.3	Evaluation of Developed Concept for Dipor Dashboard .....	108
8.4	Validity and Reliability .....	111
8.5	Usefulness of the Methodologies Selected.....	113
9.	CONCLUSIONS.....	115
9.1	Improvement Suggestion for the Implemented System .....	115
9.2	Further Studies for Developed Concept .....	117
	REFERENCES.....	119

APPENDIX A: Affinity Diagram

APPENDIX B: Interview and Usability Testing Materials

APPENDIX C: Forms used in usability testing and test task template

## LIST OF SYMBOLS AND ABBREVIATIONS

Abbreviation/Terms	Explanation
Commit	Individual changes made to a file in a GitHub repository. A new ID is created every time a change is made to keep track of the time of the change, person who made the change and content of the change.
Contributor	A person who has merged a pull request in a GitHub repository. A contributor doesn't have collaborator privileges.
Collaborator	A person having read and write access to a GitHub repository and can contribute in tasks within that repository.
Fork	Refers to a copy of a GitHub repository. By using forking feature, experiments on system code can be conducted without affecting original repository.
HCI	Human Computer Interaction
IT	Information Technology.
Kanban	An Agile development framework. In this framework, features are developed based on customer demands. The simplest Kanban flow has three phases: Requested, In Progress and Done.
KPI	Key Performance Indicator. For a dashboard to monitor digital service, it could an attribute (e.g. number of open issues in GitHub by time) to measure how development progress is going on.
Metadata	Additional characteristics of any digital material. E.g. for a GitHub repository issue, its creation date, closing date, etc. can be the considered as metadata for that issue.
MILC	Multi-dimensional In-depth Long term Case Studies
PO	Product Owner. In Scrum, the person solely responsible for managing product backlog and maximizing product value and work effort of development team.
POC	Proof Of Concept. This is realization of a method or idea with evidence derived usually from a pilot project. This is done to demonstrate the feasibility of a design concept, business proposal, etc.
PR	Pull Request. Refers to changes a collaborator has pushed in a GitHub Repository
Scrum	An Agile development framework. Product development is achieved using iterative and incremental implementation cycles.
SWE	Software Engineering. In this domain, any software is developed, operated and maintained by systematically applying scientific and technical knowledge, quantifiable methods and experience.
Threshold	Threshold indicates the magnitude or intensity of the value of an attribute or a valuable which must be exceeded for manifesting a result or condition.
TUT	Tampere University of Technology
UI	User Interface. A space containing feature for establishing interactions between a device or application and its users.

UTA	University of Tampere
WIP	Work In Progress. In Kanban workflow, this indicates the number of tasks a development team is currently working on.

# 1. INTRODUCTION

The thesis work conducted here is part of a development project. This work was undertaken by Sampo Software Oy, located at Tampere city of Finland, in 2016. The primary development work constituted of implementing a web portal for Digipalvelutehdas - Digital Factory Service community. This community consists of people (general employees, project managers, product owners, developers, etc.) from public sector organizations, municipalities, private companies providing digital solutions under different contracts and regular citizens of Finland. In the long run, Dipor Dashboard web portal was to function for two purposes. Firstly, this would have served as an idea pool containing digital solution ideas targeting to solve every day, real-life problem situations faced or raised by Finnish citizens. Secondly, the web portal would have served as a dashboard to visualize development status of various ninety-days long, open source, proof of concepts for digital public service implementations (as defined by Digipalvelutehdas) under different public sector organizations. The digital services being implemented would have been the solutions picked up from the idea pool, based on their popularity determined by the votes and feedback of community members. Sampo Software Oy worked on implementing the dashboard part for monitoring development progress of those digital service implementations.

The thesis work performed under the implementation project consisted of the following primary aspects:

- Comprehending the need to implement a dashboard from scratch by conducting competitor analysis and gathering requirements from potential users.
- Building a concept of the intended dashboard from gathered user requirements using dashboard design rules and available information from code repositories.
- Evaluating the implemented service to determine how well the web portal has been able to serve its purposes based on different usability evaluation methods and long term user study.

This chapter is dedicated to present the background and motivation of the performed thesis work. It also features the research objectives and research questions. In addition, a brief description of the overall thesis structure and methodology followed to conduct the thesis work is discussed as well.



## 1.1 Background and Motivation

Digipalvelutehdas or Digital Factory was originally founded in 2015 by Jarkko Moilanen. At that time, he had been the Head of Development under the General Education and Early Childhood Education department of the Ministry of Education and Culture in Finland. This community was his brainchild to address the problems of traditional long-cycle development periods of digital administration systems under different public sector organizations of Finland. In the past and in some cases in recent years, projects undertaken to develop digital information and administration systems for these organizations (for example different ministries and departments under them, municipalities, city metropolises, government run or semi-autonomous educational institutes, etc. in Finland) had one year-long implementation periods. This long implementation period had certain disadvantages:

- Very few chances to see intermediate progress of the service being developed.
- Lack of testable interface of the developed service to measure its efficiency, effectiveness and success to address a problem situation.
- Lack of opportunities for the intended users to participate in the development phases. This resulted with missing feedbacks and opinions about the generalized concept.
- Poor usability feedback in look-and-feel and functionality of the released system as it met very little of the pragmatic and real needs of the mass Finnish population.
- Unwanted wastage of fund allocation into projects, where the major source of the finances is tax-money of Finnish people.

It was necessary to introduce some process and regulations for digital services being implemented under different public sector contracts so that they adopt efficient development process to produce fast results. Idea solutions developed in shorter phases while producing testable interface would have been beneficial for all involved stakeholders. The stakeholders included organizations making decisions to continue further development and distribute funds accordingly; customers and citizens having a chance to use early prototypes (not the completed solution itself) and to express their opinion on whether or not the developed solution is truly addressing their problem situations; and companies working to create different solutions to determine if they are adopting the right approach or not. Jarkko Moilanen took this initiative and started collecting feedback from different people working within the Education ministry to identify needs and requirements for initiating such a process model. In 2015, the preliminary concept on the community work started with twenty organizations taking part to define the outline. Four pilot idea projects were successfully developed and amongst them were products

of two regional centers. One hundred Finnish citizens took part in ideation and evaluation of implemented services. This approach was successful collecting praise and support from people associated with different public sectors and was able to be initiated as a nation-wide movement. These efforts and needs to improve existing process led to the birth of Digipalvelutehdas.

Digipalvelutehdas is a community that brings public and private sector organizations together in order to address different problem situations faced by Finnish citizens in their day-to-day life. People can join here as company representatives, authority figures or even as ordinary Finnish citizens. The incorporated processes and tools include Open Development Model<sup>1</sup> implemented by Technical Research Center of Finland (VTT) and the model is very suitable for developing open source code services. Implemented services are under the Open Source Initiative License defined by MIT<sup>2</sup>. In this community, citizens are in the central position to bring up problem scenarios they face in everyday life or even ideas to solve them. Problem scenarios and ideas gaining good numbers of supports and followers are brought into limelight. A one-day workshop is then held where possible solutions, their feasibility and business model involving the implementation work for a specific idea are discussed. At the end of the workshop, decision whether or not to proceed with the optimal solution is made by the participants. The company awarded with the development tender needs to implement a testable Proof of Concept (POC) of the idea or solution within a period of ninety days by following Agile Development methodology. The development work is open source, with projects codes readily available in GitHub Repositories with public access. This is done so that anyone can use the existing implementation for further customization. After the 90-day development period, evaluation of the testable POC is made in order to estimate the solution's feasibility of meeting users' needs. Here Finnish citizen, for whom the service is being created, works as the active evaluators. The feedbacks and evaluation result decide about the continuation of further development.

Digipalvelutehdas aims to promote the culture of experiments and goals for government programs to digitize public services within Finland. Apart from the Ministry of Education and Culture, different public sector organizations (e.g. Population Registration Center under the Ministry of Finance, the Ministry of Tax Collection), various municipalities, metropolis organizations from different cities (e.g. Helsinki, Tampere, Oulu and Turku). The community doesn't claim itself to be the solution of all developments, but it targets to achieve 80% of the total amount. It works to develop not only backbone services for the entire Finnish nation, but also local digital services for cities and regions in order to support Finnish citizens with their everyday life. Although the department for General Education and Early Childhood Education under Ministry of Education and

---

<sup>1</sup> <http://opensource.erve.vtt.fi/licensing.html>

<sup>2</sup> <https://opensource.org/licenses/MIT>

Culture started the initiative of digitization of public services, more and more public and private sector organizations have been getting connected to this community.

It is beneficial to bring forward problem situations and solution ideas raised by people in a single platform. This helps authoritative and decision making personnel to comprehend what is lacking in people's everyday life and how it can be resolved. It is also important to know which problems are creating more awareness among people by popular demand and what ideas are gaining more interests and supports. This is needed for the community to select preliminary ideas in order to evaluate their feasibilities. People working in public sectors and in charge of ongoing service developments need to be aware of their real life progress status. Currently there are some of issues regarding this purpose:

- Firstly, the formal procedure of providing monthly report from the developing company to the Product Owner (PO) of the service being developed is time consuming and lacks real-time updates. These reports often don't address the difference of information details preferred by different stakeholders. Also they often lack appropriate visualizations to project progress of the development work.
- Secondly, open source repositories may provide options for status review. However, understanding and managing these repositories in order to access meaningful and necessary information might be difficult for people who are not used to dealing with such systems.
- Thirdly, existing systems or software available in market for monitoring project development works are often quite complex to configure. This might be difficult for people not having enough technical skill as same as that of a person expert in handling them. Also the systems might be difficult to configure in order to reflect the work process and structure of the organization in question.
- Fourthly, almost all of these systems are commercial products where initial purchase fees and yearly license renewal fees are needed to be paid. This comes as a contradiction to a community of open source process, development and services.
- Finally the community believes on openness, transparency and access towards contents and updates on the digitization of public administrative services. The reason is that public tax-money is used to develop such services. Hence it is important that Finnish tax-payers can be aware of ongoing service developments undertaken by different public and private organizations. Additionally, for allowing further customization of developed system, codes, logs, development tasks, etc. related to an implemented digital service should be available in open-source code repository(s).

By keeping the above issues in mind, the idea of Dipor Dashboard was presented towards the Digipalvelutehdas community by Jarkko Moilanen. Like all other initial ideas, based on the collective feedback from different stakeholders, Dipor Dashboard went through the one-day workshop for evaluation of its feasibility and possible business

model. The decision to implement this platform using open source were made and Sampo Software Oy won the invited tender to produce a testable interface within a period of 90-days timeline.

## 1.2 Research Objectives and Methodology

The idea of implementing Dipor Dashboard was considered a testable “Proof of Concept” (POC), where stakeholders wished to see if somehow status of service development work using Agile methodology could be determined and evaluated by incorporating information from code repositories and version control systems. POC may refer to a testable prototype of a digital solution. The prototype doesn’t incorporate all requested features within a system, but is mainly used to verify feasibility of an idea or design. GitHub had been primarily selected to be the code repository and version control system that would be integrated with the dashboard web portal. In addition, Jira was also considered to be another source for obtaining development related data. However, due to the limitation of time, resources and allocated fund, the development work had been limited to implement very basic set of features to represent organizations and development status of ongoing service implementations under them using GitHub data. So the current implemented version doesn’t incorporate all necessary features to address the very first requirements of the service.

The primary focus of the thesis work is about service development status information and their visualization in a dashboard like setting. What information obtained from a code repository (e.g. GitHub) gives highlights about development work? How this information can be presented in a dashboard like system? Is the visualization rich and intuitive enough to help users understand how development progress is going on? How intended users feel about using the dashboard service for monitoring digital service development? The thesis work consists of the following objectives:

- To Study about popular Agile development methodologies and visualizations technique to measure development progress within them.
- To study about dashboard design and ways (to incorporate code repository attributes to reflect software development progress with appropriate visualization)
- To study about available information and in-built visualizations in GitHub and if and how the information can be used to project development progress of digital service implementation using Agile development methodology.

The research questions associated with this thesis work are following:

- What information stored within code repositories and version control system can be represented in a dashboard like setting to project progress of a service development work undertaken under Agile development methodology?

- What is the user experience of the implemented Dipor Dashboard system? How intuitive it is for the intended users to determine whether or not a project is on track?

For the thesis work, the final concept of the dashboard visualization of Dipor Dashboard web portal was constructed following User Centered Design (UCD) methodologies involving user interviews, persona creation and building affinity diagrams to identify user needs and requirements. Information discovered from above was used in an iterative process of creating and evaluating low fidelity prototypes that would develop the concept of the Dashboard. The concept aimed towards incorporating needs and expectations of the intended users to achieve their work goals and overcome limitations and challenges confronted.

The testable POC product developed by Sampo Software Oy was evaluated using expert heuristic evaluation and running usability tests with intended users. A long term usage study was also undertaken. This study helped to understand how the participants are using the features of the Dashboard, if they could achieve their work goals with the provided functionality and what are their overall feelings about Dipor Dashboard. Results found out in both usability evaluation and long term usage study had been analyzed to determine how far Dipor Dashboard has been able to satisfy user requirements. The developed concept was also compared with the results from usability evaluation and long term usage study. Based on the analysis and user feedback, it was determined if the implemented web service had been successful to address the community need of Digipalvelutehdas by serving its purpose.

### **1.3 Thesis Structure**

The first chapter of this thesis document provides insights about the background and motivation for conducting this project work for the community. Also this includes the research objectives and methodologies used for conducting the thesis work. The second chapter provides competitor analysis of existing software and systems functioning as development monitoring services and their limitations for not serving the purpose of the community work. The third chapter includes the study of related works related to dashboard design principles, Agile development visualization and use of GitHub repository attribute to establish Agile workflow. Fourth, sixth and seventh chapter present individual methodologies used to conduct the empirical work in this thesis in different phases and the results associated with each method used. Fourth chapter describes the first phase of empirical work where customer and user interviews were conducted to build personas, affinity diagrams, user requirements. Also this chapter presents the low fidelity prototype for the Dashboard concept that was iteratively developed and evaluated during the interview process. An overview of the implemented Dipor Dashboard web

service by Sampo Software Oy is provided in the fifth chapter. Usability evaluation of the implemented web portal and obtained results from expert heuristic evaluation and usability testing with intended users are described as the work in second phase of empirical work in the sixth chapter. The seventh chapter gives insights to the long term usage study that conducted on Dipor Dashboard in the final phase of empirical work and results obtained from this study. General discussion about the feasibility of the implemented system, user experience, analysis of the developed dashboard concept, validity and reliability of the results, usefulness of the used methodologies are presented in chapter eight. The ninth and final draws conclusive remarks about the overall thesis work by discussing possible improvement suggestions to both the developed Dipor Dashboard and future research scopes on the developed concept.

## 2. COMPETITOR ANALYSIS

The need to create new services or to make specialized use of existing ones is felt when the services in question are not sufficient enough to fulfill user or organizational needs and goals. There can be several concerns that should be addressed when initiatives are taken to build up a new system or service from the scratch when there exists plenty of similar solutions in the market:

1. Existing systems or services are not suitable enough to represent organizational structure and workflow.
2. The configuration of such applications is often too complex to customize and always doesn't reflect the standard workflow maintained by an organization.
3. In most cases, software development and monitoring dashboards are often designed for people having advanced level of technical skills. It is always problematic for users with novice level skills to go through the details of such systems.
4. Often the means of finding required information via the systems and tools are not visible enough in front of its users.
5. The way visualizations are represented, they often fail to imply the relationship between the parameters displayed and how they project development status.
6. Dashboards require having simple, not too fancy or flashy displays that help the viewers to understand what is going on with a brief glance. Most cases, existing systems fails to do so.
7. Accessing and using these systems or services may require installing the system in personal workstation, changing system variables within the machine, etc. It would have been convenient they were hosted into platform that could be easily accessed by people (e.g. web portals, cloud-computer, etc.)
8. There is often lack of using visualizations appropriate for specific context of use. For example, people might wish to look only for milestones completed over time at an abstract level. Or they may wish to see the representation of how created and completed tasks are associated with achieving a milestone. In most case, such systems don't come up with customizable configuration. Even if they do, they are complex enough for people with novice skill-sets.
9. Getting software development progress information often requires integration of the system or service in question with central code repository and task management systems. Investigations are needed to decide how raw data should be visualized as different parameter, feasibility of using such parameters and their contexts of use. This decision making process is often difficult and time consuming.

Also code repositories and task management systems often have their own metrics. Having a common data framework that can integrate system parameters as well as repository parameters together are not always possible and may need additional development.

10. Configuring access to information for different stakeholders according to their involvement is often difficult by using these systems. There might be very little options for customization or too many options to choose from.
11. Most importantly, these tools often come under the developing company's license. To access the full set of features or for continuous usage, people need to pay money after the trial period expires. Also there are additional costs of renewing the license after the completion of a usage period (usually a year). This is not favorable for a community that works with open source, public accessible projects

So to understand the need for developing a new service hosting dashboard to monitor digital service development progress and to design its possible features, a comprehensive analysis is needed for existing systems/software/applications. An analysis gives information on characteristics and options about their dashboard features, their strengths and their limitations. This information gives insights and ideas about how limitations of existing dashboards can be addressed, what new features can be implemented and how it would fulfill specific organizational workflow and requirements.

Competitor analysis for Dipor Dashboard was done solely as part of the thesis work. Certain things were kept in consideration while looking for systems that can visualize digital service development progress:

- Service development projects under Digipalvelutehdas community are done following Agile development methodology and they last for 3 months.
- As the services are of open source nature, GitHub is used as the primary version control system and code repository. As a result, the new system needs to integrate GitHub repositories in a way that it reflects agile development status from the obtained information. So strengths and limitations of GitHub in terms of supporting Agile development methodologies needs to be considered.
- Jira is one of the most popular version control system which also provides an Agile development workflow. Also Agilefant is another well-known open-source backlog management tool. They need to be analyzed to discover their strengths and weaknesses in terms of projecting development progress in a dashboard view.
- There are some existing integration tools which can be used to manage GitHub issues and milestones. Often, they offer graphical representations of life cycles of the above entities. They can be used both online and in personal workstations.



We also need to keep them in consideration in order to justify for building a new system that would satisfy user needs of Dipor Dashboard.

When systems are analyzed, focus would be given primarily on their feasibility as a dashboard for monitoring service development using Agile development methodology. So this needs to be determined if the dashboards of the compared systems or services support appropriate information and visualizations to project agile development progress Compared Systems at a Glance

The following table includes some general information about the existing systems or services that have dashboards or similar visualizations to show progress of Agile Development works. The competitors chosen (except for GitHub) are considered to be among the top Agile Project Management Software of 2016 (Burger, 2016). The description of each system includes entities that can be considered as units to measure development progress. Also visualizations and graphs / reports included in those systems are also mentioned:

*Table 2-1: Competitor Systems of Dipor Dashboard*

<b>System Name</b>	<b>Indicator(s) for Development Monitoring</b>	<b>Graphs or Reports</b>	<b>Dashboard and other Overview Visualizations</b>
<b>GitHub</b> <sup>1</sup>	Issues, Milestones	Contributors', Traffic, Commits, Code Frequency, Dependency Graph, Network, Fork	Pulse
<b>Jira</b> <sup>2</sup>	Issue / Story, Backlog, Sprint, Epic, Version	Burn-down Chart, Control Chart, Cumulative Flow Diagram, Epic Burn-down, Epic Report, Sprint Report, Velocity Chart, Version Report, Release Burn-down	Dashboard, Scrum / Kanban Board

<sup>1</sup> <https://help.github.com>

<sup>2</sup> <https://confluence.atlassian.com/agile/jira-agile-101>

<b>Agilefant</b> <sup>1</sup> (“User Guide   Agilefant,” n.d.)	Tasks, Backlogs, Stories, Sprints	Project Burnup, Sprint Burn-down, Spent Effort by backlogs, Spent Effort by Users	Timelines, Boards, Dashboard
<b>Active Colab</b> <sup>2</sup>	Task, Project	Project Timeline, Team Timeline	Tasks View, Calendar
<b>Agilo for Scrum</b> <sup>3</sup>	Tasks, User Story, Product Backlog, Sprints		Scrum Board, Review Board, Observation/Retrospective Board
<b>Pivotal Tracker</b> <sup>4</sup>	Task, Story, Iteration, Backlog, Project, Epic	Project Overview Report, Velocity Chart, Cycle Time Report, Story Composition Report, Burnup Chart, Burn-down Chart, Cumulative Flow Chart, Iteration Report, Release Report and Burn Downs Report, Epic Report, Story Activity Report	Project View, Workspace, Dashboard
<b>Sprintly</b> <sup>5</sup>	Items, Sprints		Dashboard, Progress View (Burn-down chart), Timeline, Activity View
<b>Sprint Ground</b> <sup>6</sup>	Items, Projects, Release		Dashboard / Home Screen, Task-board

<sup>1</sup> <https://www.agilefant.com/support/user-guide/>

<sup>2</sup> <https://activecollab.com/help/>

<sup>3</sup> <https://activecollab.com/help/>

<sup>4</sup> [https://www.pivotaltracker.com/help/articles/quick\\_start/](https://www.pivotaltracker.com/help/articles/quick_start/)

<sup>5</sup> <https://www.youtube.com/channel/UC8HqCGA8vak-jN-0jZjTCAg>

<sup>6</sup> <https://sandbox2.sprintground.com>

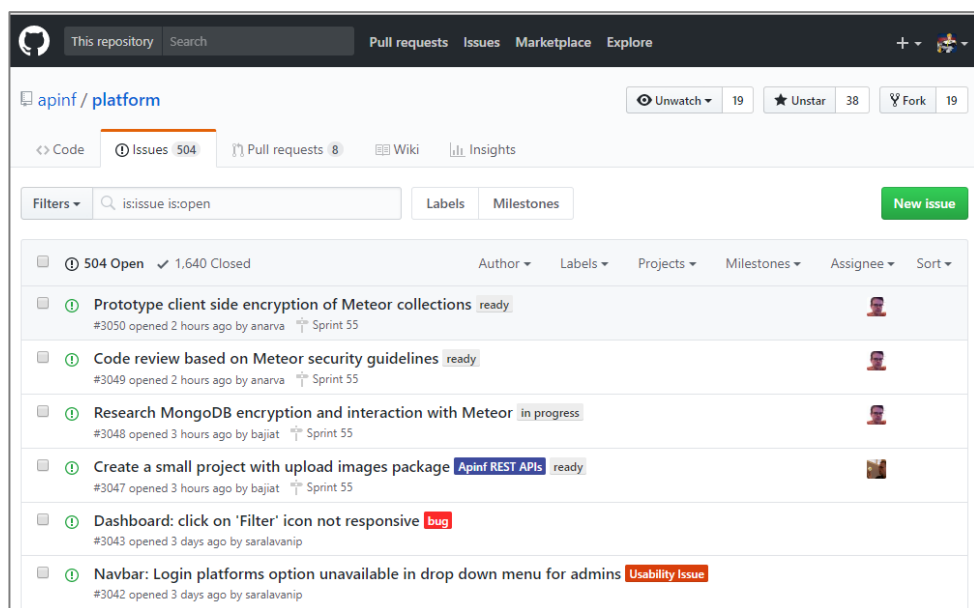
<b>Target Process</b> <sup>1</sup>	Tasks, Backlog, Epic, Feature, User Story, Projects, Sprints	Burn-down chart	Backlog Overview, Time-line View, Dashboard, Kanban board
------------------------------------	--	-----------------	---

## 2.1 General Analysis of the Compared Systems

A brief overview of the general features of the systems presented above for monitoring software or digital service development is discussed below:

- Almost all systems mentioned above have a common unit to measure the minimum amount of work done or that needs to be done to develop a software product or digital service. This unit is known as **issue** (GitHub, Jira, etc.) or **task** (Agilefant, Agilo for Scrum, etc.) depending on the system the term is being used at. Often the systems categorize this minimum work as **design, coding, testing task, bug, enhancement**, etc.

*Figure 2-1 Issues within a GitHub repository<sup>2</sup>*



Platforms like GitHub and Jira often use issues to represent **Epic** level works and this can be done by attaching customizable labels to such issues. This minimum amount of work can be assigned with different status based on what phase

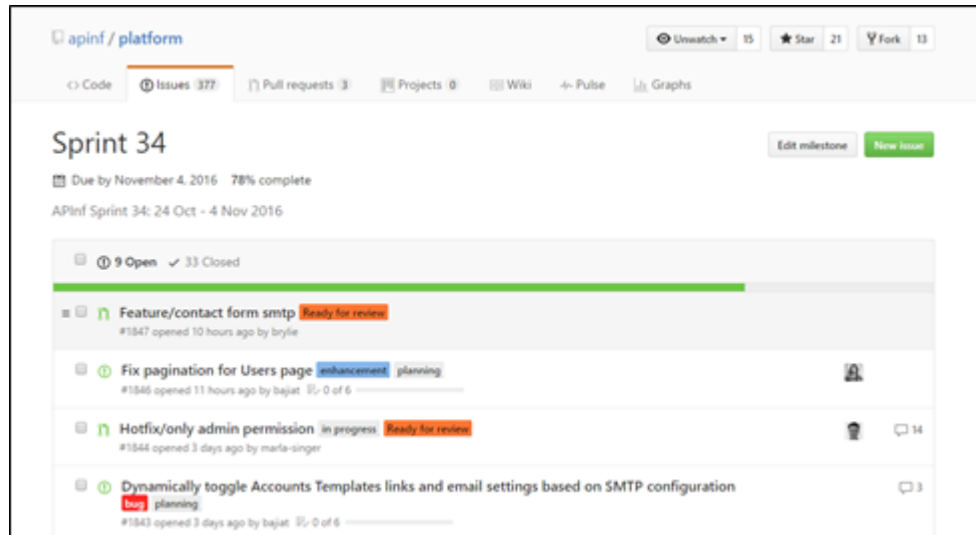
<sup>1</sup> <https://www.targetprocess.com/guide/faq/>

<sup>2</sup> <https://github.com/apinf/platform/issues>

of the development workflow the task/issue is currently at. Commonly used statuses can be **To-do**, **In Progress** and **Done**. However, in most cases these systems provide options to customize the workflow and its phases. When Dashboards, different reports and visualizations are considered, the focus is mainly on issues/tasks according to their category and status for showcasing manual or customizable views.

- **Epic** can be considered as a collection of **User Stories** that are scheduled to be developed, tested and shipped in specific Releases. Under a User Story, a feature of the intended product is described in terms of who would be using it to achieve what goal. A list of issues /tasks is defined under the user story to finish development of that particular feature. In many integration systems or software, these issues under a user story are referred as **Product Backlogs**. This is a Scrum specific terminology. The items from the product backlog that would be implemented in a sprint are called **Sprint Backlogs** for that particular sprint.
- The terms **Sprint** in the discussed systems are considered to be a time-boxed iteration period of 2-4 weeks where certain issues are selected to be implemented, tested and made ready to be delivered as part of the product in an upcoming release. As GitHub doesn't have the notion of scrum or sprint, it defines such a period as a **Milestone**.

*Figure 2-2 Details within a milestone under a repository in GitHub<sup>1</sup>*



Issues in GitHub repository can be assigned to specific milestones for implementation. Systems show alert or notification if the designated period of a sprint has expired and it has not yet been closed or there are remaining issues to be developed. There are views and widgets in different visualizations (e.g. Taskboard, Timeline, etc.) and reports (e.g. burn-down charts) within the above com-

<sup>1</sup> <https://github.com/apinf/platform/milestone/34>

pared systems that are constructed based on the sprints completed or in progress. In many cases, an entire visualization can also be filtered according to current or previous sprints.

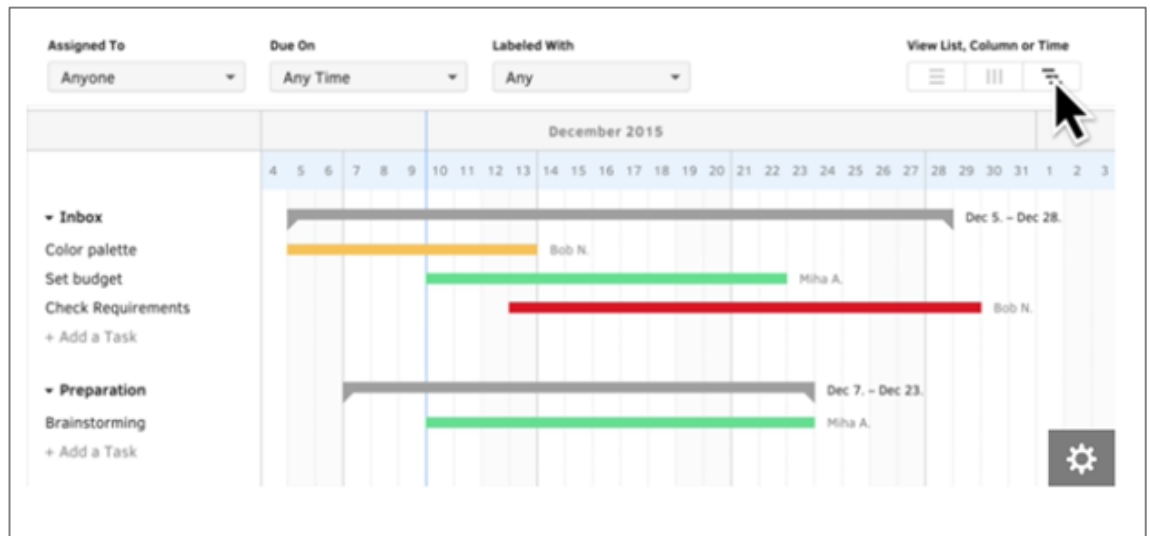
- Almost all software project management or GitHub integration tools include a **Burn-down chart** as a report to monitor and predict project progress. They might also include a **Burn-up chart** report as well. Many of the above mentioned systems have reports feature to generate similar information (e.g. Jira and Pivotal tracker have velocity chart, cumulative flow chart, Epic report, etc.). Usually these reports provide information regarding to issues (change in status, overall completion rate from start to end, etc.). There are also reports that provide cumulative information about specific sprints (Sprint Report in Jira, Iteration Report in Pivotal Tracker) and epics (Epic Report in Jira and Pivotal Tracker).

*Figure 2-3 Burn-down chart for a sprint in Jira<sup>1</sup>*

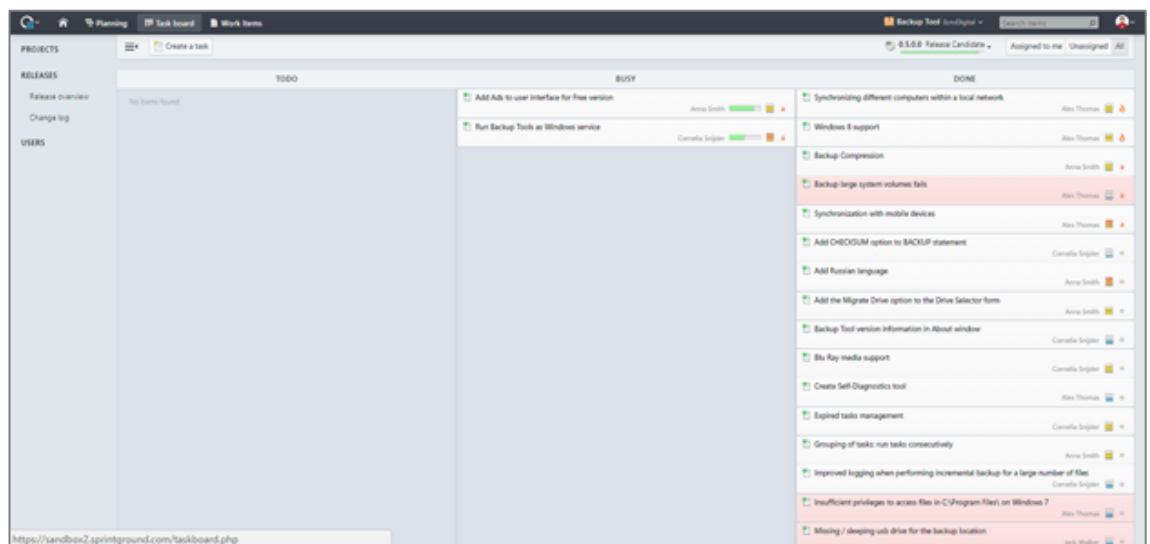


- **Timeline (or Calendar)** is a common overview visualization in many of these systems where the progress of tasks or stories (from their start to completion, including status changes) are shown against time (usually on x axis). The visualization can be shifted left and right to see past and upcoming scheduled items (sprints, milestones, tasks, etc.).

<sup>1</sup> <https://confluence.atlassian.com/agile/jira-agile-user-s-guide/using-a-board/using-reports/viewing-the-burndown-chart>

*Figure 2-4 Timeline view in Active Colab<sup>1</sup>*

- In **Task-boards**, usually user stories or issues are displayed according to their status changes and progresses throughout the development workflow. Often the tasks-boards can be filtered using different criteria like sprints, type of issue, workflow phases, projects etc. Any progress, delay or impediments within tasks can be observed from these boards.

*Figure 2-5 Task-board view in sprint Ground<sup>2</sup>*

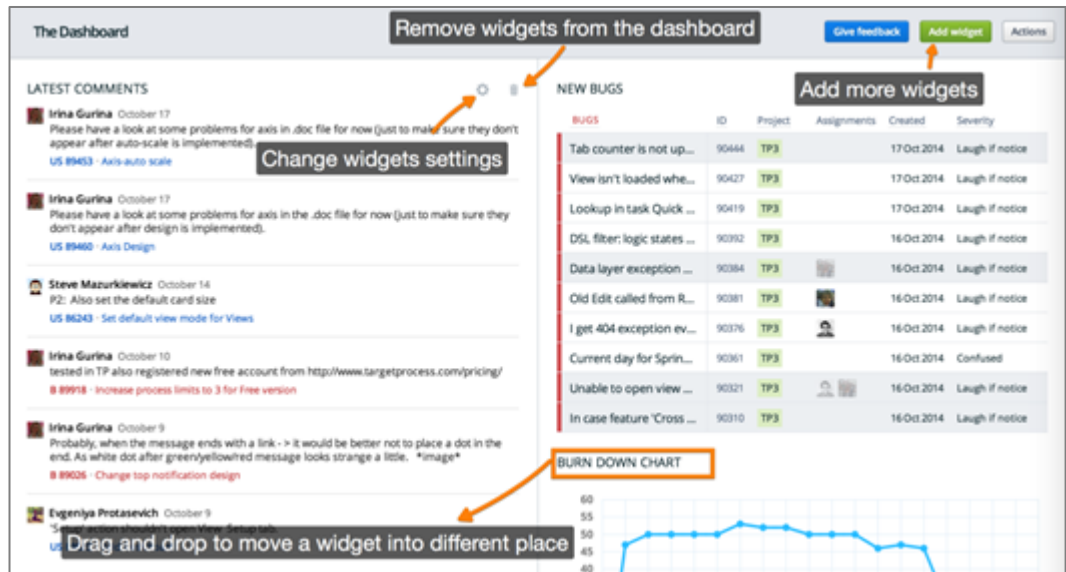
- **Dashboard** is usually the first view a user gets navigated to when logged in to a system. This view usually contains some metrics as widgets from selected stories, iterations, projects, etc. that hold interest to users. The visualizations used

<sup>1</sup> <https://activecollab.com/help/books/activity/calendar>

<sup>2</sup> <https://sandbox2.sprintground.com>

in the Dashboard can be simple numerical figures explaining a metric or charts (bar, pie or line) showing some trends.

**Figure 2-6** Dashboard View in Target Process<sup>1</sup>



## 2.2 GitHub Integration Tools

The above mentioned project management platforms might or might not make it possible to integrate GitHub repositories with them. However, there are some recommended integration tools by GitHub which have the above features. Their different visualizations and generated reports are based on the issues of one or more connected repositories. Some of the example integration tools are: ZenHub<sup>2</sup>, Waffle<sup>3</sup>, Blossom<sup>4</sup>, FogBugz<sup>5</sup>, etc.<sup>6</sup>. A common feature in all these tools is to visualize GitHub issues of a specific repository as cards and display them in Scrum, Kanban or any other Agile development workflow process the team prefers. Features of GitHub (assignee, labels, milestones, etc.) can be added or modified using these cards. Modifications made to the issue cards (i.e. adding comment, changing pre-populated labels, etc.) also take effect in the original repository. The tools also have provisions to connect and sync issues with Git pull request. Cues in commits and pull requests can be used to move issue cards automatically around the visualization boards (e.g. FogBugz allows to create bug events when pushing commits to GitHub Repository. This allows tracking code from FogBugz and provides context for the whole team about the change-set committed). These tools

<sup>1</sup> <https://www.targetprocess.com/guide/boards/dashboards/dashboards/>

<sup>2</sup> <https://github.com/marketplace/zenhub>

<sup>3</sup> <https://github.com/marketplace/waffle>

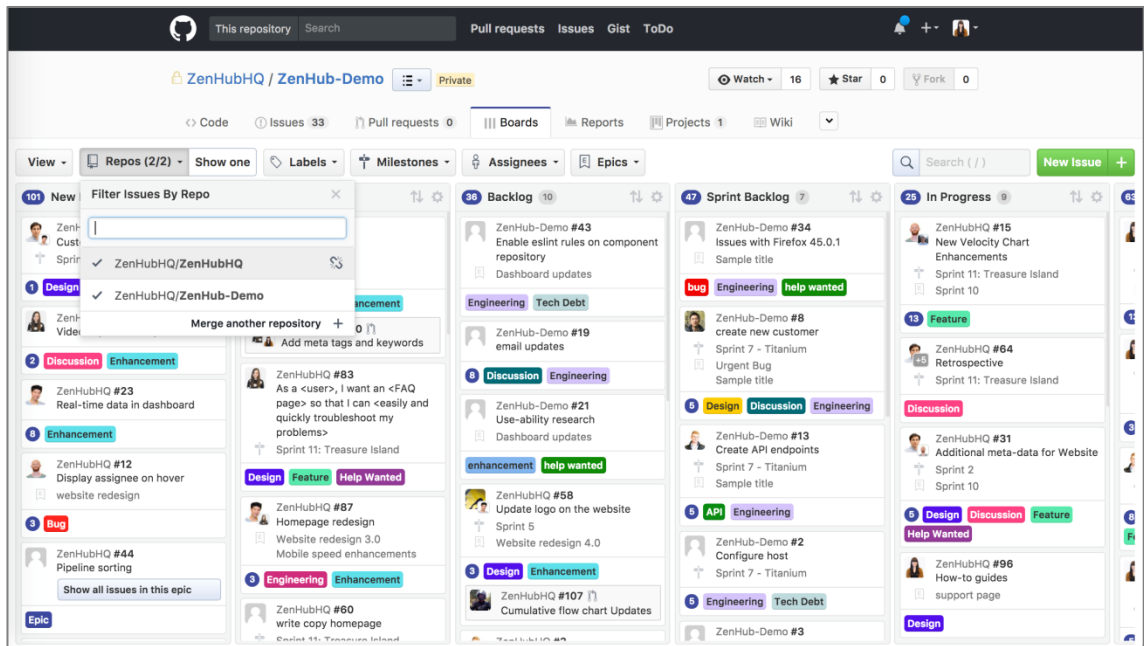
<sup>4</sup> <https://www.blossom.co/features/github>

<sup>5</sup> <https://blog.fogcreek.com/fogbugz-github-integration/>

<sup>6</sup> <https://github.com/marketplace/category/project-management>

keep track of branches made to a specific repository and status of pull requests made in GitHub.

*Figure 2-7 Task-board view in ZenHub<sup>1</sup>*



Issues located in GitHub originally have no means to add estimation. However, these platforms allow users to add pseudo-estimated task time via cards. The time estimations are usually made according to Fibonacci series (1, 2, 3, 5, 8, etc.). It should be noted that added estimations via these tools don't appear in the original GitHub Issues.

Often, the integration tools support Milestone-integrated burn-down charts, as estimations can be added to a card representing a specific issue.

### 2.3 Advantages and Limitations of the Discussed Systems

As Dipor Dashboard primary focuses on integrating GitHub repositories into the intended system, it is important to understand what facilities does GitHub, as an independent system, provide to project development progress of digital service implementations following Agile development methodology. Also we need to identify the limitations of GitHub that might be barriers towards functioning as an independent service fulfilling the needs of Dipor Dashboard. In addition, we also need to learn about the advantages and limitations of the discussed systems and platforms as potential competitor of Dipor Dashboard. This is important to comprehend why a new service or system is needed where multiple solutions in the market exist.

<sup>1</sup> <https://github.com/ZenHubIO/support>



### 2.3.1 Advantages and Limitations of GitHub

GitHub itself is a good solution if it is used as a version control system and code repository for open source development works. The free version of GitHub has sufficient features for a development team to conduct their works. The Pulse feature of any GitHub repository shows overview about how active the repository has been over a selected period of time. They include both proposed and closed pull requests, open and closed issues, most active members. Also many of the discussed project management systems above can integrate GitHub repository and can control repository issues and pull requests (Dugas, 2014) (Dewalt, 2016).

However, GitHub lacks repository specific dashboard and graphs specifically useful for detecting progress of a service or software development. GitHub doesn't explicitly support Agile development as there is no way to fix duration for an issue to complete or to add some numeric value of estimation. GitHub doesn't have any burn-down chart which is an important graph to track down work progress within a sprint. Also it doesn't contain any Kanban like visualization to show which issue is in which phase of the workflow. There is no specific process workflow unless a pseudo appearance by using default or user created labels is given. It also doesn't include any timeline visualization to indicate changes in issues (open, closed, or according to default or customized labels). The graphs "Commits", "Code Frequencies" and "Contributors' Graph" is not sufficient enough to predict Progress. They mainly indicate what amount of code has been changed, added or deleted from the master branch as well as private branches of the contributors. Addition/Deletion of code can be done because of code refactoring. But it doesn't necessarily mean that a user story or task has been completed and closed or a bug has been fixed. In Addition, Integration with different software for agile workflow management would require managing items in two places. Also the integrated software might not be free of cost.

### 2.3.2 Advantages and Limitations of the Discussed Systems for Development Progress Monitoring

The advantages of the other discussed software or digital service management and monitoring platforms (Jira, Agilefant, Sprintly, etc.) are mainly with their ability to work independent of GitHub, executing Agile methodology workflow and varieties of visualizations and graphs for selection. Most of the discussed systems and services have a dashboard like setting which users can customize to add reports, statistics and charts in order to determine how a service development is progressing. Users can select multiple sprint specific information to be added in the dashboard. So this is suitable for Dipor Dashboard related projects which usually are continued for three months' time period (either six two-week sprints or three one-month long sprints). Aside from dashboard,

most of these systems have timelines that usually show how a smallest unit (e.g. task, issue, etc.) have changed (e.g. change in workflow status or task category) against time throughout its lifespan (e.g. definition, starting progress in a sprint, completion, etc.). Timeline can be used to show when some specific sprints have taken place in a project's lifespan. They can also be filtered if the systems have multiple ongoing, upcoming and completed projects. Task-boards within these systems often provide a Kanban like visualization where users can determine if development progress is going on smoothly or any bottleneck has arisen. Last but not the least, plenty of reports and graphs (e.g. Burn-down chart, Burnup Chart, Velocity Chart, etc.) serve as powerful means to predict forecasts about whether or not the entire development work would be completed on time with current estimation and progress rate.

The primary limitation of the discussed project management systems and services is that almost all of them are not open source. So there aren't possibilities for customizing the forked systems and adding features to it by independent vendors or users to fulfill their specific needs in digital service development. Often the systems discussed require manual setting up and integration in the workspace. That might be a difficult challenge for people with little or moderate technical expertise with digital service development. Last but not the least, often features like Dashboard and most of the report generation features are not available in the free and trial versions. The license to use the entire set of the features need to be purchased and renewed when expired. Also, price of the license may vary based on the number of users and often turns out to be expensive.

GitHub integration tools often provide system codes that could be forked in their related GitHub repositories. Developers can fork the code and make some customizations. But the full-set feature of the integration tools is not often included. People have to purchase the license in order to access those additional features. And as discussed above, forking, customizing or even manually configuring those tools within personal workstations might become too complicated for users with novice skillsets.

### **3. EXSTING WORKS ON DASHBOARD DESIGN, AGILE DEVELOPMENT VISUALIZATION AND GITHUB REPOSITORY ATTRIBUTES**

In order to evaluate a new system, it is important to search and study existing works done to propose appropriate design and possible functionalities of systems or services intended for similar use. This not only reveals limitations of the scope of work done, but also provides possible guideline to develop a concept that would try to address many of the limitations (if not all).

The topic of the thesis requires exploring three specialized fields that are vastly expanded on their own: Dashboard Design, Agile Development Visualization and GitHub Attributes to design development workflow. For the thesis work, interrelation between these three subject matters is that, a Dashboard needs to be designed which incorporates and visualizes information obtained from GitHub to project progress of Agile Software Development works. As Dashboard is a generalized concept and can be implemented differently based on the field of application and problem, it is beneficial to search for useful design principles for developing a concept to monitor software or digital service implementation.

Agile Development methodology contains many development frameworks characterized with its area of focus. There are frameworks focusing on practices (e.g. eXtreme Programming - XP), on workflows (e.g. Scrum, Kanban, etc.) or on requirement specification and development activities (e.g. Feature Driven Development - FDD). However, the thesis work focuses on software projects developed using Scrum or Kanban framework. Also both of the frameworks have concepts of workflow, burn charts, cumulative flow diagrams, etc. which provide information about overall software project status. It is important to determine which of these two frameworks are more suitable to be used in projects that use GitHub as version control and issue management system. In parallel, it is also necessary to search for existing literatures that uses Agile workflows in dashboard like visualizations to show project status.

GitHub has its own visualization graphs, pseudo workflows specific to each repository. In addition, a set of repositories build an “organization” in GitHub with overview cards showing an overview graph of works done in past year. However, the graph is usually generated from code commits and may not be the ideal to indicate development progress in terms of Agile workflow. It should also be considered that projects under Digitalvelutehdas community have a 3-month long development circle before demonstrat-

ing a minimal viable product. So it is necessary to search for existing work done to use GitHub repository information to construct Agile workflow process.

### 3.1 Dashboard Design

The metaphor of dashboard usually comes from the panels that drivers face in automobiles, airplanes, etc. This panel contains instrument and controls. The information in the panel appears in such a way that it is easier for the driver to comprehend the current status of the vehicle and to take quick decision about what to do next. Interestingly, most dashboard definition found in the 1st page of Google search is stated in business context. According to Wikipedia<sup>1</sup>, dashboard provides at a glance view of key performance indicators (or KPIs) that is relevant to a particular objective or business process. For example: sales, marketing, human resource, etc. In the Wikipedia article, there are discussions about dashboards' classification (e.g. strategic, analytical, operational and informative). Also this includes how information is displayed and the level of details. However, the definition and description are too generic to point it out towards software development and is primarily based on business aspect.

A definition stated in TechTarget somewhat gives an idea about dashboard in relevance with information technology (Rouse & Sorenson, 2005). According to the definition, a dashboard provides a user interface to organize and present information so that it has easy readability. Certain dashboard may aim to integrate information from multiple sources into a single, unified projection. Usually dashboards designed for Information systems are interactive and can be customized as per user's needs. Although this definition is related to information system, this doesn't include what information to include within a dashboard and how they should be presented. Also the definition doesn't specifically point out to software development.

Stephen Few is an IT innovator and consultant with his contribution towards data visualization for analyzing and communicating quantitative business information. In his book titled "Information Dashboard Design – The Effective Visual Communication of Data" (Few, 2006), Stephen Few has given the following definition about Dashboard:

"A dashboard is a visual display of the most important information needed to achieve one or more objectives that has been consolidated on a single computer screen so the information can be monitored at a glance."

The definition contains some important aspects when considering characteristics of a dashboard:

---

<sup>1</sup> [https://en.wikipedia.org/wiki/Dashboard\\_\(business\)](https://en.wikipedia.org/wiki/Dashboard_(business))

1. Dashboard should be comprised of graphical elements and texts in such a way that information can be communicated more comprehensively than what they would do on their own.
2. Related or non-related data and facts, generating from multiple sources, should be present as a collection in dashboard to give an overview of current system status.
3. Information should appear in a single view (irrespective of screen size) within the viewer's eye span, so that it can be seen at a glance.
4. Dashboard should be auto updated and point to elements that require attention and quick actions.

Although it is descriptive and comprehensive enough, the definition is still too generic to define the need and requirements of different specializations. Also the definition directly doesn't mention that it should be possible to customize dashboard so that it can tailor needs specific for a group of people or functionality. So considering about the purpose and need of Dipor dashboard, we can define the characteristics of what a dashboard for monitoring Agile development progress can have:

- The dashboard should be able to obtain different data and facts out of a GitHub repository.
- The information should be consolidated and presented in a way that it mimics Agile metrics providing accurate facts and figure.
- The dashboard should show GitHub related information and customized graphs that indicate development status in a single view. If needed, there should be navigation toward new view to see detailed analysis.
- The dashboard should be able to indicate if some particular aspect needs attention and aspect by the means of different GitHub attributes.
- The dashboard should be customizable in terms of adding and showing information, their depth, graphical elements to represent that information.

Apart from real life implementation of software and platform for Agile development work, very few research works dedicated solely for designing Dashboards for monitoring Agile development process has been found. There are however, plenty of guidelines regarding to what "dashboard should do", "how it should work", "what it is not", etc. In the whitepaper "6 Best Practices for Creating Effective Dashboard" published by Tableau, emphasis has been given on the following aspects (Cotgreave, 2011):

- Obtaining data from any available sources (on premise warehouses or cloud) with provisions for secured access and fluent interaction.
- Blending both qualitative and quantitative data from multiple resources to get a single, holistic view on performance, tracking completion of goals, etc.
- Designing metrics that contribute to an objective and answer question about what is happening and why is happening. Dashboard should be selective to include such metrics.
- Making dashboard easy to read and understand by providing appropriate visualization (a single number, visualization than pie chart, varieties in visualizations), current data-feed and interactive interactions. Adding time trend in dashboard makes it easy to predict forecasts.
- Making simple browser based distribution for easy sharing of the dashboard.

Stephen Few has pointed out what dashboard is not and what it should not do (Few, 2006). A dashboard is not an analytic platform or a new technology that gives detailed data explaining the occurrence of an event that required attention. A dashboard can point to a separate view that would present detailed information. If a dashboard doesn't add right contexts to key measures, they won't be pointed out directly to enlighten the needed action. Visualization of information need to be encoded using quantitative scales with sections indicating which part of the section is good or bad.

The above stated characteristics are true for dashboards. However, the generalization is made for dashboards of all specialization fields. Especially when considering Agile methodology, it is important to realize and select which metrics would be suite the best in a dashboard to indicate development progress and what visualizations would be appropriate to represent them.

FASTDash (Fostering Awareness for Software Team Dashboard) is a system developed with User Centered Design approach to monitor team activity and improve team awareness (Biehl, Czerwinski, Smith, & Robertson, 2007). This system highlights activities a development team is currently engaged in by spatially representing the shared codebase. Suitable for a team of 3-8 developers, FASTDash helps team to access key information like which member checked what source file(s), which files are being viewed, what method and classes are currently being modified, potential conflict situations, etc. Team members can provide status details to activity information by adding annotations to the visualization. The system can be projected in a large display at a share workplace or personal screens. However, one of the limitations of FASTDash is that it is more suitable for technical people and hardcore developers. Also the discussed implementation

supports an internal system of Microsoft and Team Foundation Server as Source Control Management only.

Awareness 2.0 is a software development project where a dashboard component, along with a feed system under IBM Jazz IDE was used to comprehend overall project status, work contribution of individual or entire team to the project and identify bottlenecks within current tasks (Treude & Storey, 2010). The dashboard component could be customized for individual developers, an entire team or for an entire project. Customizable widgets were used to display information regarding different project aspects (e.g. current workload, members in a team, etc.) by configuring related parameter. Project management used the dashboard for making comparison between different teams' activities. Teams used the dashboard for tracking work items, detecting bottleneck, getting real-time changes on prioritized works, being aware of other team's activities and as a shortcut to more detailed information of a project attribute. However, there was no clear indication of from where and how data was populated into the Dashboard view. Also the platform was not suitable for open-source development. In spite of the above limitations, the user study conducted in this project indicated that awareness about a project's status could often be difficult to comprehend and needs additional tool integration to improve the process.

Since the idea is about creating a dashboard specifically for projecting Agile Development progress from GitHub repository, we need to evaluate its usability in terms of meeting requirements and needs defined in User Requirements. We also need to determine how well the developed platform serves its purpose to give overview of development work status and forecast about project work. Heuristic evaluation specifically used for information visualization can be helpful in determining whether or not a dashboard is capable of serving its purpose.

### **3.2 Agile Development Visualization**

Agile software development focuses to manage complexity and uncertainty of implementing software solutions for a problem. To do so and to adopt with fast changing business demand, Agile project management makes product feature delivery by short incremental and iterative development cycles. Also this promotes continuous integration of code changes (Ruhe & Wohlin, 2014). With a broad range of software development lifecycles, this project management platform helps to evolve requirements and their solution via the collaboration of self-organizing, cross-functional teams. Scrum and Kanban are among these development lifecycles that are primarily based on managing the flow of work.

Scrum uses iterative and incremental process to develop and manage software products. The framework of scrum consists of roles (product owner, team and scrum master), ceremonies (sprint planning meeting, sprint daily meetings and sprint review meetings) and artifacts (product backlog, sprint backlog and burn-down chart) (Cho, 2010). Items that would be implemented in current iteration or sprint are estimated by assigning some number of work hours to each of the items. To forecast project completion, the backlog of work hours needs to be divided with velocity to determine number of remaining iterations needed for project completion. Velocity is calculated by crediting work hours of a “Done” (all tests marked as passed) feature to the iteration were the feature was completed (Karlesky & Vander Voord, 2008). Using the velocity of 2-3 recently completed sprints/iterations, a timeline can be generated by using velocity as a weighed historical average of most recent sprints/iterations. The timeline can be either a Burn-up Chart or a Burn-down Chart. A sprint burn-down chart projects the amount of work needed to be completed before the end of the current sprint. Sprint days are shown in horizontal and remaining work hours are shown in vertical axis. Updates are made to the chart by aggregating the estimates of remaining work for all tasks in the sprint backlog. Possibility of the team finishing the committed tasks by the end of the sprint can be determined from the trend line of work hours remaining (Mahnic & Zabkar, 2012). A burn up chart is used to show functionalities built up over a period of time. Burn up charts keep track of works by tracing completed work and total work in separate lines (“What is a burn up chart?,” n.d.). The distance between these two lines indicate the total amount of remaining work. The project is completed when both the lines meet with each other. The total work line helps to determine if the project is incomplete due to slow work progress or too much addition of new work.

Using story-points, complexity or time estimations with GitHub issues is difficult. GitHub doesn’t have any specific means to associate the above estimations directly to the issues if it is used on its own. Many other GitHub integration tools (as discussed in previous chapter) have individual estimation features (story points, time estimations, etc.). When GitHub issues are visualized in those platforms, the allocated estimations appear together with them. However, those estimations don’t appear anywhere in the real GitHub issues. In addition, velocity estimation to determine predicted completion of a project works more accurately when the project has a long duration and the development team has a handful of completed sprints for calculation. However, with small projects having a fixed 3-month long duration, this is redundant.

Another popular choice of software development methodology is Kanban. Using this methodology, software development work is executed by visualizing the workflow, limiting the work in progress (WIP) in every phase of the workflow and measuring cycle time (Ahmad, Markkula, & Oivo, 2013). The aim for this methodology is to communicate priorities, highlight bottlenecks and maintain a constant work release by developing what is requested. Work wastage is reduced by allowing developers to focus on few



items at a time and creating shorter feedback loop from customers. The technique to visualize workflow in Kanban development is to use a Kanban board. Each phase of the development workflow is added as a column on a Kanban board. Cards are used to indicate units of work that needs to be implemented. The cards are placed under related column according to their status within the development process. At the very minimal level, the phases can be To-Do, In Progress and Done. However, each team may follow their own workflow process which can include additional phases (e.g. Backlog, In Review, Testing, Archived, etc.) along with the general ones. So Kanban provides the opportunity to map work process that is unique to individual teams.

A simple physical Kanban board can be setup in the work premise with a white board and sticky to visualize phases in the workflow and represent tasks respectively (LeanKit Inc., 2015). For teams working in remote locations, it is mandatory to setup such a board in web based systems (e.g. Trello, KanbanFlow, etc.). So that team members irrespective to their geo-location can modify task cards, change status of tasks and track progress of the work (Paredes, Anslow, & Maurer, 2014). Proper customization of the board and the task cards (e.g. adding an id, small description, rough estimations in work hours or other metric, assignees, etc.) helps communicate details at a glance. The most important aspect of Kanban methodology is to limit Work In Progress (WIP) in order to match capacity of the work flowing through the process (“Kanban WIP limits – Work in Progress limits | Kanban Tool,” n.d.). Each phase can accommodate limited number of tasks to ensure smooth flowing of work across the phases and preventing bottlenecks. WIP limit can be applied as a constraint to individual or all phases of Kanban process. This restricts the number of works that can reside within a workflow phase at a given time. However, WIP limit can also be set for per developer or the whole team. Setting WIP limit for a column (or a person) doesn’t restrict to assign more tasks into. Rather it helps team to take responsibilities to determine the reason behind WIP exceed and improve their work efficiency to prevent such occurrence in future.

A simple Kanban flow can be implemented using existing GitHub features to track service development works. Kanban adds value by projecting real-time visualization of work status (Ikonen, Pirinen, Fagerholm, Kettunen, & Abrahamsson, 2011). As WIP limit in Kanban allows detecting bottleneck in workflow phase, it helps to determine what work needs team’s attention. Also estimating the works in good but not required. Even so, a pseudo burn-down chart can be created based on when GitHub issues are closed (or in Done phase) for a specific sprint. So adopting Kanban methodology with GitHub features seems an ideal approach.

In the work presented in (Nakazawa & Tanaka, 2016) and (Nakazawa & Tanaka, 2015), a web based digital tool had been developed to visualize Kanban workflow with additional functions of showing and limiting WIP of individual developers in a team. The implemented Kanban board is divided vertically according to 6 workflow stages. In

addition, it is also divided horizontally in To-do, Doing and In Review phases for each developer in the team. The WIP for each developer is calculated as a sum of number of tasks in To-Do, Doing and In Review phases. WIP limit for each developer can vary and no developer can have number of tasks that crosses his/her individual WIP limit. This tool can also be synchronized with GitHub workflow and issues. However, GitHub doesn't have any explicit WIP functionality. So from the repository itself, a member can be assigned with task number that is greater than the WIP limit. So in actual case, WIP functionality of this system really can't affect GitHub.

### **3.3 GitHub Attributes to Monitor Development Progress and Visualize Agile Workflows**

The POC for Dipor Dashboard required integrating GitHub repositories of digital services that were being implemented following Agile development methodologies. From Competitor Analysis, we have come to know that GitHub itself contains some measures to view Milestones progress and graphs to show contribution, code frequency, commits, etc. However they alone are not self-sufficient to mimic an Agile workflow. A crucial factor is that, unless using an integrator, there is no way to estimate an issue by assigning some development hours to complete it and to monitor progress of the issue. Moreover GitHub also doesn't have visualizations like workflow or Burn-down charts which are important in Agile development to see work status and overall development progress. To achieve this, GitHub repositories need to be setup with other integration/project management tools. It needs to be sought out that if there are solutions which suggest using GitHub as Agile Project Management tool on its own and if the solution provides a Dashboard solution to get overview information of the project status.

In the following study conducted at University of Tampere (UTA), a guideline for using GitHub native features as Requirement Management in software development using Lean approach was proposed (Salo, 2014). The solution relied on using hierarchy while creating issues and tracking those issues with naming conventions and references. Agile method ensures quality code with incremental development following feedback for re-adjusting user requirements. Lean approach helps to apply the methodology in large scale team works (Dan, 2010). However, the study didn't propose any visualization technique to project the development workflow. It didn't contain any suggestion about determining status of development with issue traceability.

Many articles and blogs of different organizations have often described how they use GitHub. In Ian Bicking organization, milestones and labels are extensively used to determine what work needs to be done in current iteration, what should be done next and what work should be left for distant future (Ian, 2014). However, their process of label and milestone creations and assigning issues with them seemed to be complex and time consuming. Moreover, the issues are simply sorted without following any Agile workflow. The usability of the process has never been evaluated. In addition, the use of traditional GitHub repository options has been retained with its limitations as a dashboard.

An article emphasized on relating issues to each other and using GitHub Labels and Milestones extensively for project management (Bitner, 2012). However, it lacks ideas about how project can be monitored to determine its progress. Moreover, there is also suggestion to use a GitHub integrator to achieve a Kanban like work-process.

There have been suggestions to use GitHub integrators to get an Agile workflow, use GitHub directly to manage issues or to build customized tools based on GitHub API (Dugas, 2014). However, there is cost associated with using GitHub integrators to access full feature set to manage the workflow. The suggestion to build own software for Agile framework in order to support scrum masters and product owner opens possibilities of creating platform to monitor development progress and get overview of the big picture. So this approach complies with the need of creating Dipor Dashboard.

A lightweight Agile framework using GitHub had been suggested in this Article. (Dewalt, 2016). The article proposed mapping of Agile components into specific GitHub Attributes. User stories were mapped into Issues. Sprints were mapped into Milestones. Global Backlogs were marked as open issues that are unassigned and not under a milestone. Lastly Sprint Backlogs were mapped into open issues that were unassigned but specific milestones. All issues within GitHub were to pass through an Agile workflow consisting five phases: “Global Backlog”, “Sprint Backlog”, “In progress”, “In Review” and “Closed”. Under Global Backlog would rest all open issues in the repository. When issues are assigned within a specific milestone for implementation, they were to move under Sprint Backlog. Issued would be placed under In Progress when they were assigned with collaborators. When pull requests (PR) were associated for reviewing code associated with specific issues, they were to move into In review phase. The issue gets closed within GitHub and moves to Closed phase when the associated PR is merged into the repository. The process might not be suitable to determine how work has been progressed throughout a milestone as issues in GitHub can't be estimated. However, if properly visualized, a Kanban workflow could be projected to show how many issues are under each phase in a given time. If the concept of Work in Progress (WIP) is adopted, bottlenecks in different phases could be identified if issues are stacked into a phase for a long time and not moved into the consequent phase. A

pseudo burn-down chart could be compiled based on creation and closing dates of GitHub issues. But that requires further studies to determining the feasibility the above solution.

## 4. RESEARCH METHODOLOGIES: PHASE 1 – CONCEPT DEVELOPMENT

This chapter presents the details of the methodologies that were used to develop the concept of Dipor Dashboard. This was the first phase of the empirical works conducted for this thesis. Phase one included initial interviews, iterative creation and evaluation of low fidelity prototypes in subsequent interviews with intended users and constructing affinity diagrams to identify user needs out of the system to be implemented. To make the work easier to manage, the entire phase one of empirical work was divided into three sub-phases or events. Sub-phase one involved interviewing the main customer to understand the purpose and need for implementing Dipor Dashboard. Sub-phase two was used to conduct interviews of recommend personnel and testing initial design ideas with them. Sub-phase three involved consolidating the outcomes from sub-phases one and two. Results of phase one included personas of the intended system, low-fidelity prototypes of the dashboard visualization for Dipor Dashboard portal and a set of user requirements defining the overall concept.

The main development work for Dipor Dashboard had already started before the empirical research began. The developing company (Sampo Software Oy) had designed an outline of the look-&-feel and the functionalities of Dipor dashboard based on the software requirements approved by the Ministry of Education and culture on behalf of the Digipalvelutehdas community. The thesis work was initiated by interviewing the main customer (Head of Development, department for General Education and Early Childhood Education, Ministry of Education and Culture of Finland; he presented the idea of Dipor Dashboard to Digipalvelutehdas community) to get more details of about the purpose of Dipor Dashboard and to know about possible user groups of the intended system. This was the first sub-phase of the entire phase one empirical work and resulted with creation of four personas for the intended system. The construction of several affinity diagrams was also initiated in this sub-phase. The second sub-phase involved interviewing several intended users recommended by both the customer and the developing company. The purpose of these interviews was to get more insights on user needs for Dipor Dashboard and evaluating low fidelity prototypes representing initial concept of the dashboard view. Affinity diagrams projecting users' working goals, challenges to meet those goals and expectations out of a new service were also being updated in parallel. The created personas were modified when needed. The third sub-phase involved self-explorations within the collected user information and feedback. The results of this

sub-phase included finalizing the dashboard concept with low-fidelity prototypes, completing the personas and defining user needs and requirements for Dipor Dashboard.

The following table shows the timeline of the three sub-phases that took place in order to develop the concept of Dipor Dashboard.

*Table 4-1 Timeline of Phase One in Empirical Work*

<b>Event</b>	<b>Conducted work</b>	<b>Time Period</b>
<b>Sub-phase 1</b>	<ol style="list-style-type: none"> <li>1. Interviewing customer</li> <li>2. Initiating persona creation</li> <li>3. Initiating affinity diagrams constructions</li> </ol>	14 February, 2016 – 1 <sup>st</sup> March, 2016
<b>Sub-phase 2</b>	<ol style="list-style-type: none"> <li>1. Creating initial low-fidelity prototypes</li> <li>2. Interviewing intended users referred by customer and developing company</li> <li>3. Evaluating and modifying low-fidelity prototypes iteratively</li> <li>4. Updating affinity diagrams</li> <li>5. Updating persona</li> </ol>	7 March, 2016 – 15 April, 2016
<b>Analyzing Results and Finalizing outcomes.</b>	<ol style="list-style-type: none"> <li>1. Finalizing affinity diagrams</li> <li>2. Finalizing the concept of dashboard with low fidelity prototype</li> <li>3. Defining user needs out of the intended system</li> </ol>	18 April, 2016 – 29 April, 2016

#### **4.1 Sub Phase 1: Customer Interview**

After the confirmation to conduct the thesis work in parallel with the development of Dipor Dashboard was received, the empirical work started with interviewing the customer for Dipor Dashboard. From Sampo Software Oy, some initial ideas of what to implement was received by visiting their office premise. However, it was necessary to understand why a new service like Dipor Dashboard is needed to be implemented amidst many existing solutions in the market. It was also required to acquire contact information of intended users to conduct the user centered design (UCD) work for the thesis. Opinions and feedback from the customer was considered to be valuable source of information.

The interview with the customer was conducted at the Tampere Demola premise. Before initiating the interview, the customer was made aware about the purpose of the interview. Written consent was taken to make audio recording of the conversation. The

interview was conducted in a semi-structured process. The reason why semi-structured interview was chosen is because it makes data collection easier when user behavior can't be observed directly (Wilson, 2013). Interviews taken in such structure also help to understand user goals, allow probing and clarification for complex topics and allow both the interviewer and interviewee(s) to be flexible and relaxed. The duration of the interview was approximately an hour.

#### 4.1.1 Interview Questionnaire

An initial questionnaire was prepared to control the flow of the interview. However, subsequent and follow-up questions were asked only if they were appropriate with the context or needed to get more details. As the topic of the thesis work was still new, questions prepared for the customer interview were targeted to acquire background information on purpose for developing Dipor Dashboard, its possible users and goals to be achieved using the new service. Question categories focused on stakeholders, type of services that would be monitored, process management and technologies used to monitor service development in present, Challenges in meeting work goals and expectations out of a new system. During the interview, efforts were made to keep the questions open ended in order to probe more information.

The initial questionnaire was reviewed with by the thesis supervisor. Necessary modifications were made according to the received feedback in order to make the questions more precise. This helped in finishing the interview on time. The final version of the questionnaire is presented below:

*Table 4-2 Customer Interview Questionnaire*

<b>Stakeholders</b>
1. What is your role in the organization you are associated with? What are your responsibilities in your work? What duties do you perform in Digipalvelutehdas Community?
2. What are the general professions held by Digipalvelutehdas community members?
3. In your, opinion, what might be the expertise of the potential users of Dipor Dashboard?
<b>Information on Service Development</b>
4. Can you please explain how the 90-days-service development system works?
5. What are the factors (e.g. service type, timeline, purposes, finance, resources, etc.) you consider while monitoring different service development works?

6. If not confidential, can you please give an example of an ongoing service development you are managing?
7. Currently, how do you measure the success criteria of a developed service?
<b>Monitoring Process and Technologies</b>
8. What is your strategy to monitor development work of digital services? What are your reasons behind adopting this strategy?
9. What are the system / platform / application you are using currently to monitor such development work? Are they the same for all the intended services?
10. What data usually you seek to obtain status information about service development work? What is the source of such data? Do you make any statistical analysis on the obtained data? If, so what it is and why?
11. Do you make comparisons between the development statuses for two services? If so, what factors do you consider while making the comparison?
12. What are your preferred types of visualization to see the obtained data for monitoring purpose?
13. Do you need to create report and the progress in service development work? If so, how do you represent the obtained information? How do you feel about this reporting process?
<b>Challenges and Limitations</b>
14. Do you face any challenges on the strategy or technology that you have adopted in monitoring service development works? If so, what are they?
15. What limitations have you observed in the software / platform / application that you are currently using for monitoring purpose?
<b>Expectations out of the Intended Service</b>
16. What do you wish to see in a new service to monitor development work of digital service implementations? How do you think it will make improvements in your adopted strategy for monitoring such work?



## 4.2 Sub-phase 2: User Interviews and Evaluation of Low-Fidelity Prototypes

Second sub-phase for developing the concept of Dipor Dashboard involved interviewing several other recommended users and getting their initial feedback about the intended system from low fidelity prototypes. At this phase, the initial set of questions was modified to include more specific queries as the need of the intended Dashboard portal was clearer. The constructed affinity diagrams and personas were modified if needed as more interviews were conducted. To give the thesis work a more focused scope, it was also determined in this sub-phase to concentrate specifically on the dashboard part of the entire web portal.

From the customer interview, the purpose for implementing Dipor Dashboard portal was comprehended. But it was important to reach out actual intended users of a new system to understand their needs. This aimed to incorporate the collected needs as design in the actual implementation process (MAGUIRE, 2001). Interviewing intended users helped to clarify if initial system requirements defined by customers or developing team are consistent to specific user needs. This also provided a good opportunity to evaluate the designs and functionality of the intended system using low fidelity prototypes for this thesis work.

### 4.2.1 Participant Recruitment

From the customer interview, contact information of several people associated with Digipalvelutehdas community was obtained. The customer referred them as possible users of Dipor Dashboard system. Access to Digipalvelutehdas Slack channel was given to the thesis worker to communicate with the related personnel and arrange for interviews either in person or online. Initially eight people were contacted for interview purpose via Slack. Response was received from four of them. One of the potential participants was excluded as he claimed that he's not closely associated with the community and won't be using a dashboard portal for monitoring service development. The rest of the participants agreed to give interview and their feedback about the possible design of the Dipor Dashboard portal. The customer also referred to an additional interviewee. This interviewee was not directly associated with Digipalvelutehdas, but had affiliation with the customer from Demola works. The customer considered his/her experience and insights about project development monitoring to be valuable. In addition, one more participant was recommended by Sampo Software Oy as a potential user of Dipor Dashboard.

It was also decided by the thesis worker to interview a domain expert in software engineering to get some concepts on how to monitor development progress using appropriate metrics and visualizations. The chosen interviewee was a faculty member of the Pervasive Computing department of Tampere University of Technology. Although he wasn't a direct user of the Dipor Dashboard, he gave valuable ideas and feedback regarding to information radiators and appropriate visualizations to monitor software development progress.

Three of the six interviews were contacted online with the conversations recorded using a Skype voice recording extension named Amalto Call Recorder. A brief description about the six interviewees is given as following:

*Table 4-3 Information about Interview Participants*

<b>Serial No.</b>	<b>Profession</b>	<b>Conducted Interview</b>	<b>Remarks</b>
1	Project Facilitator in Demola Tampere	In person	Referred by customer; interviewee has expertise and experience in project management and monitoring
2	Faculty Member of Pervasive Computing Department, Tampere University of Technology (TUT)	In person	Chosen by thesis worker herself; interviewee gave insights on how software development progress can be monitored and visualized
3	Team lead in a startup company, decision maker on financial and development related matters	Online	Referred by Sampo Software Oy; interviewee manages projects procured from Ministry of Education
4	Employee in the Department for General Education and Early Childhood Education, Ministry of Education and Culture, Finland	In person	Referred by customer; expertise in user interface and information architecture
5	System Designer in Helsinki Public Transport (HSL).	Online	Referred by customer; followed only specific projects in Digipalveluhdas
6	Masters student at Tampere University of Applied Science.	Online	Referred by customer; interviewee liked to stay updated on service development under Digipalveluhdas; idea innovator in wellbeing technology for elderly people

The interviews were also conducted in semi-structured way. Similar to the customer interview, written consent was taken for all interviewees for recording the conversations. For remote participants, their consent was collected using Google form.

#### 4.2.2 Interview questionnaire

The questionnaire used in the customer interview was modified to include questions more specific regarding to monitoring service development works. The number of questions was also reduced to keep some time for testing low fidelity prototypes during the interview session. The questionnaire was sent in advanced to all participants before the interview so that they could take mental preparations for the interview sessions. One of the interviewees was uncomfortable with his/her command in spoken English. So he wrote back the answers within the questionnaire.

For the interview session of the faculty member of Tampere University of Technology, questions regarding to community works were not asked. Rather questions about ways for monitoring software or digital service development, helpful information radiators, appropriate visualizations, etc. were asked to get meaningful insights about the overall concept.

As the participant from Demola is not associated with Digipalvelutehdas community, questions addressing the community were skipped. Also Demola projects are about developing a concept solution addressing to solve a problem. It doesn't necessarily need to develop a working solution. So the participant was asked to provide insights about how the Demola project works are monitored in terms of their progress and results.

The following table contains the questions used in the user interview sessions:

*Table 4-4 Interview Questionnaire for User Study Participants*

<b>Association with Digipalvelutehdas Community</b>
1. How are you associated with Digipalvelutehdas community works?
2. What are your designation and responsibilities in your own organization? Do you retain your designation in the community activities? If So, why?
<b>Factors and Technology Used in Service Development Monitoring</b>

3. What factor do you consider while getting updates about the progress of digital service development?
4. Which existing software / platform / application do you use at present to get information about digital service development progress?
5. Is there any specific attribute (e.g. information radiators) of this system (s) you prefer using to know how well development work is progressing? If so, why?
<b>Challenges and Limitations</b>
6. What limitations you face while using the existing system in terms of: 1. Monitoring and 2. Visualizations?
7. Do you face any challenge to retrieve data out of the repository(s) for the digital service being developed in order to make reports, visualizations, etc. about the implementation progress?
<b>Preference in Information Radiator and Appropriate Visualizations</b>
8. Following are some attributes that can be used to monitor service development progress. Do you prefer any of these? Why? Is there any other attribute not listed here?  <b>Man Hours Spent, Commits, Pull Requests, Milestones/Sprints, Issues or Bugs</b> ( <i>reported, closed, fixed, severity, etc.</i> ), <b>development tasks</b> (in different workflow status like <i>not done, planning, in progress, testing, done, etc.</i> ) <b>number of user stories defined vs. completed, allotted time vs. spent time, etc.</b>
9. Following are some visualizations and chart types. Do you prefer any of this? Why? Is there any other attribute not listed here?  <b>Bar charts, Pie charts, Line charts, Sparkline, Burn-down chart, Burn-up chart, Timelines, Kanban Boards, Numbers in large font, etc.</b>
<b>Expectations out of a new service to monitor development progress of on-going digital service implementations</b>
10. For a dashboard to monitor progress of the development work of on-going digital service implementations, what features do you expect to see to make your work easier?

All interview sessions were kept within a length of an hour. Except for the Demola adviser and the Pervasive Computing Faculty member at TUT, participants were asked to evaluate 2 sets of low fidelity prototypes created for the intended system. The prototype

testing was done after the interview was conducted. The description about the prototype evaluation process is described in the following sub section.

### 4.2.3 Paper Prototype Evaluation of Dipor Dashboard

All interviewees except for first and second participants (Demola project facilitator and faculty member of Pervasive Computing at TUT) were asked to evaluate paper prototypes of two versions of the dashboard view in Dipor Dashboard portal. This was done because the first two interviews were needed to get insights about monitoring software or digital service development process. Also at that point of time, the initial personas and affinity diagrams were still under development to gain a basic understanding of the concept. So no prototype was ready when those interviews were conducted.

Low fidelity prototypes were used to understand how users would customize the dashboard to accommodate their work needs and goals. Usually a low fidelity prototype contains a few screenshots of the intended system (not necessarily the final versions). Participants evaluating them can give their feedback on the system concept (look-and-feel and functionality) based on the ideas and output presented in front of them. This helps to identify crucial factors in the very early phase of both concept and actual development (VAN VELSEN L, VAN DER GEEST T, KLAASSEN R, 2008). In addition, users' interactions with the low fidelity designs give provisions to bring improvements in presented ideas for further tests (Arhippainen & Tähti, 2003). Observations made during the prototype testing can provide glimpse about possible user behavior and reactions to the actual implemented solution.

Two versions for paper prototypes were produced for the intended evaluation. One version contained the design based on the system requirement prepared by Sampo Software Oy. The second version was designed from visualization and measurement notes in the related affinity diagrams. Both prototypes presented the intended Dipor Dashboard as a website suitable for desktop or laptop view. The prototypes were created using Balsamiq mockup software. In addition to the website design, there were sample visualizations (e.g. bar charts, line charts, sparkline, numeric figures, etc.) made with sticky notes to aid the participants choosing their favorite ones to visualize different data.

Although not interactive or representing functional webpages, both versions of the prototypes were evaluated in the manner of A/B testing. This was done to understand which variation of the dashboard view was working better with the participants' interactions around to prototypes ("What is A/B Testing?," n.d.). During the tests, specific suggestions about improving current features in both versions were received from the participants. Version A was the UI design provided by Sampo Software Oy. Version B was the dashboard concept designed by the thesis worker. The four participants with

whom the prototype evaluation occurred were randomly given version A or B at first turn. The remote interviewees provided their feedback by using Google form, where the wireframes were added with associated queries. Improvement suggestions received in for both versions in a particular interview session were considered and necessary changes were made to both versions before presenting them in the consecutive session.

Initially the whole web portal system was considered to be evaluated. However, to make the thesis work more focused, it was decided by the thesis worker to mainly consider the dashboard view of the portal. This was carried through the heuristic evaluation and usability testings. It was also considered to accommodate necessary improvements in version B as suggested by a particular participant and run a second evaluation by contacting him/her. However, the development work was reaching towards completion of its 3 months period and the interviewees were not available due their work schedules. So this plan had to be abandoned.

The final version of the prototype wireframe for the service designed by Sampo Software Oy is added in Appendix B. The appendix also contains suggested feedback and iterations in designs made in both versions. The dashboard concept developed for this thesis work is described in the Results subsection.

### **4.3 Results from the Executed Methodologies**

The following section describes the affinity diagrams, personas, user requirements that resulted from the conducted user interviews. Constructed affinity diagrams helped to finalize the persona designs and aided in generating user requirements for Dipor Dashboard. In addition, a concept of the dashboard feature using low fidelity sketches has also been presented.

#### **4.3.1 Construction of Affinity Diagram**

Creation of affinity diagrams from interview notes was initiated from the customer interview. The purpose for constructing the diagrams was to aid the creation of personas for Dipor Dashboard and generating requirement set for the intended system. Affinity diagrams were suitable for this purpose as they help to map insights from user information into hierarchical diagrams (Beyer & Holtzblatt, 1999). It becomes easier to identify problem scope as common patterns and structures become visible in affinity diagrams without losing the variation of works different people do. While validating design ideas, affinity diagrams help to see how the design functions as whole rather than different broken down segments.

Affinity diagrams for this thesis works were being updated frequently as more user interviews took place. Notes were collected from the written interview scripts that were prepared from audio recordings of the interview conversations. The scripts contained exact quotes of the interviewees with minor changes to make them more comprehensive. The size of user quotes for affinity notes depended on the relevancy of information included within the user speech. Participants' name initials were used to identify affinity notes. For compiling the affinity diagrams, constructed quotes were categorized into the following types and keywords:

**Roles <RO>:** Designations held by interviewees within their respective organizations and within the Digipalvelutehdas community.

**Responsibilities <RES>:** Activities and duties defined for the roles held by people in their respective work organization as well as in their contribution in different project Works.

**Current Work patterns of users <CW>:** In present times, how people monitor service development works. These include work patterns and structures maintained by respective organizations as well as the community.

**Current Situation <CS>:** In present time, how service development is being managed within both organization structure and community boundary. This primarily focused on actions taken to know whereabouts on development progress, how progress is being measured on, actions taken to communicate development status, etc.

**Limitations <L>:** Challenges and obstacles people face frequently in conducting their works regarding to service development monitoring in order to accomplish their goals.

**Expectation <EXPEC>:** People's needs and wishes for changes in work procedure, availability of new technology, etc. in order to address current work limitations. This included means to ease up monitoring works.

**Expertise <EXPER>:** People's knowledge about their field of work, educational background, technical skills, etc.

**Technology <T>:** Available market products, platforms and technology people use in present time to observe service development monitoring and progress status.

**Visualizations <VIZ>:** People's preferred way of projecting service development status by using different means of visual and informative elements.

**Measurement <ME>:** Information radiators, metrics, indicators, etc. people follow in order to understand and determine the progress status of software development process.

The created affinity diagrams were used to develop personas of the intended system. They were also used to construct primary user requirements. Often, they were consulted while making changes to low-fidelity prototypes. The primary affinity diagrams were merged to create secondary levels to show summarized view of the obtained information regarding to people, challenges and expectations to overcome them, people's understanding about monitoring progress work and preferred way of information visualization. Affinity notes have been highlighted wherever information is found suitable for developing user requirements.

The constructions of affinity diagrams were finished at the end of phase 1 of the empirical work. They are available in Appendix A of this document.

### 4.3.2 Personas

An important designing aid to convey the vision and design of a new system is to create "archetypal users" or personas in short (Calabria, 2004). This designing method helps to make decisions about functionality and design by considering needs of larger group of users in terms of their goals and individual characteristics. Creating persona also helps to distinguish between necessary and unnecessary needs and provides assistance in determining what features can be frequently used by intended users and what not (Cooper, 2004). Classification of users can be made by asking them open ended questions about work goals they need to achieve using a new service. This can be aided by information regarding to users' workflow throughout the day, surrounding environment, tasks, skill levels, etc.


From the customer interview it was understood that the members of the Digipalvelutehdas community would be the primary users of Dipor Dashboard. People in this community have different educational backgrounds (e.g. Social science, economics, Information Technology, etc.), technical skill proficiencies and responsibilities to organizations (public sector or private) they are associated with. From the notes of the customer interview, development of the personas started. With more affinity notes obtained with each interviews in this phase, the initial personas, along with the concept design, were updated and modified in parallel (Chang, Lim, & Stolterman, 2008). The work continued in parallel with the actual implementation of Dipor Dashboard by Sampo Software Oy.

It was considered beneficial to keep the number of personas into four to describe the most important goals and significant behavioral patterns. Due to unavailability of potential interviewees, generated affinity notes were examined more than once to determine what might be the activity patterns of potential personas to reflect their goals, skill sets, interaction with systems, etc. For keeping the focus on the design, three to four important goals were considered and priorities were given more toward Experience goals




than Life goals (Goodwin, 2001). It was considered to keep persona details to minimum and to focus on daily activities, behaviors, end-goals, etc. Based on the gathered information, two primary personas and two secondary personas were created (Ogle, n.d.)

### Primary Persona 1

	<p>Name: <b>Aleksanteri Jokinen</b>                  Expertise: IT specialist                  Likes: Ice Hockey and Cat</p>
<p><b>Role / Designation</b></p>	<ul style="list-style-type: none"> <li>• Development Manager, Department for General Education and Early Childhood Education, Ministry of Education and Culture, Finland</li> <li>• Product Owner (PO) of different digital service projects sanctioned by the department</li> </ul>
<p><b>Activities</b></p>	<ul style="list-style-type: none"> <li>• Communicates with project managers from different companies to know whereabouts of ongoing software projects under the department.</li> <li>• Provides rough requirement sets for a new service to develop and if needed revise them.</li> <li>• Browses version control systems to checkout different development activities, issues reported, contribution made by different team members, etc.</li> <li>• Tests intermediate released versions of a product and reports about his findings as improvement suggestions or bugs.</li> <li>• Have weekly meetings with department colleagues to change and evaluate status of different services.</li> <li>• Checks out monthly status report sent by project managers from different companies to aid decision making process of the department.</li> </ul>
<p><b>Goals</b></p>	<ul style="list-style-type: none"> <li>• To obtain real time information about development progress of ongoing services.</li> <li>• Aiding department head to make decisions about creating new services and or continuation of existing ones based on their usefulness and popularity.</li> <li>• Retaining transparency of department work to mass citizens.</li> </ul>


<b>Challenges</b>	<ul style="list-style-type: none"> <li>• Monthly reports instead of instant real time ones on service development progress.</li> <li>• Have to browse individual repositories or version management systems separately for each service to know development status.</li> <li>• Finds existing solutions not suitable enough to configure organization work process and support open source development.</li> <li>• Finds existing visualizations of information insufficient to predict status of the development work.</li> </ul>
<b>Preference</b>	<ul style="list-style-type: none"> <li>• Having at-a-glance information on all services he monitors or follows.</li> <li>• Brief as well as detailed overview of development status using appropriate visualizations and explanations.</li> <li>• Using features that are easily configurable and doesn't need expert skills of people from technology background.</li> </ul>
<b>Scenario</b>	<p>On a monthly department meeting, Aleksanteeri needs to explain to the board which service developments under the department are not progressing as expected. He has thirty minutes before the meeting. He needs to make his reasons by visiting different GitHub repositories and get information about open issues, number of reported bugs, merged pull request, etc.</p>

**Primary Persona 2**

	<p>Name: <b>Pekka Peura</b>                  Expertise: Studied in Computer Science, has knowledge in Software Architecture                  Likes: Video game playing</p>
<b>Roles/Designation</b>	<ul style="list-style-type: none"> <li>• Senior System Architecture, Helsinki Region Transport (HSL)</li> <li>• Company representative of a project under Digipalvelutehdas community</li> </ul>
<b>Activities</b>	<ul style="list-style-type: none"> <li>• Negotiates contracts and agreements with product owners from different public sector organization.</li> <li>• Maintains communication with product owner of own project under Digipalvelutehdas community and reports about project development status to him.</li> <li>• Maintains code repositories and management tools for own projects.</li> <li>• Follows some projects of interest in Digipalvelutehdas community.</li> </ul>


<b>Goals</b>	<ul style="list-style-type: none"> <li>To represent company policy and ongoing projects' milestones to clients and mass people.</li> <li>To successfully finish assigned services within deadline and allocated resources.</li> </ul>
<b>Challenges</b>	<ul style="list-style-type: none"> <li>Finds it difficult to work with complicated version management systems (e.g. Jira) and non-open source ones.</li> <li>Finds popular analytics platform (e.g. Google Analytics, PIWIK, etc.) evasive to user information.</li> <li>Finds GitHub not being fully capable to follow preferred Agile development methodologies (e.g. Scrum, Kanban, etc.)</li> <li>Doesn't get enough times to go through details from individual projects he is following or managing.</li> </ul>
<b>Preference</b>	<ul style="list-style-type: none"> <li>Knowing development team's contribution in a project.</li> <li>To get notifications about significant changes made to projects he follows.</li> <li>To get detailed information about favorite projects.</li> <li>To see correlated indicators of project progress together to get clearer picture.</li> </ul>
<b>scenario</b>	<ul style="list-style-type: none"> <li>Samu instructs his teams to use customized issue labels in GitHub. From the repository, he filters the issues with specific labels to see how many issues are open under a specific label. Based on the available resource and number of issues, he tries to determine if work is being piled up and not completed in time.</li> </ul>

### Secondary Persona 1

	<p>Name: <b>Saara Raukko</b>                  Expertise: Economics and Sociology                  Likes: Indoor climbing, knitting</p>
<b>Roles</b>	Department Head, Department for General Education and Early Childhood Education, Ministry of Education and Culture, Finland
<b>Activities</b>	<ul style="list-style-type: none"> <li>Defines organization's agenda and work process.</li> <li>Makes decisions about implementing new services, continuing existing ones, budget allocation for the services, etc.</li> <li>Conducting meetings with subordinates to know what services the department has undertaken, their statuses, involvement of department's employees in those services, etc.</li> </ul>
<b>Goals</b>	<ul style="list-style-type: none"> <li>To represent department's work and contribution in public sector digitization to the Ministry and mass citizens.</li> </ul>

<b>Challenges</b>	<ul style="list-style-type: none"> <li>• Not having enough time to go through details about different projects and their information.</li> </ul>
<b>Preferences</b>	<ul style="list-style-type: none"> <li>• At a glance information indicating status of a Development, preferably in numeric values.</li> <li>• Some simple and clear indication if a service well or needs attention.</li> <li>• Brief overview about number of services and their status including all phases of service lifecycle (from ideas to retiring phase).</li> </ul>
<b>Scenario</b>	<ul style="list-style-type: none"> <li>• On the monthly meeting, Saara needs to make some decisions about budget cuts for present services undertaken by the department. She needs to focus services that are under development phase. As she doesn't have time to investigate the repositories, she just wishes to see some indication about which services are performing poorly.</li> </ul>

### Secondary Persona 2

	<p>Name: <b>Samu Rautio</b>                  Expertise: Well-being technology                  Likes: Building things from scratch</p>
<b>Roles/Designation</b>	<ul style="list-style-type: none"> <li>• Student, doing Bachelor thesis from TAMK</li> <li>• Idea Innovator</li> </ul>
<b>Activities</b>	<ul style="list-style-type: none"> <li>• Maintains own website regarding to innovations and ideas on well-being technology for elderly people.</li> <li>• Follows public sector digital service projects.</li> </ul>
<b>Goals</b>	<ul style="list-style-type: none"> <li>• Get approval and funding to lead and manage his own public sector digital service to develop a communal model to aid senior citizens to set goals, select interests and evaluate against others their well-beings.</li> </ul>
<b>Challenges</b>	<ul style="list-style-type: none"> <li>• Finds it difficult to understand data he uses if no visualization is available for projecting it.</li> <li>• Finds it difficult to discover available actions if they are hidden behind menus.</li> <li>•</li> </ul>
<b>Preference</b>	<ul style="list-style-type: none"> <li>• Means of customizing information he would like to see with appropriate visualization options.</li> </ul>
<b>Scenario</b>	<ul style="list-style-type: none"> <li>• Samu is following GitHub repository for a service development project in GitHub. On weekly basis he usually checks number of issues open or closed and tries to guess the trend. He wishes he could see his preferred information in the same view using both numeric values and charts projecting trends.</li> <li>•</li> </ul>

### 4.3.3 User Requirements

User requirements aid to develop user centered product design (Kaulio, 1998). They help to generate measurable engineering requirements to develop a new technical solution by analyzing problems in use situations. User Requirements help to organize information from affinity diagrams so that it can be determined if developed solution meets user's necessities and expectations.

The following user requirements for the Dipor Dashboard have been developed primarily from the information obtained from the created affinity diagrams. The emphasis is made on the limitations that the users face everyday life to determine how project work is going on and how they would wish for the new system to help them solving their dilemma according to their preference and work pattern.

**User Requirement 1:** User's home screen should be comprised of a dashboard that would show overview status of services s/he is managing or following in Dipor Dashboard portal. It should be possible to configure phase of time for which the data in the dashboard would appear.

**User Requirement 2:** Users should be able to add existing services in Dipor dashboard which they have access into using quick actions. It should be possible to add the service itself or associated GitHub repository(s) for the intended service.

**User Requirement 3:** Services appearing on the dashboard should show general metadata: E.g. GitHub Repository name and link, service completion date, associated organization's name and the life cycle phase under which the service is currently in.

**User Requirement 4:** Upon adding a repository, information fetched regarding different attributes should appear with comprehensive texts and/or visualizations. If there is any change in the information for any attribute from the past instance of the selected time period, it should be indicated with respect to the attribute in question. Visualization for each attribute would indicate the trend of its change for the chosen instance of time period.

**User Requirement 5:** Users should be able to customize the repository attributes that would appear for individual services. There should be a configuration menu for each service which they can use to customize the attributes and their appearance (text only or along with visualizations).

**User Requirement 6:** The dashboard would notify user when a service needs user's attention because of issues in its development progress. Visual cues and indicators should be available to indicate which service and what attribute(s) related to service needs such attention.

**User Requirement 7:** Each service should have a detail view so that users could study past historical data in bigger visualizations. Information from GitHub repositories should be visualized so that it is possible to identify potential trends and evaluate the progress of the development work. This would be helpful specifically if services needing attention need more investigation.

**User Requirement 8:** Some appropriate visualization projecting implementation works using Agile development methodologies should be present in the details view, as most services being implemented follow such methodologies.

**User Requirement 9:** Since the services have a three-month long implementation cycle, there should be means to see overview information on development progress for every 30-day long period.

#### 4.3.4 Sketches of the Concept Developed for Dipor Dashboard


This subsection describes the finalized version of the design sketches made for developing the concept of Dipor Dashboard. The result presented here focuses on the empirical work conducted at the phase one of the thesis. The final sketches are made using Adobe Photoshop and represents Dipor Dashboard website suitable for viewing in desktop or laptop screen. As mentioned earlier, the Dashboard part of the system had been the primary focus while developing the concept.

Unlike the design prepared by Sampo Software Oy, the web portal in the concept would have three primary structures: **Dashboard**, **Organizations** and **Services**. Dashboard would be the first view where user is navigated upon login into the system. Organizations and Services would be catalogues featuring added organizations and services added to the system respectively.

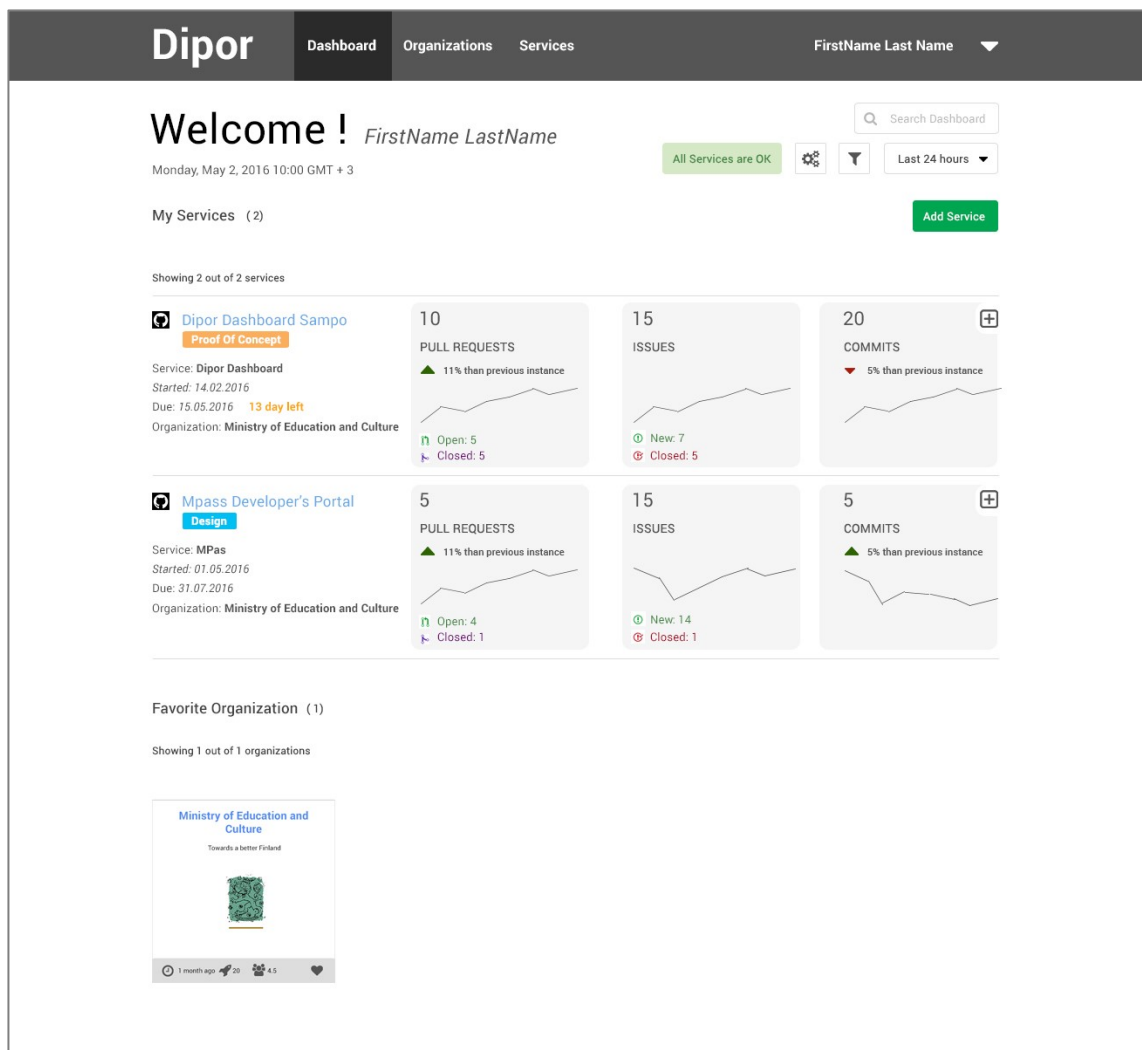
##### Dashboard

The dashboard view would primarily provide the user (who has logged into the system) about the statuses of service development s/he oversees or follows being part of associated organization(s). The dashboard would have two parts. The upper part would show

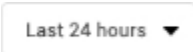
statistics related to the services added or managed by the user. A welcome message would greet the user with his/her full names. A text showing current time and date would appear to help the user making comparison between service related data associated with past time stamps.

The lower part would show Organizations that the user has kept in his/her favourite list. Organizations would appear as cards containing information like its creation date, number of services under the organization and its members/followers. User would be able to put an organization in his/her list of favourite ones by clicking on the  icon.

*Figure 4-1 Concept of the dashboard in Dipor Dashboard*




. The dashboard view for monitoring service status would begin with a few controllers. User would have the option to filter the presented information from a dropdown menu


. The time could be configured as **Last 24 hours**, **Last 7 days** and **Last 30 days**. A searching option would be present to aid users looking for specific service

or organization. To configure and filter the dashboard, specific action buttons would be available. Their description is provided in subsequent paragraphs.

Services owned or followed by the user would appear as rows in the Dashboard view. Each row would project information obtained from the GitHub repository used for developing that service. The reason to focus on the code source rather the service itself is to accommodate two GitHub repositories (if any) dedicated to develop the same service. It also provides some means to make comparison between activeness of two repositories.

Each row would contain some metadata about the service and visualizations customized by the users. The metadata would include **GitHub repository name**, **link** to the actual repository page (using  icon), name of the **service**, associated **organization** and the **start** and **Due dates** of the service. Notification message would appear when the completion date for a service approaches nearby. In addition, the associated lifecycle phase (as defined by Digipalvelutehdas community, more on this in Chapter 5) would also appear with the service. An existing service in the system and associated GitHub repository could be added to the dashboard with “Add Service” button.

### Visualizations and Widgets

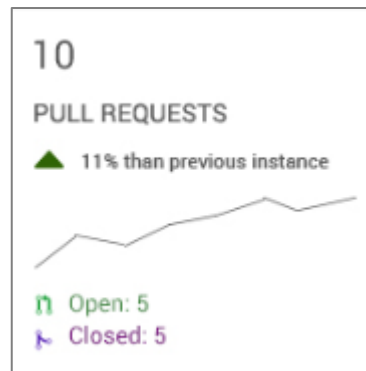
To customize the information as per users’ preference for each repository, there would be a list of available GitHub attributes that could be added to the dashboard. The list would be accessible for each row through  button. The idea of allowing users to choose information and associated visualizations was adopted from “widgets” which are often used to customize screen of smart phones (Developer.android.com, 2014). In smartphones, users can view the most important and functionality “at a glance” and quickly access them from the home screen. They help to monitor information that is crucial for the users and are compact in size. So, the concept of using widgets like visualizations seems appropriate for dashboard designs. For Dipor dashboard, the following attributes that can be retrieved from a GitHub repository have been considered: **Pull Requests, Issues, Commits, Milestones, Contributors, Issue Labels** and **Pulse**.



**Figure 4-2** Widget controller in Dipor Dashboard

To keep the dashboard view simple and not cluttered with too much information, users would be allowed to select up to three attributes to display against an added repository. Checkboxes to select additional attribute(s) would become disabled once three of them have been checked. For the attributes Issues per Milestone (“About milestones - User Documentation,” n.d.-a) and Issues per labels (“About milestones - User Documentation,” n.d.-b), the same customization approach has been adopted (e.g. controller would become disabled if user selects all three of them). Users are given the choice to show the attribute information in numbers only or with visualization as well. Upon selection, the associated attributes for a repository would appear in the related row. To maintain familiarity (as suggested in Nielsen’s 10 usability heuristics (Nielsen, 1995a)), icons and terminologies from GitHub would appear as per the attributes. Each attribute appearing on a row would have the following information: Number summing up the total amount for the selected period, changes (if any) in percentage from the last instance of the time, amount in sub categories (if any) for the attribute and a sparkline (if both number & chart is selected) to show the trend of the attribute for the filtered time. The reason for choosing a sparkline is because its ability to track and compare changes in information by time and present the overall trend (Tufte, 2006). Also it helps to keep the UI clean. The figure below shows a widget containing pull requests (PR) made in the last 24 hours in that specific repository with number of open and closed PRs.

**Figure 4-3** A widget showing pull requests for a GitHub repository



### Filtering and Configuring the Dashboard

In this concept of Dipor Dashboard, emphasis had not been given on designing the filtering option. However, possible filtering category could be based on life cycle phases of the services, services needing attention, etc. For the later criteria, it would be possible for a user to configure the dashboard to show alerts for repository(s) that needs attention. It would be possible to set threshold value to numerical attributes of GitHub (e.g. pull requests, commits, issues, etc.). A user would be able to customize multiple threshold values for an added repository. This monitoring of threshold value would work only when that GitHub attribute is added in the dashboard view. In normal situation, a message in the dashboard would indicate that all services are working well. Upon exceeding the set threshold value for any of the attributes, alert would appear beside the service. Also the attribute in questioned would be highlighted.

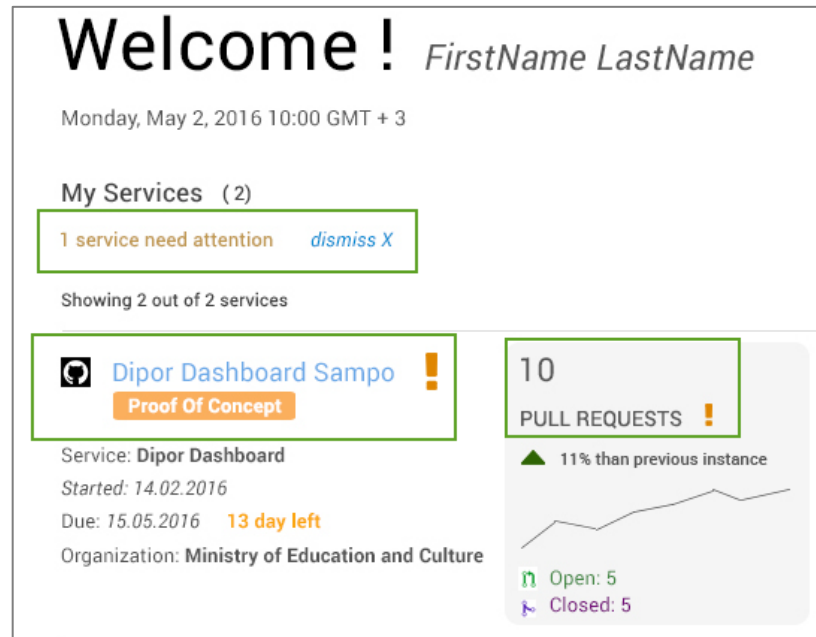
Some possible logics for setting threshold are given in the following table

**Figure 4-4** Possible logics to show alert in dashboard view

For Attribute	Condition	Time period	Action
Pull Request	If <b>Open PR &gt; 5</b>	in <b>LAST 24 HOURS</b>	Show Alert
Issues	If <b>Close Issue &lt; 10</b>	in <b>LAST 7 Days</b>	Show Alert
Commits	If <b>total commit &lt; 15</b>	<b>In LAST 30 Days</b>	Show Alert

It is advisable to present the dashboard with some default visualizations and threshold settings suitable for general audience. However, determining these default attributes and appropriate threshold values were out of the work scope and is left if further modification in the concept is done in future.

**Figure 4-5** Dashboard showing alert for a repository for exceeding threshold value for closed pull requests



By providing means to customize the dashboard in terms of visualizations and monitoring aspect, it would be possible to address needs of different personas for achieving their work goals regarding to managing and monitoring digital service implementations. For an example, an IT Specialist could use his/her knowledge and experience in setting threshold values for GitHub attributes and understand the why a service needs attention (if any) by looking at the trend changes in that attribute. He can suggest the adopted threshold configurations to the Department Head. Before setting up a meeting with the IT Specialist, the Department Head could look at the dashboard and get at glance information on which service(s) need attention for slow development progress. This would help her to take decisions on further continuation of the services based on the insights provided by the IT specialist.

### Details View

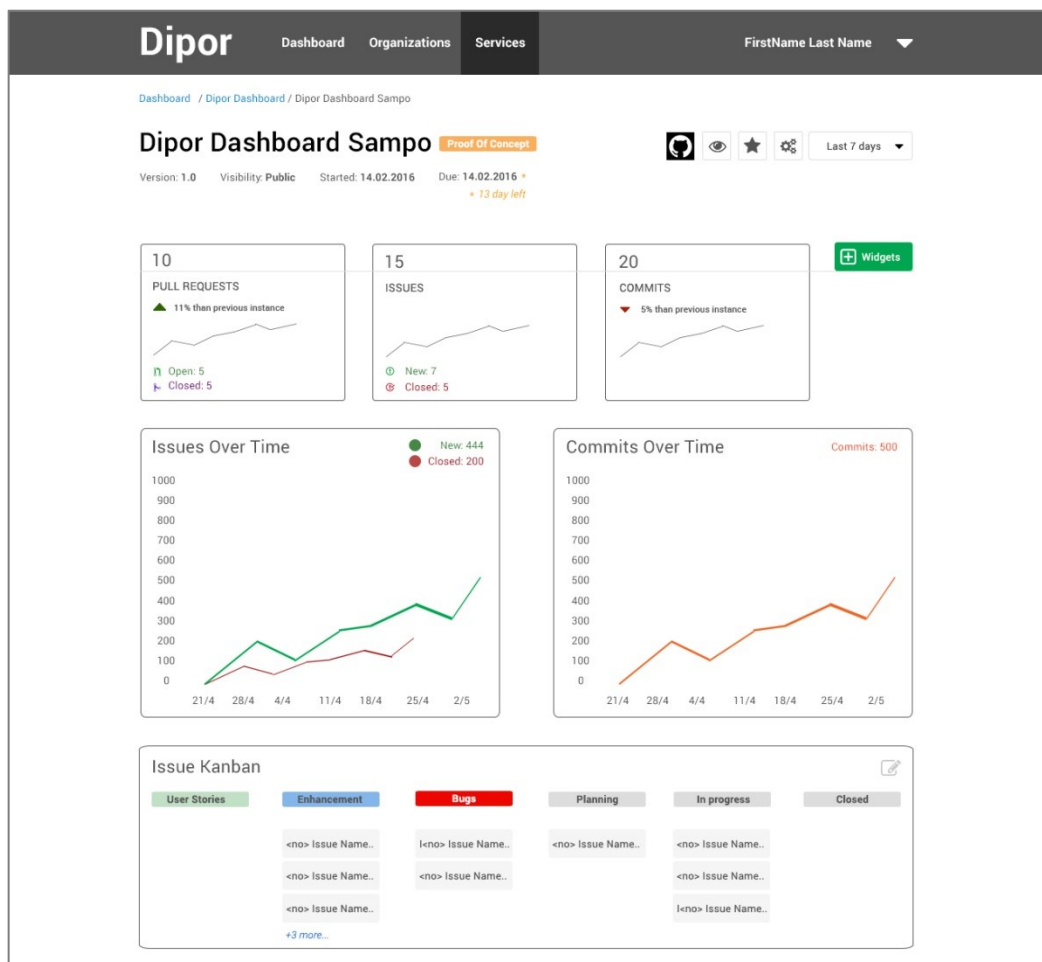
One of the features of this dashboard concept that differs from the design done by Sampo Software Oy is a separate details view dedicated for each GitHub repositories added the to the system. A details view would be beneficial in cases where the intended user wishes to go deeper into repository data for services, specifically the ones that need attention. Big amount of repository data presented with appropriate visualizations to indicate correlations, hidden patterns and other insights would make it easier to draw conclusions about how well the development work has progressed and if it worth the time, effort and money on further continuation of the service.

By clicking a GitHub repository name, user would be navigated to the details view for that repository in Dipor Dashboard. This navigation would also be possible if user tries

to access the view from the service profile itself. Details view of a repository contains the following metadata: version number of the service for which the repository is dedicated to, life cycle phase of the service, start and due date of the development work and visibility (public or private) of this repository work to general audience. Information and visualizations presented here can be filtered using the similar time periods as that in the dashboard view. In addition, familiar GitHub actions (e.g. watching the repository for updates, giving it a star rating, etc.) can be done using related buttons. Also navigation to the repository in GitHub would be possible by clicking GitHub logo.

The same set of widgets customized in the dashboard view would be repeated in the details view for minimizing memory loads for the user (Budiu, 2014). Widgets available in the dashboard view would also be available in the details view. An advantage of having the details view would be to add more than three (even all) widgets because the view contains adequate space. The options to for adding multiple Milestone and Labels have been kept to a maximum limit of three as same as that in dashboard view. This would ensure that the details view is not cluttered with too many visualizations showing similar information.

**Figure 4-6** Details view of a repository in Dipor Dashboard



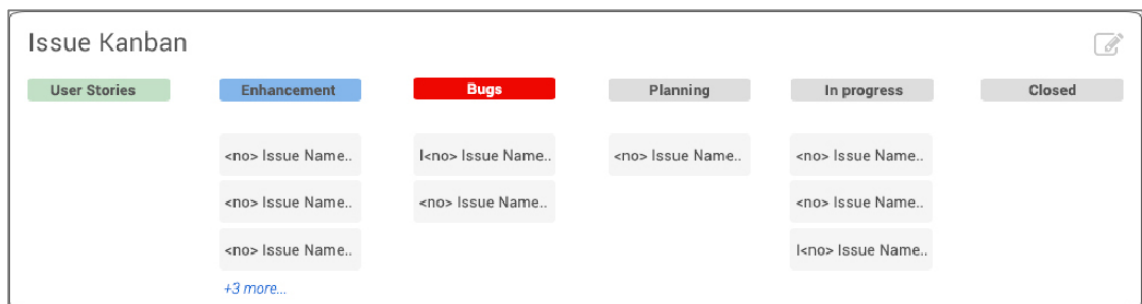
Apart from the overview information visualized in the widgets, two large line charts have been added to show changes in issue and commit history. The **Issues over time** chart would show trends in open and closed issues for the selected time period. This chart could be filtered (by using the red and green circles on the chart) to show only open or close or both issue types. To make the charts comprehensible, numeric values for open and closed issues would also be projected in the chart. For this concept, the Commit over time chart would show the trend of all commits made by all contributors in this repository. A summary value would also appear in the chart. It would have been possible to show individual committer's commit trend in the graph. However, the current visualization would not be appropriate to show individual committer's information if a repository contained a good number of them. One alternative could have been to show five most active committers' trends with the current visualization. However, the idea has been kept out of the focus of this concept.

Lines charts have been considered for these visualizations because of their ability of clear and instant showcase of the data shape (Few, 2006). If column charts were used in such visualizations, the UI would look heavy and cluttered in case user was filtering data for past 30 days.


### Visualization of Agile Development

From competitor analysis and studying existing works, it was found out GitHub repositories don't support estimating work efforts or hours needed to complete an issue. Also there is no default development workflow in GitHub. So the logic for implementing a burn-down chart to appear in the details view might be complicated. Instead, an easier way to project agile development progress is to use and visualize Kanban workflow.

*Figure 4-7 Issue Kanban for a GitHub repository in Dipor Dashboard*

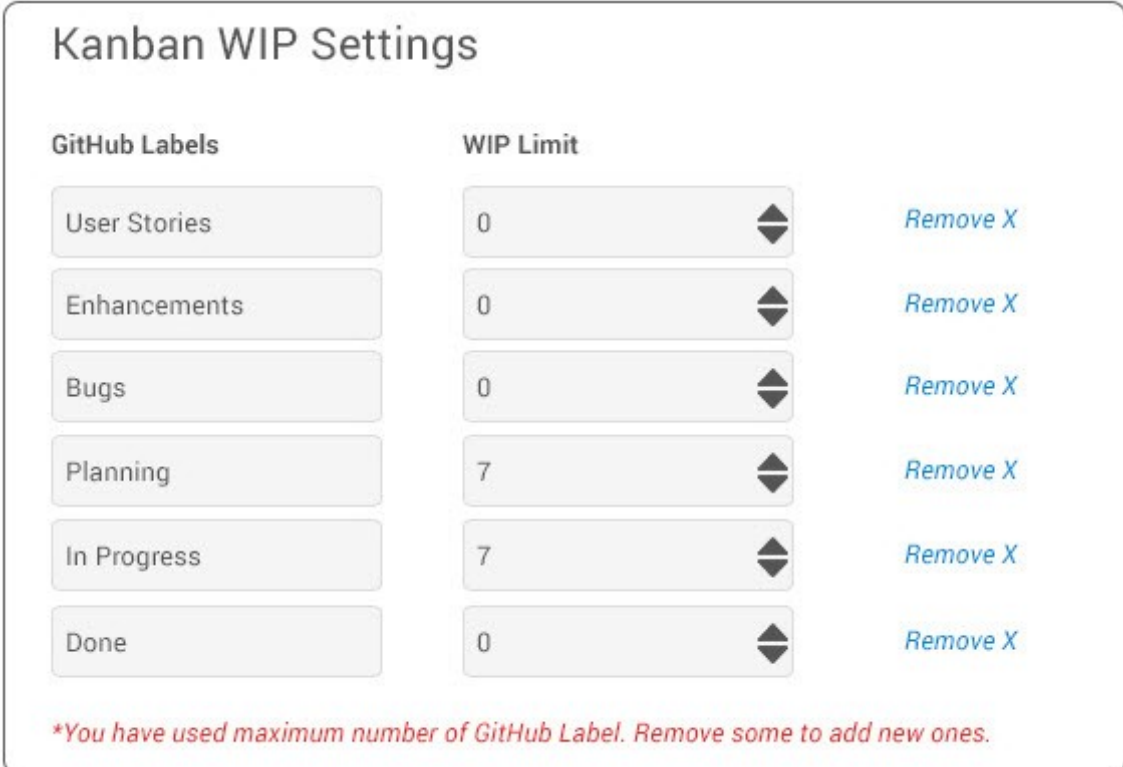


The idea of showing Kanban board has been partially adopted from a prototype Kanban tool for task assignment (Nakazawa & Tanaka, 2015). For Dipor Dashboard portal, a user would be able to customize the Issue Kanban visualization to show number of issues under a specific label and set a Work In Progress (WIP) limit for each label. This would be possible for both GitHub default labels and user created ones. In GitHub repository, it is possible to create a pseudo workflow by carefully creating specific labels and associating issues fulfilling some specific criteria (e.g. within a milestone and assigned to a contributor) to related labels (Dewalt, 2016). A user can customize WIP

limits up to 6 labels by accessing the Kanban settings controller with  button. In the controller, users can add numerical values indicating WIP limits for the selection of labels.

In figure 4-8, it is shown that customized labels **Planning** and **In Progress** has a WIP limit of 7 each. This would indicate that if there are more than 7 issues in the original repository under Planning or In progress label, there is a presence of bottle neck in works. As customized labels could be used to represent phases in a pseudo Agile Kanban work process, having bottlenecks in phases like In Progress means issues are being piled up in a specific phrase and there hasn't been any progress for implementing them. Even without a workflow, it is possible to show bottlenecks under default GitHub labels. It is also beneficial for a user to customize WIP as it may vary from person to person. Also it assures that the issues in the original repository won't be manipulated with the action.

*Figure 4-8 Controller for setting WIP limit to issue labels in Dipor Dashboard.*



GitHub Labels	WIP Limit	
User Stories	0	<a href="#">Remove X</a>
Enhancements	0	<a href="#">Remove X</a>
Bugs	0	<a href="#">Remove X</a>
Planning	7	<a href="#">Remove X</a>
In Progress	7	<a href="#">Remove X</a>
Done	0	<a href="#">Remove X</a>

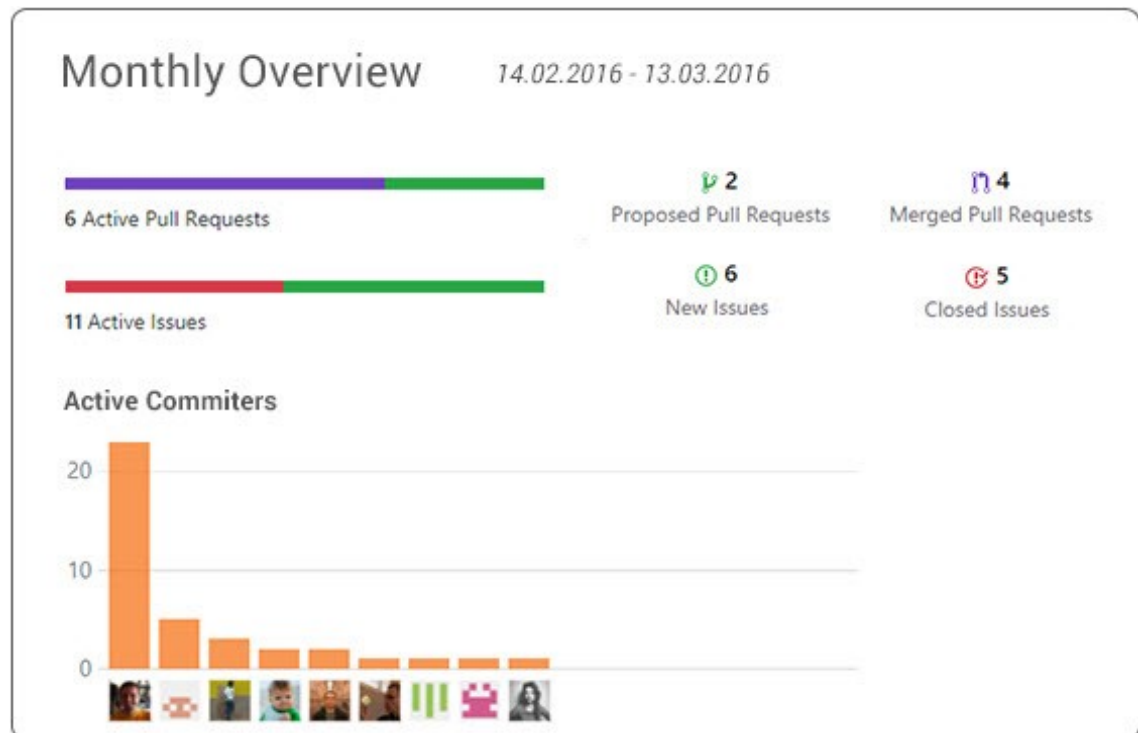
*\*You have used maximum number of GitHub Label. Remove some to add new ones.*

Issues under specific labels will appear as cards in the Kanban board with their original name and number in related GitHub repository. User can be navigated to the issue page in GitHub by clicking the name on the issue card.

## Overview of Monthly Statistics

As services under Dipor Dashboard would have a three-month long development period, it would be beneficial if service owners could have an overview of the monthly work progress. So a new widget specific for the details view can be used to serve this purpose. The idea of this widget has been influenced from the Pulse visualization in a GitHub Repository (“Viewing a summary of repository activity - User Documentation,” n.d.). The widget called Monthly Overview would show information on active PRs, active issues and active committers using familiar GitHub symbols and visualizations. The widget could be configured with monthly timeline starting from the date the service or repository was created. It would be possible to add three widgets for a service in the details view. The only limitation of this widget is that it would not be possible to show this widget in the details view before the first 30 days of the service development work has taken place.

*Figure 4-9 Monthly overview of a service development in Dipor Dashboard*



## 5. AN OVERVIEW OF THE IMPLEMENTED DIPOR DASHBOARD

This chapter provides a brief description about Dipor Dashboard that was implemented by Sampo Software Oy by following the system requirements constructed and negotiated between the company and the customer. The actual web service<sup>1</sup> can be accessed from the link provided as footnotes. The link for the associated GitHub repository<sup>2</sup> for this web portal is also given there. The description of the view has been generated by logging into the system using site administrator's credentials.

### 5.1 Home Page of Dipor Dashboard

On the home page of Dipor Dashboard, a set of organizations appears that are featured in the web service. Organizations are entities that can be used to represent different real life public (or private, if applicable). Under these organizations, works for implementing various digital services are initiated, financed and owned. List of organizations appear as cards. Each organization card contains Name, a small description about organization's work, a logo and link to see more information about it. Without login into the system, the general audience can see information that has public visibility set by the site administrator (e.g. Organization's Names, departments under organization and services being built under a department).

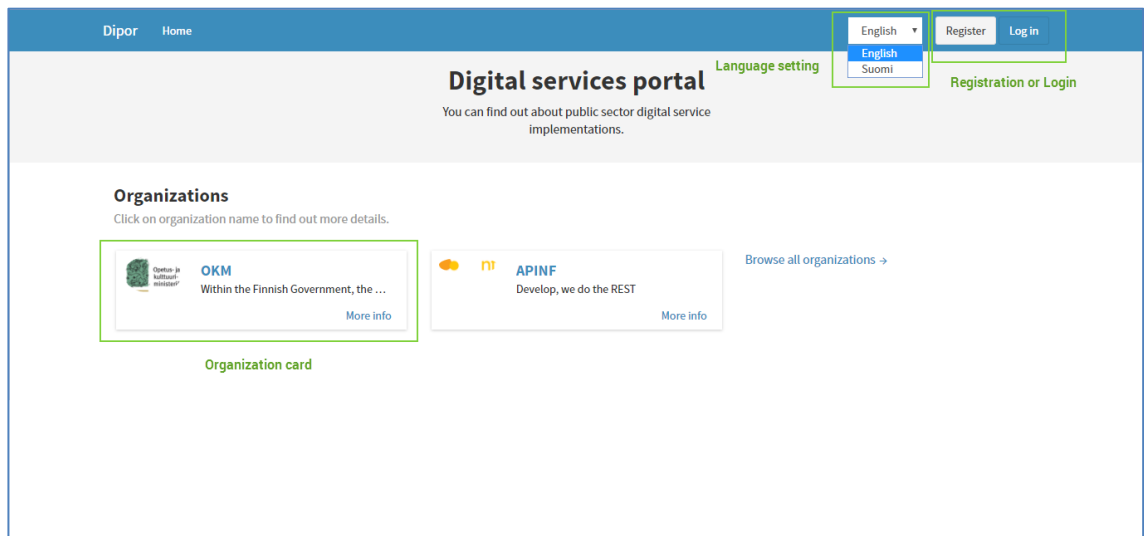
People can see the organization catalogue that contains all organizations in this web service by clicking "Browse all organization" link. There are provisions to change languages (English or Suomi) and to register or login to the system at navigation bar. People need to use a valid email address to register in the portal and later they can access the system when site admin approves the registration.

---

<sup>1</sup> <https://dashboard.digipalvelutehdas.fi/>

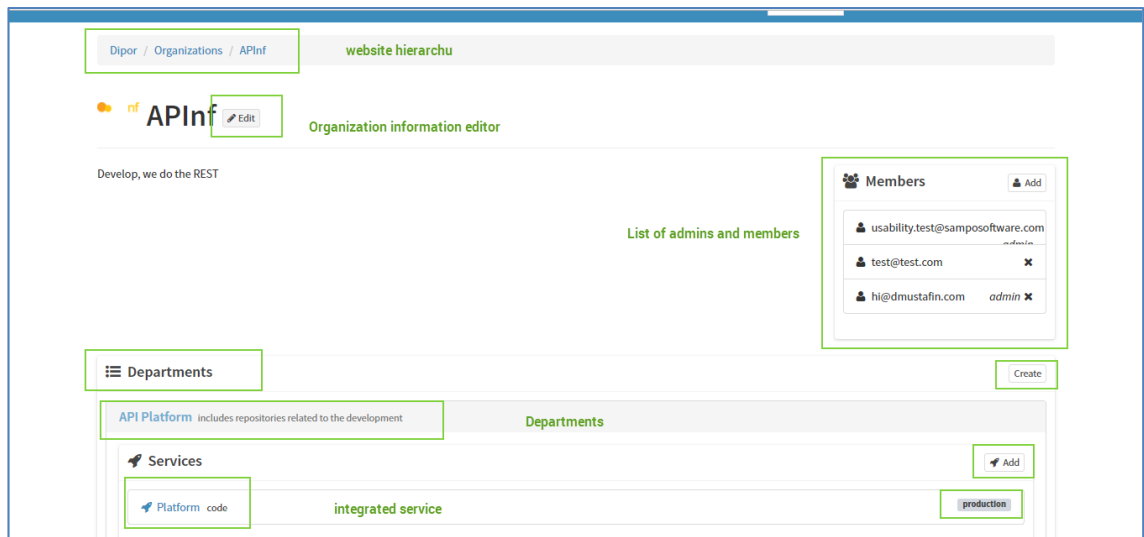
<sup>2</sup> <https://github.com/Digipalvelutehdas/dipor-dashboard>



*Figure 5-1 Home page of Dipor Dashboard*

## 5.2 Organization view

After login in, user is redirected to the Organization view of Dipor Dashboard. If logged in as a site admin, user gets the privilege to create new organizations. Clicking on the name link of an organization, s/he gets navigated to the profile page of the organisation.

*Figure 5-2 Organization profile in Dipor Dashboard*

Inside an organization profile, there is provision to add description about the purpose and work of this organization. A separate layout shows the members of this organization. Some members can get admin privilege and control different actions related to that

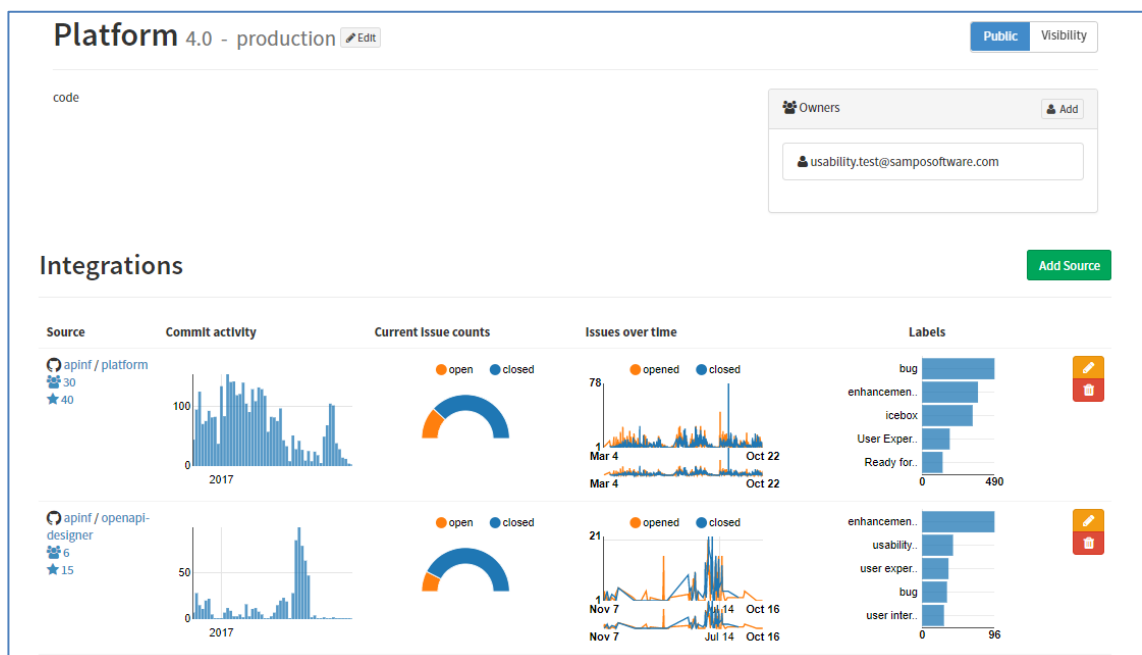
organization (e.g. modifying organization content, adding/deleting members, creating departments and services under them, etc.)

Under an organization exists one or more department which usually divide the overall works of an organization (e.g. If Ministry of Education and Culture is an organization, and then under this can exist the Department of General Education and Early Childhood Education). In real life, the digital services (e.g. either on-going or complete and in production) under a ministry are managed by different departments under it. So the similar hierarchy has been maintained in the structure of Dipor Dashboard. There are action buttons to create new departments and services under it and members with admin privilege for an organization can access them.

### 5.3 Service Integration View

The primary feature of Dipor Dashboard is its services and the integration view under it. Services refer to those ideas or solutions that are being evaluated by Digipalvelutehdas community through different life cycle phases or are managed by the organization if in production level. A service can have the following phases in its life cycle: **idea, design, deciding, development, proofOfConcept, alpha, beta, production, sunset, retired**. When adding a service, it needs to be associated with a version. Another aspect of a service is its visibility to the general audience. Usually, without login, the integration view can't be accessed by anyone. However, services that have private visibility can be accessed by only the members of an organization, who have been added to the integration page.

*Figure 5-3 Integration view of a service*



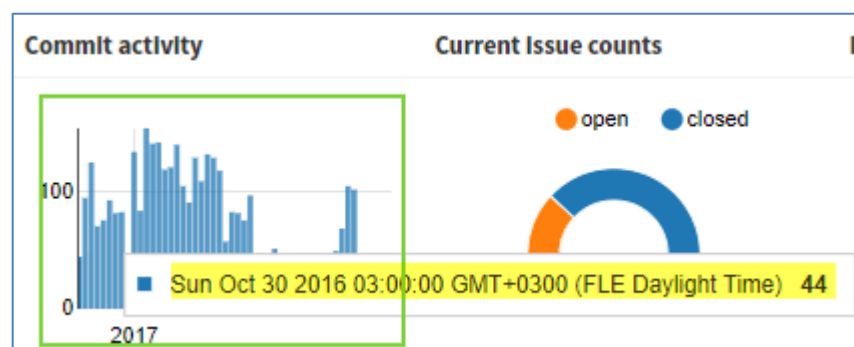
The concept of an Integration page under a service is as following. In real life, an idea or solution can be implemented by two development companies. Organization (and the department hosting the service) considers the development companies as competitors and wish to see who can produce a better testable solution for a digital service idea. This usually happens when the proof of concept of an idea is being evaluated for its feasibility and a testable solution helps in making decisions about the better solution (not required to have all expected features in it). So monitoring how active both the companies have been can be possible if repository information from their code can be projected together for comparison. This is achieved in Dipor with integration view.

To visualize a repository, the GitHub organization and the related repository names need to be added via Add Source dialogue. GitHub API<sup>1</sup> is used to pull information from the added repository and project in the Dashboard view. The information that are being emphasized in this view are: **Contributor** (people making pull request and commits to that repository), **Star rate** (number of people giving positive feedback to the repository), **number of commits**, **open and closed issues** in the repository and **issue labels**.

Name of contributors and people who gave star to the repository can be found by clicking associated icons (left most part). Other information obtained is visualized using a charting library called NVd3<sup>2</sup>.

**Commit activity** is a column chart. Individual bars show how many commits have been made on a week in this chart with Sunday being the first day of the week. Information about the week and number of pull requests can be seen by mouse hover on any column.

*Figure 5-4 Commit activity chart*

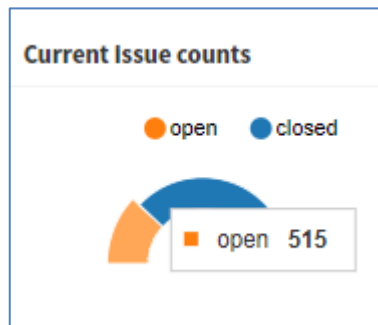


<sup>1</sup> <https://developer.github.com/v3>

<sup>2</sup> <https://nvd3-community.github.io/nvd3/>

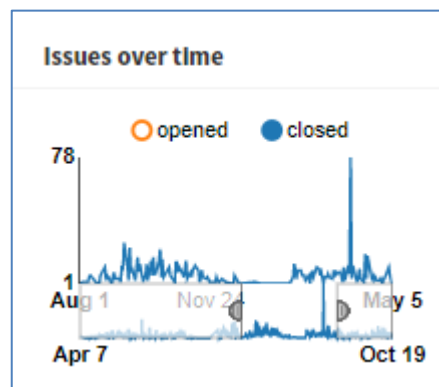
A donut chart<sup>1</sup> is used to show the ratio of **Open and Closed Issues** in the repository. This chart can be filtered with open, close or both types of issues. With mouse hover on the related part of the chart, number of open or closed issues can be obtained. Orange represents open issues and blue represents closed issues.

*Figure 5-5 Chart showing ratio of open and closed issues*



"**Issues over time**" is a line chart that shows the frequency of open and closed issue from the creation of the repository to the present date. This chart can also be filtered by open, close or both types of issues. This chart has a view finder, which is used to focus the timeline to some specific time range. This works as a zooming function to pin point issues for a specific time period. With mouse hover, number of open or close issue for a specific date can be learned. Also in this chart, orange line represents frequency for open issues and blue for closed issues.

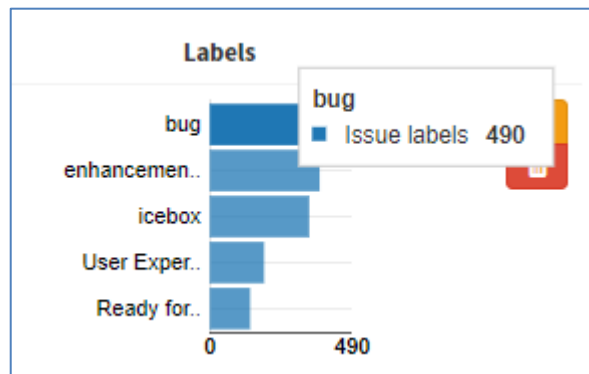
*Figure 5-6 Issues over time chart with view finder*



The last chart in the visualization is a bar chart that is used to categorize issues based on the issue labels used in that particular repository. The five labels containing the most number of issues are displayed in the chart with the label having the highest number of issues appearing on top. On mouse hover on any bar, number of issues associated with that label can be obtained.

<sup>1</sup> [https://datavizcatalogue.com/methods/donut\\_chart.html](https://datavizcatalogue.com/methods/donut_chart.html)

**Figure 5-7** Top five labels in the repository with associated number of issues



On refreshing the integration view, all charts get updated with the most recent data from the repository.

## 6. RESEARCH METHODOLOGIES: PHASE 2 – USABILITY EVALUATION OF DIPOR DASH- BOARD

This chapter describes the heuristic evaluations and usability tests conducted on the implemented Dipor Dashboard portal. The Usability evaluation of Dipor Dashboard was executed to understand how efficient and effective the dashboard view of the portal to visualize progress information of different digital service implementations with its overall look-and-feel and functionalities. It was also intended to find out how satisfying it is for the intended users to use this service in order to achieve their work goals regarding to digital service development monitoring.

The first section in this chapter presents method and results of the heuristic evaluation executed on Dipor Dashboard. The consecutive section gives an overview about the performed usability tests and their associated result.

*Figure 6-1 Timeline of Phase Two in Empirical Work*

<b>Event</b>	<b>Conducted works</b>	<b>Time Period</b>
<b>Heuristic Evaluation</b> Sessions: 2	<ol style="list-style-type: none"> <li>1. Executing one independent session each by two evaluators.</li> <li>2. Gathering results from two sessions.</li> <li>3. Recording the identified issues together with violated heuristics and severity ratings of the issues.</li> </ol>	1 <sup>st</sup> May, 2016 – 8 <sup>th</sup> May, 2016
<b>Usability Testing</b> Sessions: 5	<ol style="list-style-type: none"> <li>1. Preparing test tasks and configuring personal laptop for the test sessions</li> <li>2. Facilitating and recording both in-person and remote test sessions</li> <li>3. Identifying usability issue from records and hand-notes from individual sessions.</li> <li>4. Summarizing and recording usability issues along with their severity rating</li> </ol>	11 <sup>th</sup> May, 2016 – 31 <sup>st</sup> May, 2016

## 6.1 Heuristic Evaluation

After the development work of the Dipor Dashboard portal was completed and it was being tested for bug fixes and minor changes by Sampo Software Oy, a heuristic evaluation was conducted in the implemented system. This was done to be aware of the existing usability issues as much as possible before the usability tests began. The person(s) in charge of a heuristic evaluation go through the user interface and functionality of a system and judge its compliance with a set of recognized usability principles or heuristics. Heuristic evaluation was first introduced by Jacob Nielsen. He advised to involve multiple evaluators to ensure effective outcome from using this method as it is almost impossible for a single person to find out all usability issues from an interface (Nielsen, 1995b). As Dipor Dashboard was POC service implementation with a small number of features, it was decided to conduct two separate and independent heuristic evaluation sessions. One was conducted by the thesis worker herself. The second session was conducted by a research worker at Human Centered Design department at Tampere University of Technology.

### 6.1.1 Chosen Heuristics for Evaluation

The heuristics set chosen for this purpose included ten empirically defined new heuristics that were refined and improved from an examination of sixty three preexisting heuristics (Forsell & Johansson, 2010). This set of heuristics can be used to evaluate information visualization systems (e.g. a dashboard) as this covers up the explanation of problems done by the presented sixty three heuristics in the above studies. The ten heuristics are useful in carrying out expert evaluation of information visualization system when adequate details are provided with the description and evaluators have sufficient domain knowledge (Vääätäjä et al., 2016).

The dashboard view (including the flow to access the view) of Dipor Dashboard portal was evaluated using the given ten heuristics. In both evaluation sessions, whenever an issue was detected with the UI or functionality, a description of the issue and why it a problem was written down. One or more appropriate heuristics being violated by the detected issue was chosen to associate with the problem. A number (from zero to four) was also used to describe individual evaluation of the severity level of the identified issue. N/A label was used beside an issue if none of the heuristics was capable to describe the problem.

The descriptions of the ten heuristics used for evaluating Dipor Dashboard and the associated severity rating are given in the following pages. Heuristic descriptions are taken from this work (Vääätäjä et al., 2016).

*Table 6-1 Heuristics to Evaluate Information Visualization Systems (Forsell & Johansson, 2010)*

<b>Heuristics</b>	<b>Description</b>
<b>1. Information coding</b>	<p>Perception of information is directly dependent on the mapping of data elements to visual objects (graphing techniques, color, type and meaning of symbols, shading, transparency, etc.). This can be enhanced by using realistic characteristics/techniques or the use of additional symbols (legends, scales, drop lines, gridlines). Is the mapping correct? Is it appropriate for the task at hand, does it support user’s perceptual capabilities?</p> <p>Another important aspect is the use of alternative visual attributes or objects to represent information derived from the data like groups of elements in clustered representations.</p>
<b>2. Minimal action</b>	<p>Concerns workload with respect to the number of actions (sets of inputs) necessary to accomplish a goal or a task. The more numerous and complex the actions necessary are the more workload will increase. It is here a matter of limiting/minimizing as much as possible the steps users must go through.</p>
<b>3. Flexibility</b>	<p>Refers to the means available to the users to customize the interface in order to take into account their working strategies and/or their habits, and the task requirements. Flexibility is reflected in the number of possible ways of achieving a given goal. In other words, it is the capacity of the interface to adapt to the users’ particular needs. Example: permit users to control display configuration, to define, change or remove default values etc.</p>



<p><b>4. Orientation and help</b></p>	<p>Functions like support for the user to control levels of details, redo/undo of user actions and representing additional information (for example the path a user followed while navigating in a complex data structure) define help and user orientation features.</p>
<p><b>5. Spatial organization</b></p>	<p>Concerns user's orientation and awareness of location in the information space, the distribution of elements in the layout, precision and legibility, efficiency in space usage and distortion of visual elements. Is related to the overall layout of a visual representation and comprises analyzing how easy it is to locate and see an information element in a display (objects location) and to be aware of the own orientation in the information space, and the overall distribution of information elements in the representation (spatial orientation). Locating and analyzing an information element can be hard if some objects are occluded by others or if the layout does not follow a logical organization.</p> <p>Spatial orientation which contributes for the user being aware of the distribution of information elements is dependent on the display of the reference context while showing a specific element in detail. Concerns the possibility and easiness of specifying what information should be displayed in the context area vs. the detailed area, can the user control them separately or does selection in one area affect the other.</p>
<p><b>6. Consistency</b></p>	<p>Refers to the way interface design choices (codes, naming, formats, procedures, etc.) are maintained in similar contexts, and are different when applied to different contexts. The design choices will be better recalled, located and recognized if they are stable within the system (e.g. between screens or sessions). This way the system will be more predictable, learning and generalization are facilitated and errors are reduced.</p>

<p><b>7. Recognition rather than recall</b></p>	<p>Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate. Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to use shortcuts or tailor frequent actions for their own needs. (Focused on allowing the user additional options to sidestep regular interaction techniques)</p>
<p><b>8. Prompting</b></p>	<p>Means available to guide the user towards making specific actions whether these be data entry or other tasks. Refers to all means that help to know all alternatives when several actions are possible depending on the contexts. Concerns: status information, that is, information about actual state or context of the system, information about help facilities and their accessibility</p>
<p><b>9. Remove the extraneous</b></p>	<p>Concerns whether any extra information can be a distraction and take the eye away from seeing the data or making comparisons. Present the largest amount of data with the least amount of ink. This involves judging whether any extraneous information is a distraction and/or slow-down. Extra ink can be a distraction and take the eyes away from seeing the data or making comparisons. But removing too much can hinder the perception instead.</p>
<p><b>10. Data set reduction</b></p>	<p>Concerns provided features for reducing a data set, their efficiency and ease of use. Filtering allows reduction of information shown at a certain moment, leading more rapidly to adjustment of the focus of interest, and clustering allows representing a subset of data elements by means of special symbols, while pruning simply cuts off information irrelevant for the understanding of a visual representation.</p>

**Table 6-2 Severity Ratings of Usability Problems Identified (Nielsen, 1995c)**

Number	Severity Ratings of the Findings.
0	I do not agree that this is a usability problem at all
1	Cosmetic problem only: need not be fixed unless extra time is available on project
2	Minor usability problem: fixing this should be given low priority
3	Major usability problem: important to fix, so should be given high Priority
4	Usability catastrophe: imperative to fix this before product can be Released

### 6.1.2 Heuristic Evaluation Procedure

The two heuristic evaluation sessions were conducted independently by the evaluators at their own convenient time. Each evaluator used their own workstations to conduct the heuristic evaluations. The thesis worker used a Lenovo Ideapad with Windows 8 operating system and the Dipor Dashboard web portal was opened in the latest version of Chrome browser. She spent two hours in the Integration view in Dipor to identify issues. The second evaluator used his/her MacBook Pro laptop and ran the service in Safari web browser. S/he spent around two hours to conduct the heuristic evaluation. Both evaluators identified and recorded issues at the same time during their own session. After recording the identified problems with associated heuristic and severity rating, the second evaluator sent his/her results to the thesis worker. The thesis worker herself organized the identified issues, summarized and reported them in the thesis template.

### 6.1.3 Identified Usability Problems from Heuristic Evaluation

The issues identified in both heuristic evaluation sessions are summarized below. Each issue is associated a small description, name of the violated heuristic and the severity rating for the issue. Upon identifying a potential usability issue, both evaluators (the thesis worker and research worker from TUT) looked though the heuristic list to determine which heuristic was violated with the issue in question and what severity rate could be associated. If more than one heuristic was violated, it is also mentioned with the related severity rating. Issues containing two violated heuristics and associated severity ratings can also result from the independent evaluation performed by both evaluators.

The format of the problem description is following:

**<Issue number> <Name of the identified issue>**

**<Description of the identified problem>.**

**<Number and name of the violated heuristic> <Severity rating>,**

**<Number and name of the violated heuristic> <Severity rating>, ...**

The identified issues in the heuristic evaluation sessions are described in the following pages.

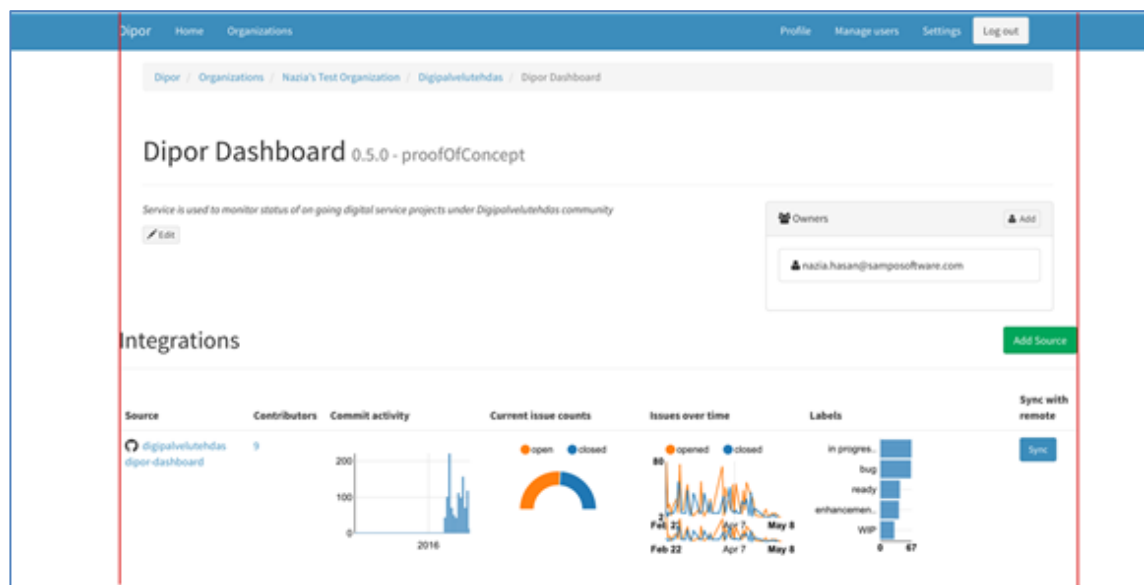
### 1. Unfamiliar relationship between system used and real-life terms

In service page, the term “Integration” does not indicate its associativity with GitHub repositories. It should be renamed with appropriate terms or help text can be included under info tip, but that reduces minimal action. [1. Information Coding] [2], [6. Consistency] [2]

### 2. Alignment issues in the view

Alignment should be maintained among GUI elements so that their borders (e.g. elements located at extreme left or extreme right of a page) have their borders in the same (imaginary) vertical line. [5. Spatial Organization] [1]

*Figure 6-2 Misalignment in dashboard view*



### 3. No choice is given for users to visit the actual GitHub repositories/version management systems

On clicking the GitHub icon or repository name, user can't get redirected to the original repository. This could become inconvenient as users would have to manually search for the repository in the internet [3. Flexibility] [2]

#### 4. No numbers associated with charts to represent actual or summed up amount

No Numeric figures and units (e.g. a total amount of issues) are provided along with the visualizations. This may create confusion in users to understand the exact value represented by the visualizations. Also users may have to go back and forth in the charts to understand the value represented by them. [7. Recognition rather than recall] [3], [2. Minimal Action] [2]

#### 5. No option to filter charts for a specific time range.

Commit activity chart shows data for consecutive week whereas issues over time may focus data for variable time range. It creates confusion among people when they are seeing data filtered by two different time range. Giving an option to select specific time period (e.g. last 24 hours, last 7 days, last 30 days) makes the data visualization simpler and more understandable to people. [6. Consistency] [3], [10. Data Set Reduction] [3]

#### 6. Cluttered view in case a page contains a large number of service integration

Four types of visualizations indicating different set of data are cluttered together in the same row. Here information becomes difficult to comprehend due to excessive data appearance, small size of visualizations and not having enough textual data (or having them in smaller texts) [1. Information Coding] [4], [5. Spatial Organization] [4]

*Figure 6-3 Cluttered view caused by too many charts.*



**7. Option for choosing specific information set to visualize is not present. (1, more or all at a time)**

If user is interested to see only specific attribute visualizations at a time, should be an option to allow him/her to select his/her preferred one(s). [3. Flexibility] [1]

**8. Visibility of Contributor Information**

Contributors of a service don't appear directly to user views. User needs to click on the number to see the people associated to it. A separate layout of visualization could be used to display contributor information. This information might include, name, no. of assigned issues, no. of pull request made and related contributor graph from GitHub. [5. Spatial Organization] [1]

**9. Visibility of textual information on charts**

User needs to hover on visualizations to know additional data about the information represented. They don't appear automatically. [2. Minimal Action] [2]

**10. Appearance of Data in Issues over time chart.**

The visualization is not represented using bigger space. So the data becomes cluttered and often difficult to comprehend. [5. Spatial Organization] [3]

**11. Option for browsing time data using focus pointer is not visible in the map.**

User may not know the existence of this option. User needs to click on map to activate the option and often with 1 click the option doesn't get visible [2. Minimal Action] [4], [7. Recognition than Recall] [4]

**12. Overview on monthly statistics is missing**

Since the services monitored in the dashboard are developed in a three-month timeframe, it would have been convenient to overview statistics for each month (or each

30-day period, depending on the starting day of the project). The same set of information but filtered within 30-day period can be presented apart from the regular information. This view can be static and can help people to understand how development work has progressed every month. **[8. Prompting] [2]**

### **13. No Information on start and completion date for a service is given.**

There is no option to manually add the start and due dates of a service or to configure them automatically while integrating the GitHub repository. It is possible to get insights about how the service development is progressing when information like closed issues, commit numbers, etc. are compared against these dates. **[4. Orientation and Help] [2]**

### **14. Chart axes are too small and incomprehensible (specifically with dynamic charts like issues over time)**

Because of the small size of the charts, no information is given on the axes about what is being measured with which unit. Only some numeric values appear with no indication about the used scale. When interacted with “Issues over time” chart, the change in axes information was too rapid to understand. **[1. Information Coding] [3]**



## 6.2 Usability Testing of Dipor Dashboard

After the completion of heuristic evaluation and categorizing the discovered issue, five usability tests were conducted. In usability testing, a product or service is evaluated by its intended users (Lewis & Raton, 2006). Participants in usability tests try to complete a given task set in the presence of observers watching and recording the session using audio, visual and hand written notes. Usability tests help to detect problems within user interface and associated functionalities of a system while end users are interacting with it to achieve some work goal. Detecting and fixing issues in early phases of development helps reduce cost in terms of resource allocation and schedules.

The conducted tests covered the main functionalities of Dipor Dashboard – primarily getting an idea about development progress from the information that was collected from GitHub Repositories and visualized in a dashboard. All the participants were given a set of tasks to perform using the dashboard view. All usability testing sessions included collecting data from user background and satisfaction questionnaire and interviewing briefly the participants with their thoughts about the process.

### 6.2.1 Participants in the Usability Tests

The selected participants were people who were the possible users of Dipor Dashboard. The first two participants were employed at the Ministry of Education and Culture and were recommended by the customer himself. They were not among the phase1 interviewees. So they were not familiar with the Dipor Dashboard System. These two usability testings were done at the Department for General Education and Early Childhood Education in the Ministry of Education and Culture in Helsinki, Finland. The 3rd, 4th and 5th participants were amongst the previous interviewees whose feedbacks were used to evaluate low fidelity sketches of Dipor Dashboard and build up the affinity diagrams continuously. Among the last three interviews, two were conducted remotely in Skype. The remaining interview was done in person at University of Tampere (UTA) premise.

The table in the next page contains the information about the participants in the usability testing

**Table 6-3** Background information of the Participant

	<b>Participant 1</b>	<b>Participant 2</b>	<b>Participant 3</b>	<b>Participant 4</b>	<b>Participant 5</b>
<b>Age</b>	28	30	41	34	35
<b>Occupation</b>	Employer	Service holder	Employer	Service holder	Student / Idea Innovator
<b>Education</b>	College / University Degree	College / university degree	Other	College / University degree	College / University degree
<b>Field Expertise</b>	Administrative Science	Business Administration and Economics	Expertise in user interface and information architecture	Software Engineering	Wellbeing technology for elderly people
<b>Computer Skills</b>	Good, uses often and fluently	Good, uses often and fluently	Excellent, knows how computer functions	Excellent, knows how computer functions	Excellent, knows how computer functions
<b>Familiarity with Digital Service Development</b>	Yes; product owner in some projects; negotiates requirements	n/a	Yes; product owner in some projects; negotiates requirements; needs updates about development progress to make decision about their continuation	Yes; product owner in some projects; negotiates requirements; needs to ensure development work is on schedule; needs to know if work is progressing efficiently with meeting the requirements; may wish to know continuation of other projects	Knows basic terminology with development work but never participated; interested to get updates of development works for favorite services
<b>Familiarity with any platform for Software Development Monitoring</b>	Trello	n/a	GitHub; Jira; Trello; Google Analytics	GitHub; Jira; Trello; Waffle; Google Analytics; Others	GitHub; Trello

<b>Frequency of Usage the above platform(s)</b>	Few times a month	n/a	Daily or nearly daily; few times a week; few times a month	Daily or nearly daily	Few times a week
<b>Previous use of Dipor Dashboard</b>	No	No	Yes; in less than a month; visited couple of times in a month; waiting for the portal to be included as part of his/her organization's work	No	No

### 6.2.2 Technical Aspects about the Conducted tests

All usability testing sessions lasted for one hour which included preparations for the tests, conducting the actual test and finalizing the recordings. The part of the session involving the participant lasted no more than forty five minutes. The web service of Dipor Dashboard was loaded from a Lenovo Ideapad laptop with Windows 8 operating system and 1366 X 768 screen resolution. The available browsers were Google Chrome, Mozilla Firefox and Internet Explorer and participants had the freedom to choose any. In remote usability tests, the participant's computer screen was shared on Skype to observe the interactions and navigation throughout the system while performing a given task. ManyCam software was used to record the interactions happening on the UI during the usability tests. For in-person test sessions, audio recordings of the tests were also made using voice recording software in mobile phone. Conversations in the remote usability testing sessions were recorded using Amalto call recorder extension for Skype.

### 6.2.3 Procedure of the Usability Tests

Each usability testing started with making introductions with the participant. Afterwards, participant was informed about the purpose of the test and given a brief description about Dipor Dashboard portal. In addition, test procedure was explained to the participant and at this point the consent for conducting and recording the test was collected from the participant. All the participants were assured about not revealing their identities and feedback to public audience except for the supervisor of the thesis. In addition,

each participant was asked to think aloud while they were performing a given task. After the participant had filled up a background questionnaire, the Think Aloud method was demonstrated to him/her by asking her to perform a pilot test task.

The actual test started once the participant completed the pilot task. There were 11 tasks focusing on the Dashboard view of Dipor Dashboard portal. The participant was given one task at a time and was asked to perform it in his/her own pace. Approximate time to complete each task was calculated before the tests. So if the participant was struggling to complete a task and was taking more time than the pre-calculated time, s/he was prompted to leave the task and start performing the next one. Notes were made separately by the thesis worker during the usability testing about success/failure of the tasks.

After completing the tasks set, the participant was given a satisfaction questionnaire to fill up. A small semi-structured interview was conducted to learn about the participant's feelings about the usability test and how the overall UI and functionality can be improved. The session was concluded by thanking the participant and giving a small token of gratitude for their contribution.

Participants who attended the usability tests remotely, they were asked to fill up the consent form, background questionnaire and user satisfaction questions via provided Google forms.


The recordings of each usability test were observed the same day the test session took place. Possible usability issues discovered during a test session were rated based on the severities they possessed. Feedback about UI and functionality and improvement suggestion given by the participants were also logged separately.

The form templates used in the usability testing sessions are provided in appendix C

#### **6.2.4 Tasks in the Usability Tests**

The following table contains the tasks that were used in all usability tests. Each task also includes its purpose in the testing. The explanation on determining the end of the task is also given. When each task was given to the participant, s/he was asked to read it aloud and then start executing it. No task included any extra materials. The task template used in the usability testing is provided in appendix C.

*Table 6-4 Tasks used in Usability Testing*

<p><b>1. Open a browser and go to Dipor Dashboard’s page and login with the given credentials:</b> <a href="https://dashboard.digipalvelutehdas.fi">https://dashboard.digipalvelutehdas.fi</a></p> <p>The purpose of the task was to determine if participant is able to find out Dipor Dashboard and login into the system</p> <p>The end of the task was determined when the participant was able to login successfully and was being navigated to the “Organization” view</p>
<p><b>2. You are really interested on works done by the Ministry of Education and Culture. You want to know more details about it. Find this organization and go to its page.</b></p> <p>The purpose of this task was to determine if the participant is able to navigate closer to the service development dashboard view. The hierarchy of the website is: Organization → Department → Services → Integration (the dashboard view)</p> <p>The end of the task was determined when the participant was able to successfully navigate to the profile page of the mentioned organization.</p>
<p><b>3. There are several departments under this ministry and each department hosts a number of services. Find out the list of services under Department for General Education and Early Childhood Education.</b></p> <p>Similar to the second task.</p> <p>The end of the task was determined when the participant was able to successfully expand the department view and locate the list of added services.</p>
<p><b>4. You wish to know about one of the services and associated information about its development. Go to the service page and find out its status, version and how it is visible to everyone.</b></p> <p>The purpose of the task was to determine if the participant is able to navigate to the dashboard view of a preferred service from the list.</p> <p>The end of the task was determined when the participant was able to successfully navigate to the Integration page of the chosen service. This also indicated that participant is now in the dashboard view of Dipor Dashboard.</p>
<p><b>5. You are interested to know which people are working under this service. Find out some of their names.</b></p> <p>The purpose of this task was to determine if the participant can find out names of the people associated with that particular repository of GitHub.</p> <p>The end of the task was determined when the participant clicked on  icon and opened the list of contributors in the repository.</p>

**6. You want to know how much active this service has been over time. Find out the number of commits from two consecutive entries and tell how the dates in these entries relate to each other.**

The purpose of this task was to determine if the participant is able locate the number of commits for two consecutive periods of time. It was observed to see if the participant can understand that the numbers are calculated on weekly basis.

The end of the task was determined when the participant was able to see the information from tooltip which appeared on hovering the mouse over “Commit activity” chart

**7. You are interested to see how many issues have been reported for this service. Find out the number of open issues and the number or closed issues.**

The purpose of this task was to determine if the participant can filter the “Current issue counts” chart and visualize the open and closed issues separately.

The end of the task was determined when the participant was able to successfully filter the number of open issues or closed in that repository using the given chart.

**8. You want more detailed information about open issues over a certain period of time. Find out about how many open issues were there between the times August 31, 2015 to December 23, 2015.**

The purpose of this task was to determine if the participant can reveal the functionality of focusing the “Issues over time” chart within the given time limits and determine the number of open issues in the focused view.

The end of the task was determined when the participant was able to use the view finder in the related chart and tell the number of open issues in the given time limit.

**9. It is easier for you to track issues if they are somehow categorized. Find out what are the different labels used for issues.**

The purpose of this task was to determine if the participant can find out different issue labels used in that repository

The end of the task was determined when the participant was able to locate the chart showing different labels from the given visualization

**10. You are interested about bugs that are produced when a service is developed. Find out the number of bugs under this service.**

The purpose of this task was to determine if the participant can identify how many issues with a particular label existed in the repository

The end of the task was determined when the participant was able to identify the number of issues under “bug” label from the given visualization

**11. You suddenly remember about a service that has similar development work. Add the given data source to get different information about that service development:**

**User: nrel**

**Repository: api-umbrella**

The purpose of this task was to determine how easy it is to add a new repository from GitHub to see the projected visualization

The end of the task was determined when the participant was able to successfully add the given repository in the system.

### 6.2.5 Results Obtained from the Usability Tests

The following table shows the task completion time, number of problems found and success criteria for each task for individual participants. Task outcomes are labelled with the following codes:

- A – Successful
- \*A – Partially Successful (with reason)
- B – Moderator help was required in performing the task
- C – Failed
- D – Suspended
- E – Not Tested (e.g. there was no more time to execute the task)
- d – Dependent on previous task.
- n – Procedure not expected

*Table 6-5 Task completion time, number of problems found and task outcome for all participants*

Test Task	Participant 1			Participant 2			Participant 3			Participant 4			Participant 5		
	Task Time	Number of problems	Task Outcome	Task Time	Number of problems	Task Outcome	Task Time	Number of problems	Task Outcome	Task Time	Number of problems	Task Outcome	Task Time	Number of problems	Task Outcome
<b>Task1</b>	2:04	1	A	0:30		A	1:02		A	0:55		A	3:36	1	AB
<b>Task2</b>	0:52		A	0:52	1	AB	0:59	1	A	0:27		A	1:05	1	A
<b>Task3</b>	0:28		A	0:44		A	0:20	1	A	0:42		A	0:44		A
<b>Task4</b>	1:30	2	AB	2:10	3	*A <sup>2</sup> B	0:57	1	A	1:26	1	A	1:37	1	C



<b>Task5</b>	1:06	2	AB	1:48	2	C	1:51	3	A	1:03	3	A	0:51	2	A
<b>Task6</b>	1:16		AB	1:00	1	C	5:00	5	*A <sup>1</sup> B	2:11	2	*A <sup>3</sup> B	2:32	4	BC
<b>Task7</b>	0:27		A	0:35		A	0:26		A	0:32		A	0:46		A
<b>Task8</b>	3:44	5	C	2:35	5	C	4:48	5	C	1:43	2	C	1:29	1	C
<b>Task9</b>	0:17		A	0:45	2	A	0:37		A	0:47	2	A	1:08	2	A
<b>Task10</b>	0:25		A	0:15		A	0:20		A	0:23		A	0:39		A
<b>Task11</b>	1:29	1	AB	1:11		AB	1:26		A	0:55	1	*A <sup>4</sup>	2:12	2	A

\*A<sup>1</sup> – could not find the how two entries are related and the time was up

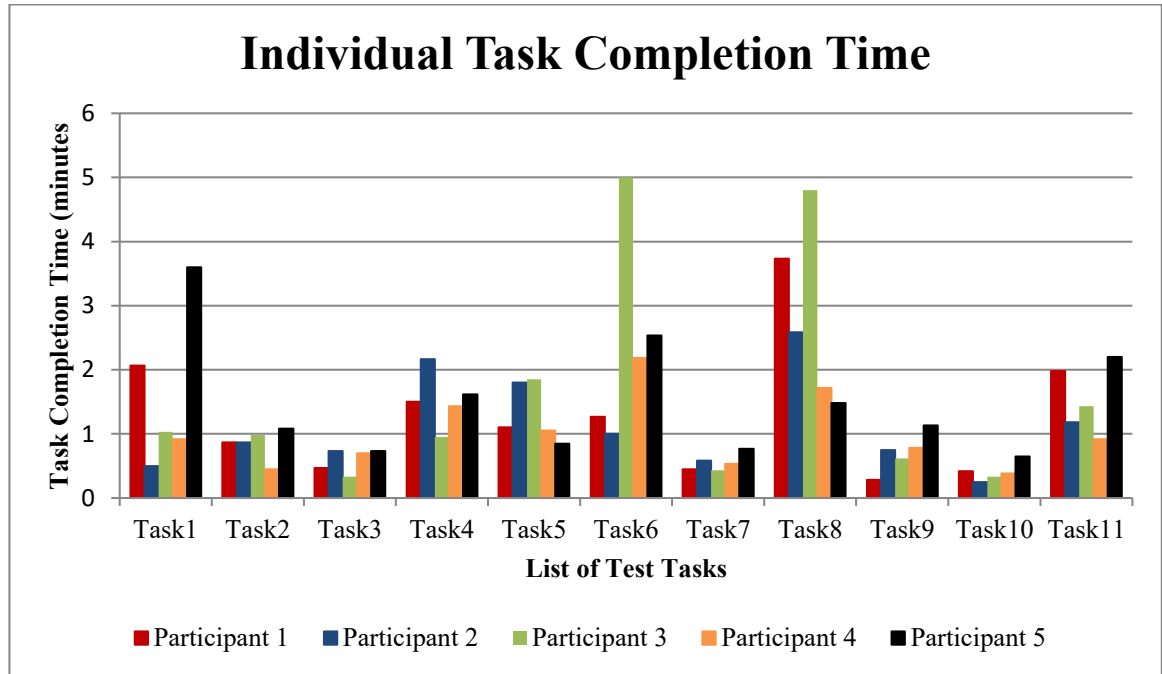
\*A<sup>2</sup> – wrong information (open close issue/issue over time) is perceived as status

\*A<sup>3</sup>- Charts didn't load in the first attempt. Had to reload the page in order to make all charts appear

\*A<sup>4</sup>- The repository was added successfully but the charts didn't appear after the add dialog was closed

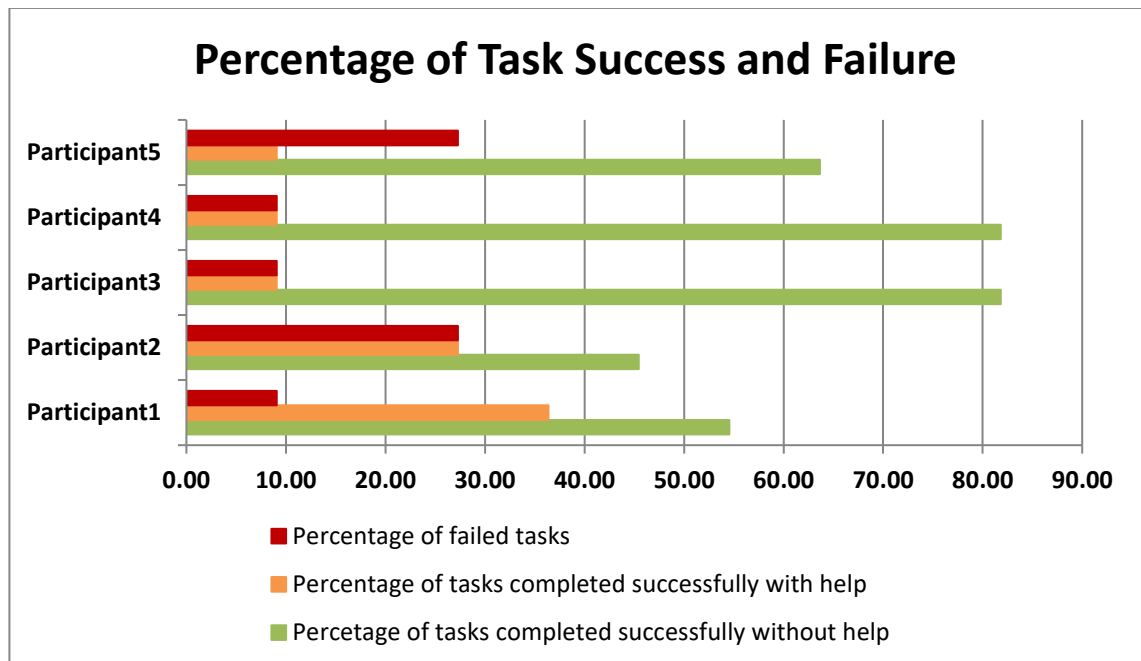
The following figure shows the task completion rate in minutes for individual participant in each task:

*Figure 6-4 Individual task completion time for each participant*



The following figure shows the percentage of successful and failed tasks for individual participants

*Figure 6-5 Success and failure rate in task completion for individual participant*



The table below contains the failed tasks and the reason the completion was unsuccessful. Information of only those tasks is provided which were failed to be complete by one or more participants.

**Table 6-6** Failed tasks, failure reasons and their occurrences

<b>Tasks</b>	<b>Reason for Failure</b>	<b>Occurrence</b>
<b>Task 4</b>	Participant couldn't locate the information of the service status amid others; s/he tried to guess that from available graphs but failed.	1
<b>Task 5</b>	Participant became uncertain that if both owner and contributor referred to people working actively in the project	1
<b>Task 6</b>	In one usability testing session, "Commit activities" chart didn't load. Another participant couldn't figure out the commit numbers beside date information in tooltip. S/he was unsure if clicks could be made on the chart.	2
<b>Task 8</b>	In two sessions, "Issues over time" chart got broken on making mouse clicks. The page needed to be reloaded for making the chart to reappear. No participant could discover the view finder in the chart to select specific time range. Clicking around the chart area often activated the filtering functionality which created some confusion. For the small UI and information format (date and content) one participant felt the task really complicated.	5

### 6.2.6 Feedback from Satisfaction Questionnaire and Interview

Each participant was requested to fill up a questionnaire after the usability test was finished. The questionnaire contained nine queries about the general impression on the look-and-feel and functionality of Dipor Dashboard. Participants were asked mark down how strongly they agree with the given statements

The following table summarizes overall feedback given by the participants about the platform usage during usability testing sessions. Frequency of chosen feedback for each criterion is shown as ( ) symbol in associated cell.

*Table 6-7 Feedback from the satisfaction questionnaire*

Feedbacks	Strongly Disagree	Disagree	I don't know	Agree	Strongly Agree
It was easy to learn to use the service					
I found the information I needed easily					
The appearance of the service was pleasant					
I am satisfied with the fluency of the use of the service					
The service included unfamiliar words and terms					
It was easy to perform the given tasks					
Using the service was frustrating					
I am going to use the service later					

Overall grade (scale: 1 = very poor to 5 = very good)	1	2	3	4	5

After filling up the satisfaction questionnaire, a short interview was conducted with each participant. The interview contained the following themes:

- 1. General Appearance of the dashboard view for Dipor Dashboard.** In this part, participants were asked how the dashboard view appeared to them in terms of colours, layout, font styling, etc.

**2. Different features and their functionalities in the dashboard view.** In this part of interview, participants were asked about their impression when they interacted with different features in the dashboard. It was also queried how much the available features helped them to obtain information they were seeking from

**3. Impression about its feasibility as a service to monitor development progress of on-going digital service implementation.** In the final part of the interview, participants were asked how they think about Dipor's feasibility to work as service development monitoring platform. Queries were made to know if they would be using the service in feature when it goes in production general access.

Answers from the participants obtained in the interview sessions are summarized below:

#### **Appearance of dashboard view in Dipor Dashboard**

Participants liked the simple layout and minimalistic colour scheme in the dashboard view. Use of white space was adequate which gave pleasant view in participants' eyes.

Participants had issues with the chart sizes as they were often difficult to interact with. Specifically Issues over time chart appeared messy to participants because of its size and content type. They would have preferred bigger size charts and information about their purpose.

Also Participants would like to see improvements about font sizes and style and how different information is presented together. There was an improvement suggestion to accommodate number of open and closed issues per label. They would like to see full names of labels or at least tooltips in case names are too long

Participants emphasized on associating numeric figures in big fonts along with the charts. They said number gives them the primary data and the charts act as a trend over time. A combination of both can become more informative

#### **Different features and their functionality in Dashboard View**

Participants found filtering options in the available charts straight forward and easy to use.

Used terminologies in the dashboard view were mostly unfamiliar to all participants, except one. They expressed their concern that how the terminologies would change when it would be possible to integrate different data source (e.g. Jira) in the dashboard view. Participants preferred that there should be common terminology associated with

visual elements or at least hints about their purpose, source and use.

Participants found it difficult to understand about the time period used to visualize data. This was apparent specifically with Commit activity chart and Issues over time chart. Their preference was to somehow filter the data based on time period (e.g. last 24 hours, last 7 days, last month, etc.)

Participants emphasized that the functionality of the charts and their contents should be redesigned. Often the charts broke down during the usability tasks and the page needed to be reloaded to make them reappear. In addition, if there are hidden functionalities, they should be visible without making extra clicks to appear them (referring to Issues over time chart's view finder). Also to some participants, design of the label chart should be reconsidered as it didn't contain all labels used in the particular repository.

The primary concern for participant was the small size of the charts as they were often difficult to interact with. The content in the charts also lacked clarity (specifically for Issues over time chart) because of the small size.

Participants wished to see some comparison in the presented information (e.g. percentage of closed issues comparing to last 24 hours.) This sort of information gives insights on development progress.

Participants said they would like to get access to original repository that is being used to generate the visualization. This could be possible for public GitHub repository.

### **Dipor Dashboard's feasibility as a digital service development monitoring platform**

Participants were uncertain on whether or not the provided features in the dashboard were sufficient enough to understand if a service implementation work is progressing well. There was no indication about service start and end date. Participants emphasized that they want to see number of closed pull requests, commits made and closed issues. Their said it is important to compare the amount with remaining time for the development to understand if indicates to positive outcome or negative.

Participants preferred that they see the dashboard as the first view when they login to Dipor. Although the logic was very simple, they felt going through three different hierarchies (Organization → Department → Services → Integration) was time consuming to access the dashboard part. The Integration view is considered favourable if someone wants to compare two service developments. But that should be a separate feature and not part of the main tasks of monitoring. For Dashboard, participants wished to see simple overview information about their favourite entities (e.g. service, organization and department) and if the status of these entities are good (e.g. a service having more

closed issues than open ones as it approaches its deadline looks positive). There can be a few widgets to customize these entities and their associated visualizations to appear in dashboard.

Participants emphasized that aside from a dashboard view, there should be a detail view for every service integrated to Dipor. This would make it possible to accommodate more visualizations and figures in appropriate size. A detailed view could work as Analytics and would help to give more insights about the project work, which might often not be possible from a glance in the dashboard.

Lastly as the services are being developed within a three-month period, participants preferred to look at monthly statistics for each month in the analytics view of a service. This would have given them an idea on how the development work progressed in each month.

### 6.2.7 Problems Found in the Usability tests

This sub-section scribes the problems that were identified in the conducted usability tests. The problems are numbered for quick reference. Each identified problem contains a title and a description on the details.

In addition a severity rating is associated with each problem. This rating indicates how the problem has been evaluated based on its impact on the system use. The severity ratings have been referenced from Nielsen (Nielsen, 1995c):

*Table 6-8 Severity Ratings of Usability Problems Identified*

Number	Severity Ratings of the Findings.
0	I do not agree that this is a usability problem at all
1	Cosmetic problem only: need not be fixed unless extra time is available on project
2	Minor usability problem: fixing this should be given low priority
3	Major usability problem: important to fix, so should be given high Priority
4	Usability catastrophe: imperative to fix this before product can be Released

It is important to mention that many of the issues identified during the usability tests were already encountered during the heuristic evaluation sessions of Dipor Dashboard. However, they were not fixed by the time the usability tests started. The final phase of the development project and high priority backend development works for Sampo Software Oy might be a few reasons behind this. So issues that are similar to the ones mentioned in the heuristic evaluation results are not repeated here.

The following list summarizes the problems found in the usability tests. There might be some problems not directly related to the dashboard view. But since they were discovered during the test sessions, they can be considered in future for improvement.

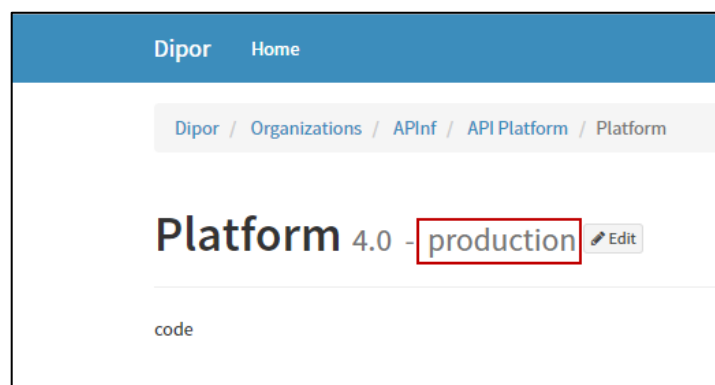
### **Problem 1: Login page of Dipor Dashboard takes long time to load**

In most cases the login page of Dipor Dashboard took more than 15 seconds to load. Possible reasons behind can be loading of JavaScript code scripts in the client side which blocks other processes until its execution is finished (“The most common reasons for a slow website response time,” n.d.). Similar case is applicable for block-rendering CSS and Fonts. Also data loaded per page, images with wrong dimension size can increase webpage loading time. Network speed can also be a crucial factor here. [2]

### **Problem 2: Terminologies used in the system are often vague (status)**

Participants often got confused when they were asked to identify in what is the present phase of the service they were exploring. It seemed that they were unfamiliar with the different phases (e.g. idea, design, proof of concept, production, etc.) a service goes through in its life time. [0]

*Figure 6-6 Location of a service status in Service Integration View*

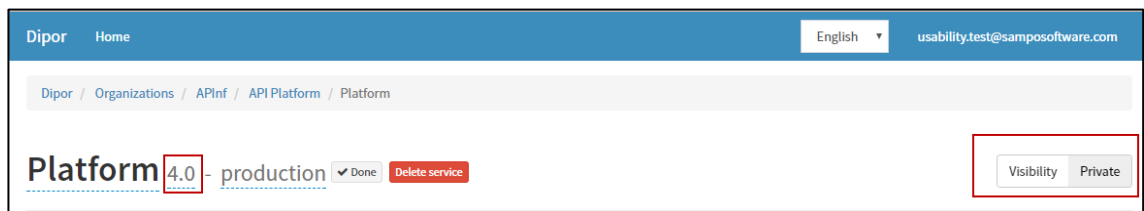




### Problem 3: Placement of information (version, status, etc.)

Participants had to look around for a while when they were asked to identify what is the current version of the service they are exploring. Also it took a while for them to figure out if the service can be accessed by general audience or only members within the organization where the service belongs to. These meta-information could be grouped together and place on a different level than service name for better visibility and identification. [1]

*Figure 6-7 Version number and Visibility option of a service*



### Problem 4: Charts UI often breakdowns on clicking

When random clicks were made on the area close to the charts, they often disappeared or didn't show any contents on them. This occurred with the donut chart showing "Current issue count" and "Issues over time" line chart. [4]

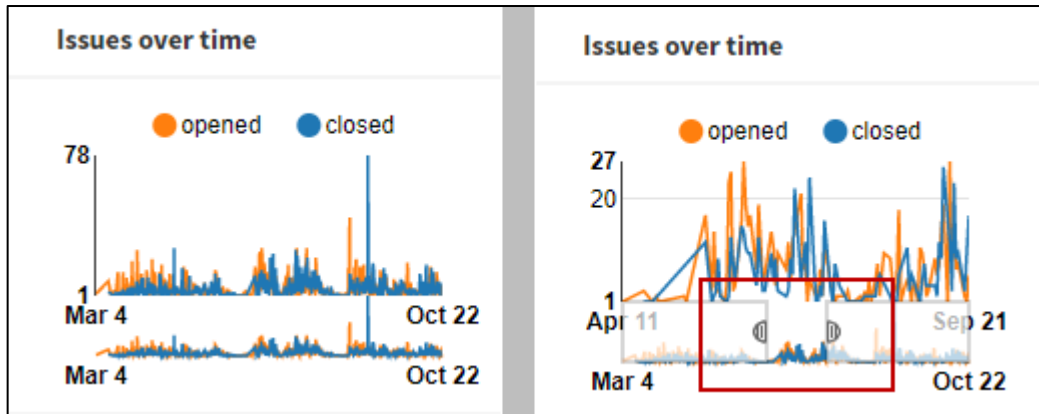
*Figure 6-8 Broken charts in the service integration view*



### Problem 5: View finder (if contained) within a chart is almost impossible to discover

The "Issues over time" line chart had a view finder to select time range in the chart. This could be activated by clicking on the small overview chart appearing under the main one. This didn't become visible automatically when the page was loaded. So the action was hidden in participants' view in all tests. This made the associated test task unsuccessful in all five usability tests. [4]

**Figure 6-9** View finder on Issue over time chart



**Problem 6: View finder (if contained) within a chart is slow to respond.**

One participant was capable to locate the view finder. But it was too slow to respond. So it was difficult to set the view finder to focus the given time period on the test task. This made the associated test task unsuccessful to execute. [3]

**Problem 7: Some charts don't load in the first attempt when Service Integration page is loaded.**

This scenario occurred when Integration page for a service was loaded for the first time with already existing repository visualizations. This also happened when a new source (i.e. GitHub repository) was added in the Integration view. The issue was verified by navigating to the original GitHub repository and obtaining the number of commits and issues in that particular repository. [4]

*Disclaimer: The added repository had no issues but 8,798 commits*

**Figure 6-10** Charts not appearing with first time page load.



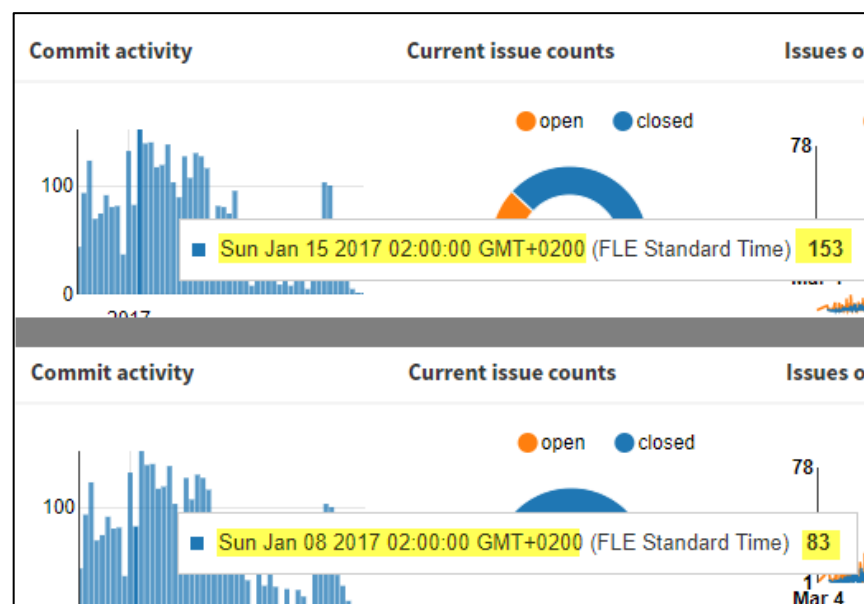
**Figure 6-11** Number of Commits in the original GitHub repository



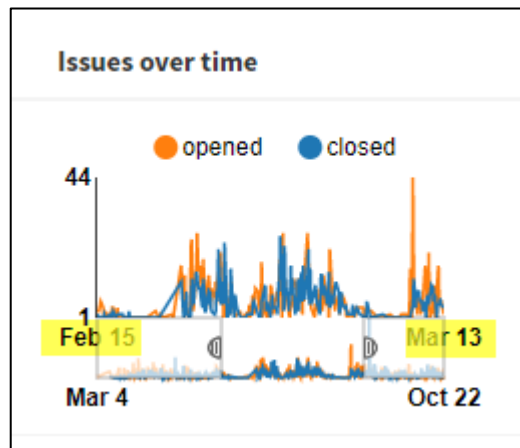
**Problem 8: Incomprehensive presentation of information.**

The figure below shows how commit information appears in the Commit activity chart for two consecutive bars. However, it was difficult for participants to understand how two bars are related to each other in terms of time range. Also the timing information appearing on the tooltip obscured the visibility of the number of commits. This is also applicable for Issues over time chart where the information appearing on the view finder doesn't indicate if the time is being measured per date or per year. [3]

**Figure 6-12** Unclear relationship between consecutive bars in Commit activity chart



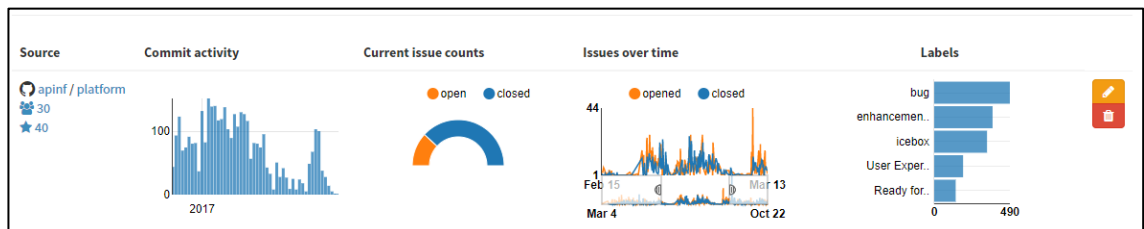
**Figure 6-13** Unclear date format in Issues over time chart



**Problem 9: Chart sizes are too small to interact efficiently.**

Because of the small size and restricted response area, it was often difficult for the participants to effectively interact with the given charts. Often clicking in an area in between two charts or a visual element (e.g. contributors) resulted with undesired (charts getting broken) and unexpected actions (activating chart filtering). [3]

**Figure 6-14** Small size of charts.



**Problem 10: Incorrect labels are used for data source repositories**

In GitHub, a repository is under an organization. However, on Add / Edit source dialog, organization was incorrectly labelled as user. Also full name of the terminology was not used. This could create confusion to a user habituated in using GitHub as s/he might not relate the association. [1]

**Figure 6-15** Incorrect and / or incomplete terminologies.

The screenshot shows a form titled "Add source". At the top, there is a "Type" dropdown menu with "github" selected. Below this is a section titled "Github" which contains two input fields. The first field is labeled "User" and the second is labeled "Repo". Both labels are enclosed in red rectangular boxes, indicating they are the focus of the usability issue. At the bottom right of the form, there are two buttons: "Add" (in blue) and "Cancel" (in grey).

**Problem 11: Quick action items are not informative enough (Add Source)**

For some participant it was unclear what the purpose of the Add Source button is. This was apparent when they are asked to execute the test task for adding a new repository in the Integration page. [1]

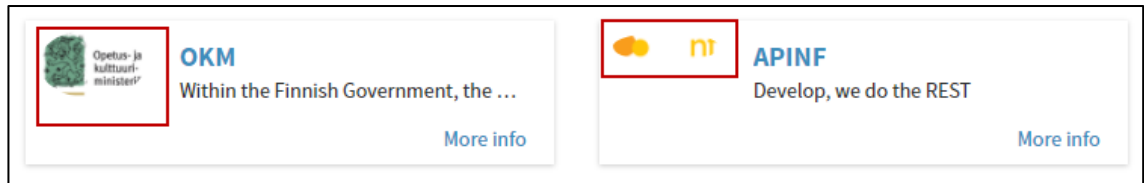
**Problem 12: No Search functionality is included in Dipor Dashboard**

While navigating towards Integration view, participant often tried to look for quick information about departments and services. They were expecting to look for them using a search bar, but found none. [2]

**Problem 13: Logos in organization cards are too small.**

In the Organization page, the card containing overview information about organization has placeholders for organization logo. The logos uploaded for an organization appears too small to understand its overall content. Also the logo placeholder on the cards doesn't show the entire logo. [1]

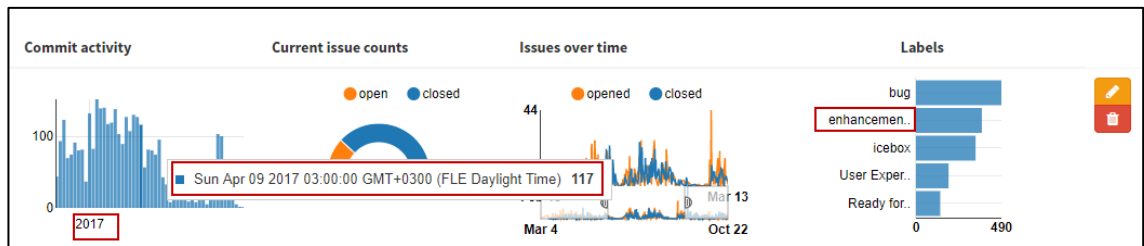
Figure 6-16 Small placeholder for organization logo



### Problem 14: Font size of texts is too small

Because of the sizes of the charts, associated text fonts (in tooltips, chart axis, etc.) appeared really small. Participants often had to bring their eyes close to screen to read texts. [2]

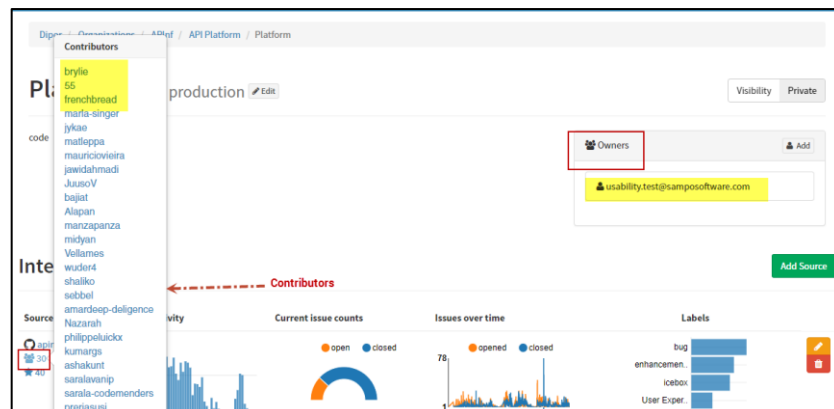
Figure 6-17 Small font size around dashboard view



### Problem 15: Information layout is often misleading (owner vs. contributors)

Participants found arrangement and contents of owner and contributors of a specific integration to be confusing. They asked if owner is somehow associated with the contributor team. The reason might be because of the usage of the same icon to represent different information. Also only an email address was shown to represent the owner, where contributors' information showed full names. [2]

Figure 6-18 Ambiguity in owner and contributors' information



**Problem 16: Redirecting to the original repository source isn't possible.**

In the integration view, an icon indicates the repository (e.g. GitHub) where the information is obtained from. Participants expected to be navigated to the original source repository on clicking the icon. But the icon was inactive and no redirection happened on clicking. [2]

*Figure 6-19 Inactive repository link*



## 7. RESEARCH METHODOLOGIES: PHASE 3 – LONG TERM USAGE STUDY

Keeping record of platform usage by target user group and learning about their success/failure stories in achieving their work goals can be effective in assessing if the platform in question was able to address user needs. This also helps to provide remedies to their problems. So a long-term usage study of Dipor Dashboard was scheduled after the completion of all usability tests. This chapter describes the methodology that was used to conduct the studies. In addition, the study procedure and obtained results from the usage study are also discussed

The following table contains the timeline for the conducted long-term usage study

*Table 7-1 Timeline for Phase three in Empirical Work*

Events	Conducted Works	Timeline
<b>Long term usage study</b>	1. Conducting online interviews with participants at the end of each study week. 2. Making written records of participants' insight from audio recording of the interview conversation	31 <sup>st</sup> May, 2016 – 14 <sup>th</sup> June, 2016
<b>Analysis of findings</b>	3. Analysed the obtained feedback	21 <sup>st</sup> June, 2016 – 30 <sup>th</sup> June, 2016

### 7.1 Description of the Method Used

To conduct a long term usage study on Dipor Dashboard, a research method named Multi-dimensional In-depth Long-term Case-study (Shneiderman & Plaisant, 2006) was chosen. This method is emerging well in HCI research field to evaluate and improve effectiveness in information visualization systems. This longitudinal study method observes expert users' interaction with the visualization system in question by documenting their usage via ethnographical participant observations, interviews, surveys, automatic logging of user activities in system, etc. Participants in this method are usually experts in their own work field. Before conducting the study, participants are given adequate training and are assured to provide aids and helps when needed. Usually the pro-



cedure contains logging users' activity in the system as well as maintaining diaries by participant to write down their interactions, discovered problems and feedback.

The aim for MILC is to develop multiple insights on the usage of visualization system by domain experts to solve their problems in their work field. On successful execution, this method reveals the outcome on how well the participants have been able to achieve their goals within their work domain. It also provides a set of suggestions on making improvements to the visualization system.

## 7.2 Description of the Study Procedure

This section presents details of the long-time usage study procedure that was adopted for Dipor Dashboard using MILC methodology.

### Participants

The initial plan for executing MILC method to evaluate Dipor Dashboard was to conduct a longitudinal study for four weeks involving three previously acquainted participants. One of the participants was the customer himself for the Dipor Dashboard project. The other two participants previously had given interview in phase one of the thesis empirical works and taken part in the usability testing sessions. The selection of participant was done to understand how they have familiarized with the developed system and if they are capable to determine status of digital service implementation works. In actual scenario, third participant was excluded from the study as he neither interacted with the system, nor responded on the interview sessions.

The following table contains the description of the participants in long term usage study. It is worthy to mention that both participants had knowledge and experience in Information Technology (IT) and Software Engineering (SWE) domains.

*Table 7-2 Participants of the Long-term Usage Study of Dipor Dashboard*

Serial No	Participant Category	Role	Remark
1	Customer of Dipor Dashboard	Head of Development, Department for General Education and Early Childhood Education, Ministry of Education and Culture	Close association with Dipor Dashboard's implementation work; was testing the service himself.
2	Possible user of Dipor Dashboard	System Designer in a public sector transportation organization in Finland.	Prior knowledge of Dipor Dashboard from interviews and usability tests

### **Goal of the Long-term Study and Participants' Success Criteria**

The goal of this longitudinal study was to understand if the associated features and functionalities of the Integration View for a Service in Dipor Dashboard were sufficient to determine development status of digital service implementation works. Participants were free to use the entire web service, but the thesis work kept its focus on the evaluation of the service integration view.

It was clarified with both participants on what they would describe as success to their everyday professional works. As both of them were in charge of digital development works in their relevant organizations, successful completion of project works within designated schedules were their work goals. For achieving this, they needed to monitor the development progress and decide if any of the projects needs their special attention due to impediments identified in the implementation work.

### **Study Procedure**

The main barrier towards executing the exact suggested guidelines by the MILC method was the location and work schedule of the participants. Both participants were living in Helsinki. So it was impossible for the thesis worker who lives in Tampere to be in close proximity for making regular observations on their usage activities of Dipor dashboard. In addition, both the participants were about to leave for their summer vacations, so they had busy schedules to complete their pending works and other obligations. So the following actions were agreed upon with both participants:

1. Use the dashboard view of Dipor Dashboard for fifteen minutes, each day of the weekdays for four consecutive weeks.
2. Write down their usage activities, identified issues and feedback regarding to their interactions and / improvement suggestions in online diary (provided as Google Doc).
3. Meet the thesis worker online at the end of each week and discuss about the findings.

Because of the difference in location, the primary observation was done during the online meetings when participants were pointing out issues and improvement ideas on the UI of Dipor Dashboard in their shared computer screens. Conversations of the online meetings were recorded using Amalto call recorder extension in Skype. As mentioned earlier, both participants were aware of Dipor Dashboard from either their association with the project or previous encounter in usability testing. So they received very little training about different features in the system. In addition, both of them had expertise in IT and SWE domains, so it was convenient for them to discover how Dipor Dashboard works on their own. The thesis worker however assured that they would be

free to ask for assistance any time they want. Both participants were supplied with credentials for an account having system administration privileges. It was not possible to acquire any software for automatically logging usage activities in Dipor for both participants. So the insights given by both participants were considered valid from the discussions that occurred in the weekly meeting and by keeping good faith on them.

### **7.3 Results from Long Term Usage Studies**

In real life, it was possible to conduct the usage study using MILC method for two consecutive weeks only. The participants didn't write down their usage information in the individual Google Docs provided for them. They also didn't use the service everyday (considering weekdays only) in each study week. The second participant didn't use the service from the second week of the study and he was unavailable for the interview. However, both participants had their thoughts and experience about the Dashboard portal and they expressed those in personal slack channel of Digipalvelutehdas or in Skype conversations during the appointed online meeting. The first participant (the customer) continued to use Dipor Dashboard for the first two weeks and provided his findings. However, he discontinued from third week. The possibility for the participants for this discontinuation might be because of their busy work schedules. Also since it was the start of summer holidays, they were not regularly available to continue the long-term usage study

The following tables contain summarized information about the answers provided by both participants as part of their long-term usage studies. The information about their usage, identified issues and feedback had been obtained from the audio conversations of the online interviews. Insights provided by each participant on their relevant session(s) are shown in separate table. The tables are categorized as per each week of the usage study

**Week 1**

<b>Participant 1 – Session 1</b>
<b>Activities within Dipor Dashboard</b>
Added one organization and department
Added a service in the created department and configured the Integration View with one repository
<b>Issue Found</b>
Participant had to reload Service Integration page twice before all the four graphs appeared on the view. It was frustrating to participant to see “No Data Available” text for charts as he knew the repository contained associated data
The size of the charts made reading data (for individual entries) or interaction with them very difficult. Information layout had poor readability, specifically chart axes values, time information and actual entries (e.g. Commits Activity chart)
The Issues over time had been the most difficult one to interact with. Participant had made random clicks before discovering the functionality of focusing the timeline in the chart for specific time intervals. Also the dates displayed were not intuitive enough for him. In addition, the small size of the chart made it difficult to comprehend the number of open and closed issues.
Participant found that Labels chart doesn’t show the full name of issue labels from GitHub. Also He didn’t find how many issues were open or close under a particular label. Also with another service he added separately, the labels in the Label graph appeared colliding with each other.
<b>Improvement Suggestions</b>
Participants emphasized on enlarging the charts, especially the ones containing time series values. He said only visualizations are not enough. There should be textual information indicating the amount or quantitative value visualized by the charts. Axes and their values should be clearer and information layout in the chart tooltip should be more intuitive

The participant feels that charts should indicate if there has been any change in data with time. This is important for charts with time series data (or has a focusing functionality). He emphasized on filtering the view for specific time intervals and showing the difference with respect to it.

To understand if development progress is going well or not, participant suggested customizing some value in the presented attribute. He said if all quantitative attributes can have some limit value set, there should be notification or alert when the chart shows trends that have exceeded the set limit. (E.g. if open issues with “Bug” label is more than 15, the service should notify the user on it.

Participant wanted some quick access to the Service Integration view for his favourite/owned services from the Dipor homepage. It would improve the deep navigation through related organization and department. Also he preferred to have quick navigation to the GitHub repository itself

Participant felt that for novice user seeing too many attributes in a smaller space might become confusing. He suggested about customizing which attributed to appear in the Integration view. The customization can be both default (1<sup>st</sup> time visit) and user specific (done by the logged in user) for individual accounts.

**Participant 2 – Session 1**

**Activities within Dipor Dashboard**

Participants used his/her own credentials than the provided one to create his/his own organization and services

**Issues Found**

Participant felt the access privileges of different user account to be confusing. S/he said that despite the provided credentials having super-user privileges in Dipor Dashboard, it couldn't find the service added by the participant's own credentials. Later s/he identified that service visibility needs to be public for all users to see it. Participant felt this somehow violates the logic for super user accounts in a system.

Participant asked if there had been any changes made to the charts to enlarge them. When answered negative, s/he said the graphs won't be useful to her/him in that case as it is difficult for her/him to track changes in data from miniature graphs.

Participant was a bit confused with the difference of people added to an organiza-

tion/department and contributors in a GitHub repository.
<b>Improvement Suggestions</b>
As stated earlier, participant emphasized on enlarging chart sizes. Especially the ones that are interactive and contain time series data
Participant wanted to see the development progress by following some Agile development methodology. S/e asked if a burn-down chart can be added there or the Issues over time chart could be redesigned as a burn-down chart.
Participant suggested adding another timeline chart for Pull Requests as well.
Participant suggested showing overall schedule of the service,
Participant also suggested considering a view to make comparison between two services under two separate organizations or departments.
The most significant feedback the participant gave is about filling Organization information (specifically members) from GitHub. Participant mentioned that GitHub has an information structure of Organizations, People in it and associated Repositories. People belonging to an Organization in GitHub has access to all repositories belonging to it This structure is quite similar to that of Dipor Dashboard. So in Dipor, instead adding organizations members manually, this information can be imported from GitHub’s instance of a that organization
Following the above feedback, participant also gave an idea about customizing team for each service repository. S/he said that the team can be customized by both active contributors in a repository and non-active contributors belonging to the same organization. There can be two separate commit graphs showing commit histories for active and non-active contributors. Participant said that if non-active contributors have made more commits than those of the active ones, then the team might be facing some crisis that can affect the development progress. So having such feature would be beneficial for monitoring service development.

**Week 2**

<b>Participant 1 – Session 2</b>
<b>Activities Within Dipor Dashboard</b>

Participant browsed around organizations and home pages
Modified some services he added before.
<b>Issues Found</b>
Participant was reluctant to use the platform because the graphics and the charts were not usable at that moment. He didn't get the overall idea about how the development work for a service was progressing. He also gave the reason that he needs to wrap up his other works before vacation, so he didn't also have much time to concentrate on Dipor
Participants could partially comprehend how the development work was going on by looking at the four available charts in the integration view. He said that "Commit activities" and "Issues over time" charts together gave him some idea about how development work is advancing. But nothing concrete. He wished to see views generated for these charts as per time. That would have given him trends on the exact situation. To the participant, it was more effective than having individual instances like how many bugs are open or how many labels are there in the issue
<b>Feedback and Improvement Suggestions</b>
Participant said that home page should be service focused than the current organization oriented one. There should be at least 3-5 latest services added to Dipor. There can be a separate section of latest organizations and a link to browse all of them. Also a searching functionality should be included for quick look up of organization or services.
Participant felt the organization and department profiles should also have some overview related to their services. Services can appear either as the ones with good progress or the ones that need attention. As mentioned in previous session, participant said this can be based on the value limits set for attributes of individual services.
Participant emphasized on using widgets to configure and visualize attributes for different services. This could be done for Integration view, department view, organization view or service. The widgets should be customizable for individual user accounts, so that people could organize their own dashboard as they see it fits.

## 8. DISCUSSIONS

This chapter presents detailed discussion about the feasibility of the implemented solution for Dipor Dashboard by analysing the platform in terms of dashboard design principles, existing systems and results obtained from usability evaluation and long-term usage studies. Feelings of participants from usability testings and MILC studies have been discussed in the User Experience section. In addition, a self-evaluation of the developed concept for Dipor Dashboard is presented. The feasibility of the developed concept is judged based on the feedback from usability testing sessions and longitudinal study conducted. Reliability and Validity of the obtained results from user interviews, usability testings and long-term usage studies are also discussed. The chapter concludes by answering how useful the methodologies have been for conducting the thesis work.

### 8.1 Analysis of Dipor Dashboard with respect to Existing Solutions and Dashboard Design

The primary purpose for implementing Dipor Dashboard was to provide a transparent outlook to general audience on how digital service implementation works are conducted in public sector organizations in Finland. The implemented proof of concept (POC) projected the hierarchy using the entities Organizations (e.g. different ministries or public sector institutions), Departments (secondary division of works under each organization), services (indicating part of the digital project development works of that organization under a specific department) and Integration (different instances of implementation work for that particular service). Integration view allowed to add open source repositories and projected repository related information with visualizations. The Integration view was meant to allow owner of a service to make comparisons between progresses of two separate implementation works being done for the same service. People having access to a specific service would have been able to obtain overview information and updates about its development works without directly visiting code repositories or version control systems. The Integration view of each service also allowed selecting different life cycle phases to associate with a service. This could be used to project the workflow process of Digipalvelutehdas community, which aimed pushing all iterations of an idea or solution development into three month long periods before testing the feasibility of implemented results. In existing market solutions for managing and monitoring software or digital service development, users have to go through a prolong (and sometimes complex) technical process of defining projects, user stories or requirements, multiple instances of minimum work unit to complete a user story, associating stories or work



units with specific development phases (milestone or sprints if done using Agile development), manually generating reports and graphs about development status, version release planning etc. The developed POC of Dipor Dashboard is unique in the sense that associated users would be able to get a familiar workflow and system hierarchy for public sector digital service development without manually navigating through or configuring a long and complicated process. For a three month long work to implement a testable solution with limited resource, the developed web service for Dipor showed fair results for visualizing work hierarchy and service development monitoring for Digitalvelutehdas community.

However, as part of the thesis work, we need to evaluate the feasibility of the developed solution as a Dashboard. Also we need to determine how well does Dipor Dashboard portal can stand among existing solutions and standards for designing dashboards.

The first limitation of Dipor Dashboard that can be identified is that Organization catalog had been added as the primary view either in anonymous or logged in view. Services or their integration views were not present in the home screen and couldn't be accessed quickly. A dashboard like view shows the most important information in a single screen for monitoring in a single glance (Few, 2006). The integration view should have been the primary view judging that monitoring service progress was the primary objective for Dipor Dashboard. Most competitors discussed in chapter 2 section 2.2 have a Dashboard view which is usually the user home page (on logged in session) or some separate menu in the navigation bar. Depending on the system, the view can be customized with reports, charts and information specific to user stories, milestones, etc. from different project. POC for Dipor didn't have similar UI or functionalities.

Integration view for a specific service was only meant to visualize information from multiple repositories aimed to develop the same service. The monitoring part was possible per service to compare activities of related repositories. So the user couldn't see at-a-glance status of all the services s/he has access to. There was a possibility that a user might add two repositories working on two different services (might not be belonging to the same organization as well) to monitor in parallel their development statuses. However, this would have surely created confusion and ambiguity in the overall system hierarchy. Integration view for a service included fixed visualizations which couldn't be modified or customized. Users couldn't add or remove any charts in the view. Visualization to show Pull Requests (PR) in a repository was missing. A PR in GitHub usually indicates some addition or modification in code proposed by a contributor and closing a PR usually closes one or more issues related to it. So this works as a pivotal information to know how development activities going on.

Both in the heuristic evaluation and long term usage study, the charts used in integration view appeared too small to interact and with defects. Important data couldn't be

highlighted in the integration view as no texts appeared along with the charts to tell users what information is being summarized by a specific chart. Textual information only appeared on tooltip when mouse was hovered on a chart. The information layout had issues with readability and comprehensiveness. “Issues in time” was supposed to be a timeline but the view finder to control the time range was hidden in the first glance. Labels chart contained incomplete information (open and closed issue numbers were absent). Since all attributes in the integration view showed values against time, showing all measurements in line charts would have ensured a cleaner UI. Showing 4 different information in 4 chart types (e.g. bar chart, donut chart, line graph and column) charts, in miniature sizes, in a single row caused information cluttering.

A limitation in the integration view was the lack of filtering the data appearing in the charts by specific time period. From the charts themselves, it was difficult to understand based on what time range the data is being shown. For this reason, there was no means to compare changes in trends in past instance of time. None of the integration view had any information about the schedule of the service, specifically its completion date.

As mentioned earlier, the development of public sector digital services is often done following agile development methodology. Despite GitHub repositories not supporting estimations in issues, many GitHub integration tools (e.g. Waffle) show burn-down chart for specific time range based on number of issues being closed within that range. But no visualizations like burn-down chart or Kanban workflow were present in the integration view in Dipor.

One crucial limitation for the integration view is that users would have needed to rely on their instinct to understand if a service development is going well or not. Within the system, there was no logic or functionality to determine if development was behind schedule. Lack of completion date and necessary measurements within the system made it impossible to predict future progress. Both activeness within the development schedule and future prediction are needed for measuring and monitoring development progress of a service (Jones, Rubin, Garmus, Putnam, & Clark, 2002). Lack of such functionality caused the charts to show no alert if one or more attribute(s) had anomalies in trends and needed attentions (Few, 2006). There was lack of overview information (something similar as GitHub pulse) to understand past performance.

## 8.2 User Experience

Satisfaction questionnaire from the five participants in the usability testings had positive results. Participants expressed about the ease of using the service and finding the infor-

mation they were looking for using the Integration view. They also talked about the simplicity and clean UI design as the first impression. It should be noted that tests task had descriptions that were prepared in the manner of storytelling (e.g. participant is in a situation and s/he needs executing some actions). As participants agreed on the ease of the tasks, this could be a reason behind Dipor Dashboard appearing favorable for them to use. In the interviews conducted after the usability tests, most participants however expressed that they can't tell for sure if Dipor Dashboard would be sufficient to aid them monitoring digital service development progress. They emphasized they need to familiarize themselves with the functionalities in the entire portal and use it for longer time in order to make decisions about Dipor Dashboard's feasibility.

The two participants in the longitudinal study were more vocal about their frustrations with unfixed impediments and newly discovered issues during their usage. Since one of the participants took part in an earlier usability testing, s/he was able to identify several issues and suggested possible improvements. However, during her/his turn in the long term usage studies, s/he saw that existing issues were not fixed and suggestions were not taken into the account because of finished development schedule by Sampo Software Oy. So his/her failed expectations could have been the reason for him not continuing the study after the first week. The customer, being another participant, was reluctant to use Dipor dashboard in production scale because he considered the service to be unfinished. For him, Dipor Dashboard turned out to be more organization centric than service development centric. He wished to see a redesign in the dashboard concept, used visualizations and the portal to focus on service development.

### **8.3 Evaluation of Developed Concept for Dipor Dashboard**

The design prepared for the concept of Dipor Dashboard was developed iteratively based the feedback received during the interview sessions conducted in the first phase of empirical works in this thesis. However, due to the unavailability of the interviewed participants, the final version of the designs could not be evaluated. The usability testing sessions and the long term usage studies were conducted to evaluate the usefulness and users' overall feelings about the implemented proof of concept for Dipor Dashboard. So there were little possibilities on spot to present the developed concept and seek users' feedback on the final designs. However, to understand how feasible the developed concept is and how much it could address users' need on monitoring digital service development progress; it is convenient that the design is compared against the obtained results and feedback from the last phases of empirical works.

The difference between the developed concept and the implemented instance of Dipor dashboard is the approach adopted to show information hierarchy. In the implemented Dipor Dashboard, the first level of hierarchy was Organizations. Although mainly in-

tended to monitor services, they could be only accessed via organization and departments. The concept design for Dipor Dashboard included separate accesses to organizations and services, with services kept in the central focus. Organization could be accessed by their separate catalog. Services could be accessed from either Dashboard view, through navigation within organization hierarchy and also from their catalog as well. Dashboard view was presented as the user home page in a logged in session. Since dashboard is meant to provide at-a-glance visualization to entities that need attention, the developed concept supported this. A message for OK status in the beginning of the dashboard in case all services having smooth development would have allowed users to concentrate on some other works.

One participant in the long term usage studies mentioned that s/he would have liked adding repositories of two different services to monitor which one is more active and has a better work progress. In the designed concept, the dashboard has the feature to add a service as itself or as the original GitHub repository. This functionality would meet the users' expectation of monitoring multiple services or multiple implementations of the same service in parallel.

The feature that distinguished the developed concept for Dipor Dashboard the most from the implemented instance would have been details view for each repository for a specific service. The limitation in the number of widgets in Dashboard view could be remedied in the details view. In addition, this view would have aided the user to make in-depth investigation of services that required attention because of detected impediments in development progress. In his feedback during the long term usage studies, the customer mentioned that comparing related attributes side by side could give him insights on how development work for a service is progressing. Allowing multiple visualizations as widgets side by side in Details view would have aided this need.

Details view would have provided means to add charts in suitable size for smooth interaction, which would have been difficult in Dashboard view. Details view in the concept design would have aided to project development using Agile methodologies by visualizing Kanban workflow and allowing WIP limits to be set in workflow phrases for detecting bottleneck. Since services in Digipalvelutehdas community are kept in a specific life-cycle phase for three months, the GitHub pulse alike visualization idea would have aided to show user trends in the development work for past 30-day instances. One participant in the long term usage study mentioned about including service schedules in the Integration View for Dipor. The concept design supported this improvement suggestion as it had start and end dates of a service within a life cycle phase.

One of the common issues found in both heuristic evaluation and feedback in long term usage study is the small size of the associated charts in Integration view. The size acted as a barrier for spontaneous interaction and good readability. In the developed concept, if implemented, charts in both Dashboard view and Service view would have addressed this problem. Dashboard view contains overview line charts for displayed attributes along with numerical information about attribute values for chosen time range. In addition to the overview charts, time series charts were shown in larger size. Charts contained sufficient textual information to indicate attribute value, measurement units and changes in trend from the past. Also symbols from GitHub associated with related attributes would have ensured familiarity for representing repository information among users.

From both user interviews and long term usage feedbacks, it had been mentioned that people would have liked to customize their own Dashboard with repository attributes that are important to them. The developed concept provided this feature in both Dashboard and Details view for a service. The customization of the dashboard was limited to three attributes per service for preventing information cluttering. The visualization and associated information for each attribute were projected in widget style, which would have been preferable by users. In addition, this customization would have also aided the need if a user preferred to see different attributes for different services. However, setting up such customization might not be beneficial, if user intended to use the dashboard view for comparison purpose as well.

The idea for setting thresholds for monitoring attributes within the repository was generated based on the interview notes from the customer and other participants. Some mentioned that if number of open issues is greater than that of closed issues, there might be some hidden obstacles hindering the development work within the team. In such case, user might need to communicate the team members and ask them details. One participant mentioned that if number of bugs is large, there might be some issues in the code. Setting up these various logics seemed feasible by using threshold values for attributes that could be obtained from GitHub repositories. The idea of setting thresholds and showing alerts on exceeding thresholds needs further testings to evaluate how suitable it is for monitoring development progress of a service implementation work.

However, some rooms for improvement had been discovered in the developed concept. Because of GitHub's inability to estimate issues, burn down charts couldn't be shown in the details view of a service. However if Jira is integrated in future in Dipor Dashboard, it would be possible to generate burn-down charts as issues in Jira can be estimated with their start/end dates. Also Jira by default has a burn down chart which could be included in the details view for a service. In addition, burn down chart had been favored by inter-

view participants and long term usage study users. So apart from the present Kanban view, a common framework for estimating both GitHub and Jira issues is needed to project agile development progress of these services.

There could have been possibilities of users wanting to customize the time for which they want to see the overview and details of a service. The developed concept currently contained data filtering based on last 24 hours, last 7 days and last 30 days. So customizing time range would not be possible in it.

Both the dashboard and details view in the developed concept had options to customize what attributes would have appeared in both views. However, detailed visualizations were not sketched for the non-appearing attributes in these views. Some more works and design ideas are needed to be studied in order to improve this limitation.

The developed concept for Dipor Dashboard was designed for using GitHub repositories. For aiding version control systems and information from Jira, additional studies are needed to conduct for developing a common data model to represent attributes from both systems.

The dashboard in the developed concept had alerts to show which services need user's attention. However, it is also important for notifying users whenever an alert occurs rather than waiting for the user to be aware about it after logging into the system. Automated notifications via email, SMS or within the system aid to this purpose. However, efficient notification design and their application in Dipor Dashboard wasn't part of the thesis work. So the developed concept lacked this feature.

## 8.4 Validity and Reliability

Reliability indicates the degree to which the same outcomes obtained from an experiment, test or measurement procedure can be produced repeatedly (Carmines & Zeller, 1979). Selected methodology(s) is considered to be valid, if it measures the attribute it is supposed to measure (Carmines & Zeller, 1979). Although closely associated with quantitative researches, the term reliability can also be linked with all kind of research works (Golafshani, 2003). Validity might not always be applicable for qualitative results, it is always important to make some qualifying checks on the adopted measurements (Golafshani, 2003). This section discusses both concepts to evaluate how successful have been the methodologies used to conduct the research work.

Developing the concept of a dashboard using iterative design and evaluation of low fidelity prototype is considered reliable. In every iteration, improvement suggestions

that sounded logical were taken into account and incorporated within the latest version of prototype B (the concept designed by thesis worker). Having participants of different field of expertise ensured that both technical and non-technical elements would be addressed in the proposed sketches. Feedback and design suggestions obtained from participants in usability tests indicated that the developed concept includes more or less all features (e.g. bigger chart size, quick overview and access to services, visibility of possible actions, details view of service, etc.) that the intended users would like to see in the original implementation of Dipor Dashboard. Also feedback received from the two week's long term usage indicated that participants are providing the similar opinion as those of the usability tests. There were also new improvement suggestions (e.g. customization option of available information, setting threshold values to monitor different attributes) that was accommodated in the concept design up to many extents. So the produced concept is considered valid to reflect how people monitoring digital service development would like to use a dashboard.

Conducting usability testings with a fair number of participants makes reliability higher. Two among five participants weren't interviewed before and they had very little idea about Dipor Dashboard project. In this scenario, identical issues were detected in almost all test sessions. Most participants were able to complete most of the tasks successfully with or without help from the facilitator (the thesis worker) in the usability test sessions. Failure in completing certain tasks could be caused due to: 1. existing usability issues or bugs in the system (e.g. discoverability of view finder in the Issues over time chart) 2. Issues undetected but emerged during the tests (e.g. chart related to a task not loading) and 3. exceeding of time limit allocated for each task. From satisfaction questionnaire, all participants agreed that the task descriptions were easy to understand and execute. If failed tasks are analyzed, it is seen that all participant failed to complete one common task. Apart from that, one task wasn't possible to complete by two participants and there were two individual tasks incomplete by two separate participants. For participants, using Dipor Dashboard wasn't frustrating and they were able to find out most of the information they were seeking off. For finding issues that can hinder seamless interaction within a system, conducting usability testings seems valid.

The reliability of long term usage study was reduced due to the number of participants. Also discontinuation of the service usage by participants could be another vital reason. Both participants mentioned about issues they found in Service Integration view of Dipor either during the longitudinal study or in earlier encounters (e.g. usability tests) with the service. However, Sampo Software completed the designated implementation schedule and no further development was initiated. So the implemented service still had the unfixed bugs and impediments discovered during the usability evaluation. Discontinuation of development also meant that no new features were added as improvement suggestions to the system. So technical, functional and aesthetics violating issues could

cause the reluctance within the participants for using the service. This reduced the reliability of the results as well. Not to mention the timing close to Finnish Summer vacation didn't favor the usage studies. Absence of the observer in the longitudinal study might partially impact on lower reliability. However, reports and feedback made by the participants seemed valid as they matched with those obtained during usability testings.

## 8.5 Usefulness of the Methodologies Selected

Among the methodologies used in different phase of the empirical work of the thesis, user interviews have been the most successful one. Although talking with Sampo Software Oy gave ideas on what system the company was going to implement, it was the customer interview which gave deeper insights on why such a new solution is needed amidst existing ones in the market and how it is going to aid the Digipalvelutehdas community. Interviewing personnel having domain knowledge in project management and software engineering helped to decide which aspect of the entire proof of concept project should the empirical work concentrate on. The most fruitful interviews have been with the possible intended users of Dipor Dashboard. Their opinions and feedback helped to shape up the designed personas for Dipor Dashboard. Interviewees aided the design process of developing the concept of a dashboard to monitor development progress of digital services. Their improvement suggestions for both designs (one prepared by Sampo Software Oy and the other being iteratively developed by thesis worker) were helpful in making necessary modifications in terms of UI and functionalities.

Usability evaluation was chosen to determine how easy the testable proof of concept for Dipor Dashboard was to use by its intended users. Conducting two heuristic evaluations ensured potential issues that might cause barrier in spontaneous interactions with the UI and functionality were identified earlier. With five usability testings, several bugs and impediments were identified by the participants. One of these impediments was visibility of available actions. None of the participants were capable of completing a test task that involved discovering and using the view finder functionality to focus issues within a time interval of an available chart. Possible reason for the emergence of such issues during the usability testings could be the time and phase of the actual implementation work. Development of Dipor Dashboard was approaching towards its completion and the team was mostly busy with finalizing backend and maintenance related activities. Changes made to the backend functionalities could have resulted with undetected effects in the UI and functionalities. However, it is considered good to have these bugs and impediments discovered, since the team could fix them when further development for Dipor Dashboard would be initiated. Participants provided with good feedbacks on how further improvements can be made in the system. In summary, it could be said that conducting usability evaluation was worth the effort.



The using of MILC procedure for conducting long term usage study can be considered partially successful. A good number of participants couldn't be recruited. It wasn't possible for the thesis worker to be present for observations when the participants were actively interacting with the dashboard view. The participants didn't fill up the provided diaries with entries about their usage interactions. It was not possible to continue the study for the desired length of period. A very probable reason for the above outcome could be the unfavorable timing. The Finnish summer holidays were about to start and the participants had other priorities before leaving for vacation. While keeping diary studies as a mean for assessing service usage, it should be kept in the mind that participants themselves need to be committed and dedicated in order to make the process successful (Bolger, Davis, & Rafaeli, 2003). Participants can grow monotony out of repeated responses, obligation to fulfill sections irrelevant to their experience and from unattractive medium for journal entries. In addition, the actual development time was completed before the studies started and no further investment was made from customer's organization to continue the service. This caused the product to lack a finished look. Also existing usability issues and bugs remained unfixed. This could have been another reason for the participants from discontinuing further usage of Dipor Dashboard. However, on interviews conducted each week, both participants gave the indication that Dipor Dashboard could indeed aid to their works for monitoring service development progress. But that would require dedicated time allocation for fixing existing issues, making the interface more interactive and customizable and keeping advanced features for monitoring purposes.

## 9. CONCLUSIONS

As a proof of concept to monitor development status of various public sector digital service implementations, the system for Dipor Dashboard, developed by Sampo Software Oy, provided fair outcomes. The web portal projected organization hierarchies maintained within public sector in Finland and demonstrated development and integration of open source, digital services within Digipalvelutehdas community. Results from usability testings proved the UI of Dipor dashboard to be simple and clean to the participants. Participants also expressed their ease in understanding how the system works

However, from the issues identified and feedback obtained from long term usage study, it is clear that Dipor Dashboard isn't yet fully competent in helping its users to determine how development work of different services are going on. If compared to competitor systems discussed earlier, the implemented proof of concept has a long way to go before achieving the polished UI and functionalities. The implemented Dipor Dashboard focused on hierarchies instead of services, which was contrary to the expectation of the customer. The dashboard wasn't the primary view as the name suggested. Instead it was implemented as an integration view visualizing information obtained from different GitHub repository for an intended digital service. At very best, the view could be useful to some extent in comparing which repository is more active. But there is no indication how well the development is going on. Neither there is any prediction about completeness on the service. Participants in usability testing expressed their uncertainty about Dipor's feasibility as a monitoring platform for digital service development. Unlike many GitHub Integrator tools, the integration view lacked Agile development progress visualization either as burn-down chart or Kanban workflow. The charts used in the integration view had readability and interaction issues. A lot of impediments were discovered during the heuristic evaluation and usability testing sessions which were not fixed. Due to the lack of further funding and conflicts with development schedule, Sampo Software Oy didn't continue any more development to improve the UI and functionalities of the System. The customer himself showed his reluctance to use it for Digipalvelutehdas community as the implemented service didn't meet his expectations.

### 9.1 Improvement Suggestion for the Implemented System

The implemented Dipor Dashboard system requires a complete redesign in term of look-and-feel and functionalities. The services should be considered as primary elements rather the organizations to match the intended purpose of the portal. If a dash-

board is considered, navigation should be designed so that the view is accessed quicker than going deep down a number of hierarchies. Dashboard should be redesigned in a way that accommodates a quick overview of development progress of either different services or multiple instances of the same services.

Major improvement is needed in the appearance, size, readability and interaction of the available visualizations for different GitHub repository attributes. Charts, specially line or sparkline, might be helpful in showing changes in attributes with time. However, numeric and textual information should also be associated with the charts to make the overall picture clearer.

In advanced level, a lot of customization options are needed in terms of available attribute visualizations and logics to determine development statuses of on-going services. This is important as development pace may vary from service to service. Also, attributes within a repository and their numeric values and type may hold differences in their significance from person to person. These logics might involve highlighting an attribute in user's visual periphery in case there is anomaly in the attribute's value for a given time period.

Logics for forecasting completion of a service would be helpful in aiding the users to use this platform for determining development progress. If not in dashboard, some separate view to investigate further information regarding to the repository attributes should be provided in Dipor Dashboard. Having separate detailed view for each service (or repository) would accommodate provisions for larger charts and newer visualizations. The visualizations might include means to show development progress via agile frameworks (e.g. burn-down charts, burn-up charts, Kanban workflow, etc.) as the services follow such development methodology. This would require studying the GitHub API to determine what information is available to construct such view. Also studying logic adopted by open source GitHub integrators to visualize burn-down charts using GitHub issue would be worth the effort. Since services are run on a 90-days development period, it would aid product owner to make decisions on further continuation if overview status per 30-days period is available. A separate service details view seems to be an ideal option to accommodate the above discussed features.

Last but not the least; Sampo Software Oy should consider integrating other code repository or version management system (e.g. Jira) apart from GitHub to accommodate wider selection of information sources in Dipor Dashboard. This requires studying about consolidation of information obtained from separate sources and implementing a common data model framework to represent attributes from different repository systems. A study conducted at Tampere University of Technology presented a concept of gathering and combining software engineering data from different issue management systems,

version management systems and monitoring platform (Mattila, Lehtonen, Terho, Mikkonen, & Systä, 2015). The study developed a model to mash up data obtained from different sources (Jira, Mercurial and Splunk) and projected a combined data in a single visualization. Ideas from this study could be considered for integrating new data repositories and version control systems.

## 9.2 Further Studies for Developed Concept

The first step to continue further work with the developed concept would be to evaluate the designed sketched with intended users of Dipor Dashboard. As mentioned in the discussions, the ideas suggested for the dashboard, visualizations, etc. in the concept matches mostly with the improvement suggestions given by participants in usability testings and long-term usage studies. However, there was no chance to evaluate the final look-&-feel and functionalities of the concept and to decide if further design iterations are needed. Obtaining feedback from users and providing the logics for its feasibility would help Sampo Software Oy while reconsidering changes in design and functionality of the implemented Dipor Dashboard.

Some more studies are needed to understand optimal way of customizing dashboards with preferred visualizations and information. This is especially important if different data sources (e.g. Jira in next phase) are to be made available for integration. As the dashboard currently allows customization of different attributes per service, we need to investigate if varieties in the attributes per service would create difficulties for users to understand which repository is working well in case they are comparing development progress of two instances of the same service. The two instances could be either two separate GitHub repositories or one GitHub repository and one Jira management system. The feasibility of introducing a separate view for comparing service development could also be measured in during this study. In addition, ideas of new widgets that could be added for customization should also be studied.

The logic of setting threshold values for GitHub repository attributes need to be practically evaluated. As mentioned earlier, considering a service being developed in a good pace might differ from person to person. Product owners use their experience and knowledge in forecasting the completion of service development work. So it needs to be determined that if customizing threshold values to repository attributes in order to understand service development status is beneficial and easy to use for users of Dipor Dashboard. In the book “IT Measurement: Practical Advice from the Experts” an idea for determining progress of a software project is proposed based on number of activities, their start and planned due dates and individual percentage of completion (Jones, Rubin, Garmus, Putnam, & Clark, 2002). Graphical visualization of planned vs. actual completion percentage of listed works is suggested in order to represent at a glance

view for determining progress of the software project. This could be easier to visualize if attributes from Jira are used as issues in Jira can have start and end dates. For GitHub issues, some studies are needed to understand if similar approach can be adopted by calculating the information from the dates a milestone is assigned to an issue and its closing dates. This also requires studying what metadata regarding to an issue could be retrieved from GitHub API.

The developed concept included visualizations to present Agile development workflow using Kanban board in the details view. Burn-down charts could be another possible option for projecting development progress following Agile methodology. However, some researches are needed to determine how GitHub issues could be used to display similar visualizations, considering issues can't be estimated in the original repository. Studying the logic in existing GitHub integration system that can visualize burn-down charts are recommended as well.

A new direction for research could be incorporating notification logic within the developed concept of Dipor Dashboard. It is important to notify or alert users instantly if there have been some significant changes within the project. Also it should be kept in mind that notifications don't become unnecessary source of interruption for users. The study and integration of a user-friendly notification system can be considered as a separate topic related to Dipor Dashboard.

## REFERENCES

- About milestones - User Documentation. (n.d.-a). Retrieved October 26, 2017, from <https://help.github.com/articles/about-milestones/>
- About milestones - User Documentation. (n.d.-b). Retrieved October 26, 2017, from <https://help.github.com/articles/about-labels/>
- Ahmad, M. O., Markkula, J., & Oivo, M. (2013). Kanban in software development: A systematic literature review. In *2013 39th Euromicro Conference on Software Engineering and Advanced Applications* (pp. 9–16). IEEE. <https://doi.org/10.1109/SEAA.2013.28>
- Arhippainen, L., & Tähti, M. (2003). Empirical evaluation of user experience in two adaptive mobile application prototypes. *Proceedings of the 2nd International Conference on Mobile and Ubiquitous Multimedia*, 27–34. Retrieved from <http://www.idealibrary.com>
- Beyer, H., & Holtzblatt, K. (1999). Contextual design. *Interactions*, 6(1), 32–42.
- Biehl, J. T., Czerwinski, M., Smith, G., & Robertson, G. G. (2007). FASTDash: a visual dashboard for fostering awareness in software teams. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 1313–1322. <https://doi.org/10.1145/1240624.1240823>
- Bitner, J. (2012). Managing Projects with GitHub. Retrieved November 11, 2017, from <https://www.lullabot.com/articles/managing-projects-with-github>
- Bolger, N., Davis, A., & Rafaeli, E. (2003). Diary Methods: Capturing Life as it is Lived. *Annual Review of Psychology*, 54(1), 579–616. <https://doi.org/10.1146/annurev.psych.54.101601.145030>
- Budiu, R. (2014). Memory Recognition and Recall in User Interfaces. <https://doi.org/10.2307/westhistquar.45.1.0103>
- Burger, R. (2016). 7 of the Top Agile Project Management Software. Retrieved October 20, 2017, from <https://blog.capterra.com/agile-project-management-software/>
- Calabria, T. (2004). An introduction to personas and how to create them. *KM Column*, (March). Retrieved from [http://www.steptwo.com.au/papers/kmc\\_personas/](http://www.steptwo.com.au/papers/kmc_personas/)
- Carmines, E. G., & Zeller, R. A. (1979). *Reliability and validity assessment* (Vol. 17). Sage publications.
- Chang, Y.-N., Lim, Y.-K., & Stolterman, E. (2008). Personas. In *Proceedings of the 5th*

*Nordic conference on Human-computer interaction building bridges - NordiCHI '08* (p. 439). <https://doi.org/10.1145/1463160.1463214>

- Cho, J. J. (2010). An Exploratory Study on Issues and Challenges of Agile Software Development with Scrum. *Issues in Information Systems*, 9(2), 599. Retrieved from <https://pdfs.semanticscholar.org/713c/41d6f352f97a94af1af94c56e0ba2584effd.pdf>
- Cooper, A. (2004). The origin of personas. *INNOVATION-MCLEAN THEN DULLES VIRGINIA-*, 23, 26–29.
- Cotgreave, A. (2011). 6 Best Practices for Creating Effective Dashboards. *Tableau*, (August), 15. Retrieved from [https://www.tableau.com/sites/default/files/media/whitepaper\\_6bestpracticesforcreatingeffectivedashboards\\_engfinal.pdf](https://www.tableau.com/sites/default/files/media/whitepaper_6bestpracticesforcreatingeffectivedashboards_engfinal.pdf)
- Dan, W. (2010). Why Lean And Agile Go Together. *Forbes*. Retrieved from <https://www.forbes.com/2010/01/11/software-lean-manufacturing-technology-cio-network-agile.html#2df6fcc5b09b>
- Developer.android.com. (2014). Widgets - Android Developers. Retrieved from <https://developer.android.com/design/patterns/widgets.html>
- Dewalt, J. (2016). GitHub Issues Can be Agile if You Do it Right. Retrieved October 27, 2017, from <https://zube.io/blog/agile-project-management-workflow-for-github-issues/>
- Dugas, C. (2014). Agile workflow with GitHub issues. Retrieved October 29, 2017, from <http://www.position-absolute.com/articles/agile-workflow-with-github-issues/>
- Few, S. (2006). *Information Dashboard Design: The Effective Visual Communication of Data* (1st ed.). California: O'Reilly Media.
- Forsell, C., & Johansson, J. (2010). An heuristic set for evaluation in information visualization. In *Proceedings of the International Conference on Advanced Visual Interfaces* (pp. 199–206).
- Golafshani, N. (2003). Understanding reliability and validity in qualitative research. *The Qualitative Report*, 8(4), 597–606.
- Goodwin, K. (2001). Perfecting your personas. *Cooper Interaction Design Newsletter*, 295–313.
- Ian, B. (2014). How We Use GitHub Issues To Organize a Project. Retrieved November 11, 2017, from <http://www.ianbicking.org/blog/2014/03/use-github-issues-to-organize-a-project.html>
- Ikonen, M., Pirinen, E., Fagerholm, F., Kettunen, P., & Abrahamsson, P. (2011). On the impact of Kanban on software project work: An empirical case study investigation. In *Proceedings - 2011 16th IEEE International Conference on Engineering of*

*Complex Computer Systems, ICECCS 2011* (pp. 305–314).  
<https://doi.org/10.1109/ICECCS.2011.37>

- Jones, C., Rubin, H., Garmus, D., Putnam, L., & Clark, E. (2002). *IT measurement: Practical Advice from the Experts* (1st ed.). Addison-Wesley Professional.
- Kanban WIP limits – Work in Progress limits | Kanban Tool. (n.d.). Retrieved November 13, 2017, from <https://kanbantool.com/kanban-wip-limits>
- Karlesky, M., & Vander Voord, M. (2008). Agile Project Management (or, Burning Your Gantt Charts). *Embedded Systems Conference Boston (Boston, Massachusetts)*, (October 2008), 1–16. <https://doi.org/10.1145/1101779.1101781>
- Kaulio, M. A. (1998). Customer, consumer and user involvement in product development: A framework and a review of selected methods. *Total Quality Management*, 9(1), 141–149.
- LeanKit Inc. (2015). What is a Kanban board? Retrieved November 13, 2017, from <http://leankit.com/learn/kanban/kanban-board/>
- Lewis, J. R., & Raton, B. (2006). Usability Testing. *Human Factors*, 6–8. Retrieved from <https://www.usability.gov/how-to-and-tools/methods/usability-testing.html>
- MAGUIRE, M. (2001). Methods to support human-centred design. *International Journal of Human-Computer Studies*, 55(4), 587–634. <https://doi.org/10.1006/ijhc.2001.0503>
- Mahnic, V., & Zabkar, N. (2012). Measuring progress of scrum-based software projects. *Elektronika Ir Elektrotehnika*, 18(8), 73–76.
- Mattila, A.-L., Lehtonen, T., Terho, H., Mikkonen, T., & Systä, K. (2015). Mashing up software issue management, development, and usage data. In *Rapid Continuous Software Engineering (RCoSE), 2015 IEEE/ACM 2nd International Workshop on* (pp. 26–29).
- Nakazawa, S., & Tanaka, T. (2015). Prototype of Kanban Tool and Preliminary Evaluation of Visualizing Method for Task Assignment. In *Computer Application Technologies (CCATS), 2015 International Conference on* (pp. 48–49).
- Nakazawa, S., & Tanaka, T. (2016). Development and Application of Kanban Tool Visualizing the Work in Progress. In *Advanced Applied Informatics (IIAI-AAI), 2016 5th IIAI International Congress on* (pp. 908–913).
- Nielsen, J. (1995a). 10 Usability Heuristics for User Interface Design. *Conference Companion on Human Factors in Computing Systems CHI 94*. Nielsen Norman Group. <https://doi.org/10.1145/191666.191729>
- Nielsen, J. (1995b). How to Conduct a Heuristic Evaluation. *Nielson Norman Group*. Retrieved from <https://www.nngroup.com/articles/how-to-conduct-a-heuristic-evaluation/>
- Nielsen, J. (1995c). Severity Ratings for Usability Problems. *Papers and Essays*, 54,



- 11–99. Retrieved from <https://www.nngroup.com/articles/how-to-rate-the-severity-of-usability-problems/>
- Ogle, D. (n.d.). Persona Categories - Fluid - Fluid Project Wiki. Retrieved October 27, 2017, from <https://wiki.fluidproject.org/display/fluid/Persona+Categories>
- Paredes, J., Anslow, C., & Maurer, F. (2014). Information visualization for agile software development. *Proceedings - 2nd IEEE Working Conference on Software Visualization, VISSOFT 2014*, 157–166. <https://doi.org/10.1109/VISSOFT.2014.32>
- Rouse, M., & Sorenson, M. (2005). What is dashboard? Retrieved November 17, 2017, from <http://searchcio.techtarget.com/definition/dashboard>
- Ruhe, G., & Wohlin, C. (2014). *Software Project Management in a Changing World*. Springer.
- Salo, R. (2014). *A guideline for requirements management in GitHub with lean approach*. University of Tampere. Retrieved from <https://tampub.uta.fi/bitstream/handle/10024/95846/GRADU-1404127019.pdf?sequence=1>
- Shneiderman, B., & Plaisant, C. (2006). Strategies for Evaluating Information Visualization Tools: Multi-dimensional In-depth Long-term Case Studies. *Proceedings of the 2006 AVI Workshop on BEyond Time and Errors: Novel Evaluation Methods for Information Visualization - BELIV '06*, 1–7. <https://doi.org/10.1145/1168149.1168158>
- The most common reasons for a slow website response time. (n.d.). Retrieved October 23, 2017, from <https://www.quora.com/What-are-the-most-common-reasons-for-a-slow-website-response-time>
- Treude, C., & Storey, M. (2010). Awareness 2 . 0: Staying Aware of Projects , Developers and Tasks using Dashboards and Feeds. *International Conference on Software Engineering (ICSE)*, 365–374. <https://doi.org/10.1145/1806799.1806854>
- Tufte, E. R. (2006). Beautiful evidence.
- User Guide | Agilefant. (n.d.). Retrieved October 20, 2017, from <https://www.agilefant.com/support/user-guide/>
- Väättäjä, H., Varsaluoma, J., Heimonen, T., Tiitinen, K., Hakulinen, J., Turunen, M., ... Ihantola, P. (2016). Information Visualization Heuristics in Practical Expert Evaluation. In *Proceedings of the Beyond Time and Errors on Novel Evaluation Methods for Visualization - BELIV '16* (pp. 36–43). <https://doi.org/10.1145/2993901.2993918>
- VAN VELSEN L, VAN DER GEEST T, KLAASSEN R, S. M. (2008). User-centered evaluation of adaptive and adaptable systems: a literature review. *The Knowledge Engineering Review*, 23(3). <https://doi.org/10.1017/S0269888908001379>
- Viewing a summary of repository activity - User Documentation. (n.d.). Retrieved October 27, 2017, from <https://help.github.com/articles/viewing-a-summary-of->

repository-activity/

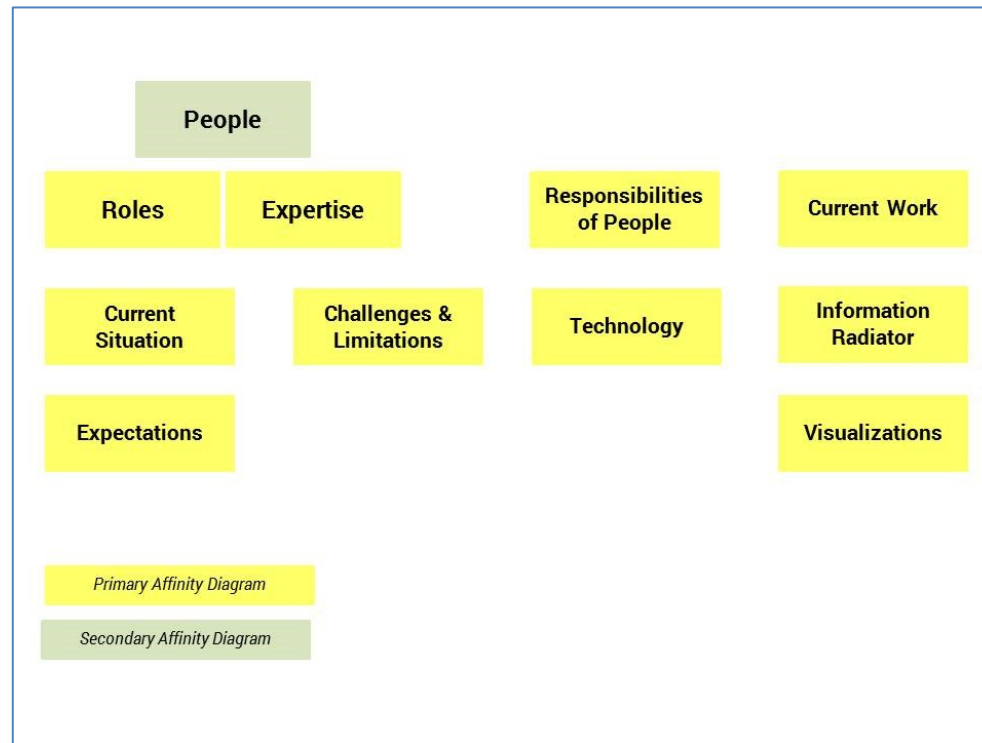
What is A/B Testing? (n.d.). Retrieved October 21, 2017, from <https://www.optimizely.com/ab-testing/>

What is a burn up chart? (n.d.). Retrieved November 13, 2017, from <http://www.clariotechnology.com/productivity/blog/whatisaburnupchart>

Wilson, C. (2013). *Interview techniques for UX practitioners: A user-centered design method*. Newnes.

## APPENDIX A: AFFINITY DIAGRAMS

*Figure 0-1 Overview of affinity diagrams*



## AFFINITY DIAGRAM A-1: PEOPLE

### People

**MM 1 (RO):** My Title in my company is **Development Manager**, but actually I am the team leader

**MM 6 (RO):** I usually maintain the same role throughout all the projects

**MM 6 (RO):** We have subcontractors in the MPass Project.

**TuH 5 (RO):** I am system designer in my company.

**JM 2 (RO):** My Primary role is to act as the **product owner** of these services.

**MM 3 (RO):** I am most likely the **administrative person** who does the budgets and allocates work.

**VK 7 (RO):** In all projects, one important concept is facilitating. So for each ongoing project, we have a **facilitator**.

**MM 31 (RO):** We also have different partners considering to different projects. We have municipalities, wellbeing organizations. In addition, we have private sector subcontractors doing different subparts of a project.

**JM 1 (RO):** I work as **development manager** at one of the departments of the Ministry of Education and culture. It is Department for General Education and Early Childhood Education.

**VK 1 (RO):** I am the head of Tampere Demola

**VK 15 (RO):** The companies or institutions who are acting as our project partners need to have some personnel who is interested with the projects and willing to see the teams in every couple of weeks.

**VK 21 (RO):** We have project partners from both private sectors and public sectors (e.g. ministries, cities, institutes, associations, foundations, etc.).

**TH 1 (RO):** I work with in the Department for General Education and Early Childhood Education under Ministry of Education and Culture and my title is **special adviser**.

**JM 9 (RO):** We have our Finnish citizens who can suggest an idea or can list problem(s)

**MS 13 (RO):** I study in TAMK and I am doing my Master's thesis in **Wellbeing Technology**.

**MM 18 (EXPER):** I originally have expertise in computer linguistics.

**TH 3 (EXPER):** I am not an Engineer but I am from humanistic side of education field. For example, User interface studying, information architecture, etc. are my Background

**TuH 6 (RO):** There is a national project named Digitransit. I am a member of the development team and represent HSL in the project. I am system architect of this project.

**MS 11 (RO):** I am someone like an **innovator**. I like a place where I can sketch up some ideas and get comments and feedback from others. I like to be part of a community, so I think Digipalveluohjelma is an ideal place for me.

**MM 20 (EXPER):** With my linguistic works, I did some scripting in programming languages like Python, Perl, etc. I also have some experience in UNIX tools (e.g. text editing)

**JM 52 (RO):** The managers (Director General) of various departments within the Ministry will also be using this dashboard.

**TuH 4 (EXPER):** I **studied computer science**. I am specialized in SW architecture.

**MM 19 (EXPER):** I have also learned some **development platforms and coding skills**.

**VK 12 (EXPER):** I have the technical expertise of Automation and Software Production, as well as industrial engineering.

## AFFINITY DIAGRAM A-2: RESPONSIBILITIES OF PEOPLE

### Responsibilities of People

**MM 71 (RE):** I have to compose report to let others know about project status.

**JM 6 (RE):** Now I make sure that companies working under me come out quickly with some system that is not finished yet, but can be testable.

**MM 5 (RE):** I am also responsible for making contracts of different projects and services.

**JM 53 (RE):** Director Generals don't see progress of a project like a product owner. They like to see an overview of all that services that we have now.

**MM 13 (RE):** The people from ministry are actively working in the development of the projects as well. (e.g. Marketing). They create awareness and visit different places to talk about it. That's their role.

**JM 55 (RE):** The Director General, most likely will not check the details of a project. If she finds some indications to be concerned of, She is going to throw an email to me or call me to know what's going on.

**JM 7 (RE):** The citizens test the ideas.

**MM 15 (RE):** I am, in project point of view, leading 7 teams right at this moment.

**JM 56 (RE):** As a PO, I also start from the overview of project progress information. Then if I see something wrong, then I go to the service. Then I see at the details. If I find out some sort of indication, I go to check at GitHub to see what is going on there.

**TH 4 (RE):** In Digipalvelutehdas I mirror the ideas about project management initiated by Jarkko Moilanen.

**MM 54 (RE):** In the small projects containing 1-2 developers, the lead developer does the fine grained estimation of time.

**VK 6 (RE):** Practically I am responsible to maintain the holistic model of Demola implementation.

**JM 57 (RE):** If it is a high priority project, (e.g. database containing student information in the entire Finland) and that project shows yellow or red alert, the director general will immediately call me to meet her in order to explain what's going on.

**MM 77 (RE):** I have to send report to someone working in the ministry. He in turns needs to report back to his superior.

**VK 2 (RE):** I take care of the implementation of Demola model here at Tampere.

**VK 3 (RE):** My core responsibility is to facilitate Demola projects in three different university campuses (TUT, UTA and TAMK).

**TuH 1 (RE):** I am really not associated with any responsibility with Digipalvelutehdas. I was following the APIKA project.

**VK 9 (RE):** We also compare our experiences as facilitators from different projects.

**MM 77 (RE):** I have to send report to someone working in the ministry. He in turns needs to report back to his superior.

**VK 10 (RE):** We ensure that the connection between project partners and the teams is built and becomes solid. We also make sure both the team and the project partner are active. We also make sure that the teams are doing good and following the Demola model in their work process.

**VK 16 (RE):** It should not be like the project is thrown at Demola and the meetings will take place in every three months. The workload is not big for company representatives but they need to be available and willing to see how things are proceeding in every couple of weeks.

**JM 59 (RE):** The Product owners can decide what information about a service should be accessible to companies. This is up to the project owner to decide.

**MM 4 (RE):** I am also more or less active in substance of certain projects. I am not doing the strict development (coding) work, but facilitating things in discussions and meetings; creating the big picture I'd say.

**TuH 2 (RE):** I work in Helsinki Region Transport HSL. We do various open projects.

**VK 4 (RE):** I also plan for new projects in collaboration with our project partners.

**VK 5 (RE):** My side responsibilities are taking care of the facilitator team, a five member one. I also lead activities like marketing; promotion; improving our facilitation process, campaign model and the concept itself, and so on.

**TH 2 (RE):** My main of work is, for example, right now I am doing development of information gathering on early childhood education. We are doing a project that within three years we'd try to gather existing information from digital systems like registry to one central database

**MS 6 (RE):** I try to connect the right people to solve problems that elderly people face in their lives. One example can be having a central event management system where different organizations can post upcoming events instead of company homepage. Usually in company homepages, people don't check the event list frequently.



## AFFINITY DIAGRAM A-3: CURRENT WORK PROCESS OF SERVICE DEVELOPMENT MONITORING

### Current Work

**VK 8 (CW):** Facilitator can take care of roughly 5-10 projects depending on the project contexts. The number of projects depends on some calculation.

**VK 17 (CW):** Demola projects always have a time period of 3-4 months. This is not only to keep a short period of time with very intensive work, but also so that process go onwards for quite sometimes in order for the teams to have enough time to implement some minimal viable products; having discussion with the end users running several iterations - in order to change and pivot the initial ideas.

**VK 49 (CW):** We discuss about project situations, because the results vary pretty much.

**VK 11 (CW):** We try not to be too much intrusive towards the team. However, we wish the teams to commit to the projects when they come to Demola and the facilitators make sure that teams are well aware of this and give further explanation if necessary.

**VK 24 (CW):** I am not that much concerned with the formalities and bureaucracy when dealing with public sector projects. We need the decisions made by specific committee of people responsible for this.

**VK 20 (CW):** We also have some projects that don't require technical solutions. In such cases we require that the team doesn't come up with only a set of features. It is important that team is proposing a concept and how it should be arranged.

**VK 22 (CW):** We are neutral about the source of the projects. However, if there is only one individual working as the project partner, then we need to discuss this thoroughly.

**VK 41 (CW):** In Demola, we want to be flexible with the way team works. So we can't force to use certain repositories for the teams to use

**VK 44 (CW):** From Demola POV, we don't have any clear set of project managements or Instant Messaging tools. They change more often and we prefer to see what teams are choosing.

**VK 51 (CW):** In our everyday life, we have discussions with facilitators about project status. Also the experience the facilitators accumulate.

**VK 53 (CW):** For monitoring purpose, we have some documents and processes to follow. But they are not based on some high technology, but on our own concept. The facilitator groups surveys some factor weekly. Also we expect the teams to update their work now and then.

**JM 8 (CW):** We try to build up any ideas of a digital system that come out from the citizens.

**JM 11 (CW):** We collect enough information about problems and then try to generate idea of a solution, business model associated for it, approximate cost to implement it, its sustainability, etc. Lastly, we decide are we going to fund building such mockup service.

**JM 13 (CW):** If companies see that there is a potential service that can be implemented, they can start it immediately. They don't need any kind of permission.

**JM 4 (CW):** I prepared the Digipalvelutehdas project with the idea of developing a portal to monitor development processes and everything else for a few months.

**JM 26 (CW):** I am always looking at the ideas to know what is going on. I want to know what people need actually. If I see something interesting, I would like to join in and express my Interest.

**JM 29 (CW):** After the one day workshop and drafted decisions, I can check our own roadmaps from our department and can decide upon allocating a budget for the relevant idea in questions. I then make a suggestion to my bosses that I want to do this and it fits our needs.

**JM 72 (CW):** There would be some projects which I follow more tightly. I go deeper in the details.

**JM 18 (CW):** pro-openness and pro-transparency are in everything that we do.

**JM 24 (CW):** we include the lawyers in the beginning of the project so that they can start the processing the legal issues immediately.

**JM 10 (CW):** If there isn't any current solution to the proposed problem, then we start to ground-source the ideas, content, user stories, business models. etc.

**JM 27 (CW):** When there are almost a dozen people showing their interest in an idea, we hold a design workshop. It is one day workshop, a part of the processes defined by Digipalvelutehdas. Some of them have taken place at Tampere. Here we try to clarify what is the problem, what are the reasons behind a solution, sustainability, possible number of people using the solution, user stories and business models. The decisions are not finalized immediately, but a draft version of further to-dos is made. The information collected in one day workshop is made available in public.

**JM 32 (CW):** After 90 days, the implementer company provides a link to the service metadata so that everyone can find it. The people testing the system can give their feedback and comments about the implemented service. Based on the feedbacks, our experiences and part of the internal process, we make the decisions of going forward or not.

**MM 17 (CW):** The smaller 1-month long projects are consulting type. We write papers or do some POC works within a short time. (e.g. The project for Finnish Tax Authority has duration of Max 3 months). But it would be finished sooner if we get the results earlier.

**MM 24 (CW):** In public sector project, apart from the financial aspect, we need to consider about the regulations, possible roles of different associated organizations, etc. while choosing a project.

**JM 30 (CW):** We hold a public bidding to hire a company in order to implement a testable system. This is based on the requirements defined in the workshops.

**JM 79 (CW):** I work with startups. They move fast and they fulfill more needs.

**JM 34 (CW):** Digipalvelutehdas stops after validating a service and giving it an approval to proceed further. Further development of that idea is a different process. However, that process contains the same 90 days development cycle as well.

**MM 16 (CW):** Our projects are quite small and we are an 8-person company. Each project consists of 1-2 people. The same people are active in multiple projects.

**JM 33 (CW):** We want to know what people need and want and based on the judgment, we make our decision.

**MM 22 (CW):** We need to follow some formalities regarding to manage public sector projects.

**JM 51 (CW):** When I see something out of ordinary or odd in the development work, I send an email or a message via Slack to the team manager to ask what is going on.

**MM 53 (CW):** I actually do not do the estimation of bug fixing time in coarser level.

**MM 25 (CW):** We also need to consider our own set of expertise while choosing a project. We also need to consider whether or not if we have sufficient knowledge and experience in the application area.

**TH 8 (CW):** In Digitransit project, we provide journey planning web service for people where they can put their departure and destination locations. The system then recommends them bus/tram/train route for them.

**MM 50 (CW):** We write down the estimated amount of work in the agreement.

**TH 18 (CW):** If a project is favorite to me, I'd like to follow it more closely in GitHub. So I'd follow more of the radiators here.

**MM 26 (CW):** As we are owned by Finnish Ministry of Education and owner, we need to follow certain processes and regulations. So we have to consider would it be OK by our owner if we wish to work for other ministries.

**MM 27 (CW):** While choosing a project, we also need to consider if we have free resources, their availability time, the estimated work needed, etc.

**TH 7 (CW):** HSL is a public organization. We design the public transport system in the Helsinki region. We design transport system for bus, tram and train

**MM 58 (CW):** I mainly notice how much time has been allocated, how much of it has been spent in terms of man hours.

**MM 53 (CW):** We have a very light process about reporting. We send emails where the lead developer writes down in bullet points about performed work. Then I find out from Tiima system how much hours were used and how much money was spent.

**TH 19 (CW):** For a less favorite project, I might follow fewer amounts of radiators here.

**MS 12 (CW):** I only monitor public sector projects

**MS 7 (CW):** Also I have started hYTE.fi website. Its Question and Answer site about wellbeing technology (like devices for helping people grab things from floor). thing on table on this photo.  
[http://likioma.fi/wp-content/uploads/apuvalineet\\_poydalla.jpg](http://likioma.fi/wp-content/uploads/apuvalineet_poydalla.jpg)

**MS 9 (CW):** I work in Likioma -project. We don't code anything. We don't have anything else to manage than your employer's calendar.

Likioma project in English: <http://likioma.fi/likioma-in-english/>

**MS 15 (CW):** I monitor the **Graph Tab in GitHub**. This is because if I am interested in a new program or product, **I need to see if the project is still active. It is really easy to see these visualizations from this graph tab.**

## AFFINITY DIAGRAM A-4: CURRENT SITUATION IN SERVICE DEVELOPMENT MONITORING

### Current Situation

**MM 7 (CS):** We have 2 major services that are running and operational.

**MM 8 (CS):** We also have 2-3 bigger projects. The big projects are one year of time.

**MM 9 (CW):** We have a couple of smaller projects. They might have a timeline of one month or so.

**MM 28 (CS):** There are different varieties in the type of project work we do. For example: MPass is more about well-being education and it differs from the usual university projects that we do.

**MM 39 (CS):** We, along with other companies are under the umbrella of CSC, and it gives us the tools for registering and recording hours.

**MM 10 (CS):** The projects are not fully functional as a service, but they are in development phase in current tie.

**MM 11 (CS):** One project is the Mpass project, for Ministry of Education and Culture. There are some people working in the ministry who have certain roles there. And we have certain people from our company. So the collaboration is cross organizational.

**MM 14 (CS):** Quite recently, the Ministry of Education and Culture have a Digitization Team. They have some background on coding. They usually don't do coding, but they have a generalized idea about what's going on.

**JM 5 (CS):** Digipalvelutehdas is a simple process, in which the governmental organizations are forced to produce quick first results of idea of one of the services. Whatever it is, a testable user interface needs to be produced in 90 days.

**JM 13 (CW):** Digipalvelutehdas brings all the public sector organizations, municipalities and the development companies together. It also includes the customer: the citizens, who reside in the central. It is a community system built around the customer needs. It is a framework of processes and tools to do development which actually serves the customer needs. The purpose is to create services to solve citizens' problems and address their needs.

**MM 21 (CS):** Our company is part of the public sector. We are owned by the Finnish Ministry of Education. Almost all our projects are public sectors.

**JM 36 (CS):** At the moment, most of the projects are related to education. But we often have ideas from different aspect.

**JM 21 (CS):** We have different development managers in each of the departments under a ministry. However, other departments in our ministry don't have digital teams like us.

**JM 43 (CS):** One way to collect the feedback we depth is via the workshops. Then we need additional automated ways to follow people's behavior and everything else.

**JM 22 (CS):** I started up a digital team within our department with the permission of the head position in this department. The development team is built with people who are ICT professionals, substance people, and lawyers. We try to bring the right people together to address the customer needs, collect the needed information and start the process immediately.

**JM 28 (CS):** We need to consider business model to make a solution sustainable. We can't grant fund to all services to run for infinity.

**JM 16 (CS):** I have several project managers under me working on developing different digital services.

**JM 14 (CS):** Some of the other ministries have already become part of Digipalvelutehdas. For example: The population Register Center under the Ministry of Finance. They have already done some projects under Digipalvelutehdas. The Ministry of Tax Office is also involved already and really eager to participate. The city of Tampere, Oulu, Turku and Helsinki, some software companies are there already.

**JM 15 (CS):** Digipalvelutehdas is ready to go, but it needs to formal approval and papers from the Ministry of Finance so that it can be announced

**JM 44 (CS):** The teams that are going to be part of **Dipor can use either Jira or GitHub**

**JM 40 (CS):** We are making a qualitative framework for making judgments about the results. This is already in implementation. We already have some criteria for measuring the success/end result. For example, we require some end user survey/feedback collection in every development project. People are asked the value of it.

**JM 32 (CW):** **After 90 days, the implementer company provides a link to the service metadata so that everyone can find it.** The people testing the system can give their feedback and comments about the implemented service. Based on the feedbacks, our experiences and part of the internal process, we make the decisions of going forward or not.

**JM 30 (CW):** We hold a public bidding to hire a company in order to implement a testable system. This is based on the requirements defined in the workshops.

**MM 22 (CW):** We need to follow some formalities regarding to manage public sector projects.

**MM 24 (CW):** In public sector project, apart from the financial aspect, we need to consider about the regulations, possible roles of different associated organizations, etc. while choosing a project.

**JM 34 (CW):** Digipalvelutehdas stops after validating a service and giving it an approval to proceed further. Further development of that idea is a different process. However, that process contains the same 90 days development cycle as well.

**JM 33 (CW):** We want to know what people need and want and based on the judgment, we make our decision.

**JM 72 (CW):** There would be some projects which I follow more tightly. I go deeper in the details.

**JM 51 (CW):** When I see something out of ordinary or odd in the development work, I send an email or a message via Slack to the team manager to ask what is going on.

**JM 58 (CS):** The services that are being built, we have priorities among them.

**JM 60 (CS):** The general overview of each service is public to even normal citizens, but the details are not.

**MM 16 (CW):** Our projects are quite small and we are an 8-person company. Each project consists of 1-2 people. The same people are active in multiple projects.

**JM 79 (CW):** I work with startups. They move fast and they fulfill more needs.

**MM 17 (CW):** The smaller 1-month long projects are consulting type. We write papers or do some POC works within a short time. (e.g. The project for Finnish Tax Authority has duration of Max 3 months). But it would be finished sooner if we get the results earlier.

**VK 14 (CS):** We need to have open eyes and be prepared for elements of surprise regarding to the solution.



**JM 48 (CS):** As communication, I am getting feedback and progress information as monthly reports. I get to know the situations monthly when I get the result as report.

**VK 17 (CS):** The first phase of the projects is really important for us. We need to make sure the project partners are comfortable with how Demola model works, is willing to commit here.

**JM 61 (CS):** The citizens and software companies not part of Digipalvelutehdas can see development of the projects as they are open source. They can access GitHub and obtain the information. This is because public money is being used for developing the services. They can go to see issues about development works and what discussion is going on there.

**TuH 36 (CS):** I don't have to make any reports for showing project status to any client

**JM 88 (CS):** Right now the Kanban wall has: 1. The idea phase - just idea with post it notes; 2. Design Phase: where each idea needs to have some specific fields filled in an A4 paper; 3. Implementation Phase; 4. Maintenance Phase and 5. Retirement Phase. Right now the post it notes goes to different phases as the services evolves. But we want to get rid of this wall and represent it in a digital way.

**VK 13 (CS):** One of the basic characteristics of our projects is that the starting point, the solution or the results is not clear yet. We wish to have the needs and starting point clear. But if the initial solution is already quite clear, then it is all determining the starting points.

**MS 23 (CS):** I don't have enough time to monitor other projects right now except Hyte project.

**TuH 9 (CS):** The data and API provided and developed by HSL are of open source.

**VK 56 (CS):** The problems and challenges I face are related to specific projects, not general.

**VK 52 (CS):** There are teams which are following the processes but are not performing well. Also there are teams who are not going according to the process, but still are producing good results.

**TH 5 (CS):** I have not taken part in the one day workshops of evaluating ideas but I am familiar with the procedure.

**TuH 14 (CS):** We have Agile two weeks sprints and we follow a burndown chart to know how the progress is going. This is something missing in APIKA project.

**VK 61 (CS):** Teams keep blogs where they update their statuses weekly. So this is one way to see in which directions the projects are going on. There is no structure about the blog itself. Teams can use Demola provided simple blogging tools or something of their choice.

**TH 13 (CS):** In the weekly meetings with give out a memo about the meeting and that contains some information about the progress of different projects.

**TH 7 (CS):** Some projects are of longer durations with results of vast dimension. Some of them are shorter in time frame with small results. Also the longer projects, we try to cut them out in 90 days.

**TH 6 (CS):** We have the weekly meetings to know about progress of ongoing projects under Digipalvelutehdas. One of the projects is the Dipor Dashboard. Another project is the information gathering on Early Childhood. I think we have right now five projects ongoing at the moment

**TA 8 (CS):** A skilled coder with naturally be able to give more time. However, others might not be an expert, so they won't be able to give that much input based on their work skills and personal obligations.

**TH 16 (CS):** We used to visit GitHub around 6 months ago more frequently. But most of the time GitHub is run by the companies developing projects. They are mainly in charge of monitoring the GitHub.

**TuH 24 (CS):** For Digipalvelutehdas, the project sizes are not so relevant because they are simpler and of three months' duration. They are simpler than projects of 2 years.

**TH 12 (CS):** We don't get very detailed information about some particular project right at this moment. The person in charge of the management of specific project keeps us up to date about project progress. The updates come in weekly basis.

## AFFINITY DIAGRAM A-5: CHALLENGES AND LIMITATIONS

### Challenges & Limitations

**VK 19 (L):** The technical solutions might not be feasible if the concepts and values are not thought beforehand. If the teams rush into building technical solutions, there are often risks that the solution might not address the needs properly.

**VK 50 (L):** Often it is hard to make comparison because projects differ from one another by nature and requirement.

**VK59 (L):** One problem is that from time to time we need to emphasize about following the processes and documents to everyone.

**MM 35 (L):** From the Demola POV, if we have a defined task list at the beginning which might need to be changed somehow and **if we focus too much into measuring task breakdowns, then it leads to the fact that the development is done as waterfall model.** That you are not willing to change it so much when it might be necessary.

**VK 25 (L):** Sometimes formalities and bureaucracy are obstacles in the way of starting a project.

**VK 56 (L):** The problems and challenges I face are related to specific projects, not general.

**JM 5 (L):** Previously all the development projects used to be three years long. And when they are released, they have poor usability feedback.

**VK 26 (L):** In Demola we talk about a 3 month long, innovative, part-time project work with low barriers. So if it is stuck with decision making processes at the very beginning, the results at the end may be disappointing as it takes a lot of time and effort to get started.

**VK 44 (L):** From Demola POV, we don't have any clear set of project managements or Instant Messaging tools. They change more often and we prefer to see what teams are choosing.

**VK 31 (L):** If we are making too much measurement of one specific aspect, then it makes limitations to know the actual progress. We get what we measure.

**JM 17 (L):** The work process of different projects and the transparency of the process is not enough for me.

**JM 23 (L):** Law is the biggest obstacle for everything. The Law making process takes almost a year.

**JM 84 (L):** I am **unable to have real time overview of the project progress/status**

**JM 41 (L):** One of the problems we have is the collecting feedbacks or indication of feedback automatically. Because people usually don't have separate time to give their feedback. Pushing out surveys or trying to force people to answer them via email is not effective.

**JM 64 (L):** The border about what is visible and not visible is a bit blurry at the moment. We need to define about how much information should be limited from public access.

**JM 76 (L):** The existing software or applications for visualizing software development progress provide almost all features that we actually need. But they are not out of the box. Often, we need to configure the tools/systems so that they can address our needs.

**JM 80 (L):** The usual charts and other visualizations, they are not attractive and interesting enough for most citizens. They find the information presented difficult to understand. So we need new ways of visualizations to express these project related information to mass public.

**JM 89 (L):** Right now I need to go at different places to obtain project related data. I have to go to GitHub, I have to go to Jira. I want all of these integrated at one place.

**MM 23 (L):** One limitation in public sector is formulation of laws and regulations. This is very slow. And it turns its delays the development works as well.

**JM 83 (L):** The problems with current solutions in the market are that they are designed for engineers, not for ordinary people, not for managers who give little importance to technical details. This is the primary reason why current solutions fit to our needs.

**JM 77 (L):** The existing software/systems for visualization contain too much features. It makes the system are complex and hard to configure and modify.

**MM 42 (L):** In past we have used Jira, but to me it was quite complex. I don't use it anymore.

**MM 52 (L):** In the resource allocation scheme, of course we need to be prepared for unexpected problems.

**MM 49 (L):** We don't have any specific tool to come up with a figure about the time needed to fix bugs/issues. This is done by the experience of the managers and experts. They discuss and make estimates.

**MM 62 (L):** I spend quite a long time in getting report from Tiima.

**MM 82 (L):** With Tiima, I have to login, click several pages to enter associate information. It is really complex and takes a lot of time. It doesn't support on spot reporting about work hours.

**JM 74 (L):** The problem with reporting I face is that I need to report about the hours and money spent the way the sponsor wishes to see. You end up reporting in several ways.

**MM 79 (L):** Everyone needing to log hours in Tiima complains that it is very slow and cumbersome. The tool needs a lot of time to start. We need to make a lot of clicks.

**MM 78 (L):** The reports contain very simple texts. **They include raw data.**

**MM 69 (L):** Nowadays, finding out this trivial information requires way too much works. We need to dig down several systems and ask people. It is not durable. It takes too much time from our busy schedule.

**MM 85 (L):** Right now, we don't have any tool to optimize who does what. Also we need to identify the appropriate people person for conducting a job optimally.

**MM 75 (L):** Sometimes you need to report about the same work in two different formats which ends up with a lot of hand-work.

**MM 63 (L):** Also I need to spend some time in making excel entries as well to find out things I require.

**MM 83 (L):** We now put our hour entries at the end of the month because Tiima is very slow and time consuming. So throughout the whole month, information is not updated regularly.

**TuH 40 (L):** **The main limitation of monitoring project is lack of time. There are so many open source projects. So it is difficult to follow everything interesting.**

**TuH 26 (L):** I have used Google analytics very little. I know it is comparable; it has more features and more polished. But we wanted to use something preferably open source, preferably something not in United States. Because we have the problem that our users use the service very often and they are often on move. So the user information is private and personal. **We don't want the travel information to go in any other system except ours.** And we wanted to stay within Finland, maximum within the EU. So for privacy concern we could not use Google analytics.

**TuH 27 (L):** **When you use Google analytics, all your user information will go to Google.**

**TuH 44 (L):** One **problem with Jira is milestones can't be made public as well.**

**TuH 15 (L):** For APIKA project, I followed **GitHub issues, what bugs have been reported, etc. But there I did not have a good view on the progress. I did not know if the listed bugs were fixed, how many of them were fixed, how quickly they were fixed, etc. And I didn't have any visualization for that.**

**TuH 22 (L):** **We tried to use GitHub for organizing sprints, but it was really difficult.**

**TuH 42 (L):** **Jira often feels really complicated.** There are too many things. It is good to be adaptive to a lot of projects, different type or organizations with different features. But I need to configure the system to see only relevant information that suits me.

**TuH 23 (L):** We also used **waffle.io** where you can have a sprint board of your GitHub issues. But it was **not good enough when our projects became big.** We were working here for more than a year

**VK 61 (L):** Teams keep blogs where they update their statuses weekly. So this is one way to see in which directions the projects are going on. There is no structure about the blog itself. Teams can use Demola provided simple blogging tools or something of their choice.

**TH 9 (L):** Law is our biggest concern about starting up a project

**TA 9 (L):** It's difficult to say how the personal hours contributing to the course is being developed or implemented.

**TuuH 43 (L):** One **big problem we have with Jira** is that, we **can't make our project public** so that everyone could see it. People can see only partial information. E.g. you can see individual issues and comments. But you can't see the burndown chart. Jira won't allow us to make it public

**TuH 45 (L):** **Jira requires payment for making user accounts. So if information is public there will be little or no need for paying for multiple user accounts.** That's why Jira doesn't allow information to be public. We have many customers and it is not possible for us to make individual accounts for all of them.

**TH 17 (L):** The information shown in **GitHub graphs doesn't give me enough details.** For example if I am outside a project, then I don't know how individual members are contributing in GitHub, are their level of input is satisfactory or is it lacking something.

**TA 2 (L):** Every end user may have several opinions about system interface. It is difficult to take into account of every opinion when a system is developed.

**TA 4 (L):** In Demola they have blog entries about progress, but they are difficult to read and understand. Blogs should contain a min. Requirements of contents. People should be able to understand what their blog is about when they read it in later times.

**TA 11 (L):** **stacked bar gives the worst kind of readability**

**MS 1 (L):** If I use some data I need but there is no visualization available to show it, this would be problematic.

**TH 20 (L):** If I am within a project, I can know who has what skills and how much contribution would be made by the members and that would be recorded in the agreement. If I am outside, I really don't know about this information on people.

**TA 10 (L):** **Pie chart is difficult to understand.** Perhaps it misses the exact explanation about what are the big/small chunk stands for and how much of the entire amount do they represent.

**MS 25 (L):** **If it takes more than 2 clicks to get some information, it is problematic.**

## AFFINITY DIAGRAM A-6: TECHNOLOGY

### Technology

**MM 29 (T):** We are using the same kind of technology for developing both the university projects and well-being project.

**MM 30 (T):** In more depth, there are differences between technical solutions of the project works, but they are still based on the same expertise set and technology protocols.

**MM 41 (T):** We used GitHub for publishing the code for MPass. It was set to be mandatory by the Ministry of Education and Culture

**VK 28 (T):** Digital Application for measurement 6Aika

**MM 44 (T):** There is another tool that we use to set targets in every half year. It is called Skills. It is a web based tool. It is within our company intranet.

**MM 56 (T):** I think Jira is being used by some groups.

**MM 57 (T):** When we are working with subcontractors, they may use their own estimation tools and repository.

**MM 65 (T):** Tiima generated reports as excel sheets with raw data. You need to work on your own to create visualizations. This is time consuming.

**MM 43 (T):** Now I use Trello. That was also a choice for the Ministry of education. We don't use in in a very complicated way. It is used as a board of To-Do Lists.

**MM 46 (T):** The real life usage of time is logged in Tiima. (e.g. if a person spends hours in fixing up a bug, s/he puts it in Tiima.)

**MM 40 (T):** The tool that we use is Tiima, developed by the Finnish company Visma. We use it for registering and reporting the hours spent. www.tiima.com

**MM 51 (T):** We also use an internal tool within our company based on excel. This is done to allocate resources (person) and the needed time by that resource to work in that project.

**VK 39 (T):** We don't follow any specific data repository because it depends on the project itself. Some projects use GitHub and Jira.



**VK 43 (T):** It depends on the working team to decide their choice of repository. Usually the team has prior experience on using a specific repository, so they use it during the project work.

**TuH 16 (T):** GetDeckhub looked interesting to me. It is a desktop client for GitHub. I'd probably take a look in it. It lets know what is happening in different GitHub Projects.

**VK 45 (T):** If a team is confused about using some repository systems or other services, then we provide some help and guidance about what to use; provide some ideas from other teams.

**TuH 21 (T):** In Jira, in some way there are similarities like GitHub. You can have graphs. Also there are sprints.

**TuH 13 (T):** In Digitransit project, we use JIRA

**TuH 25 (T):** We have web analytics. We use Piwik.

## AFFINITY DIAGRAM A-7: INFORMATION RADIATOR

### Information Radiator

**JM 67 (ME):** There should be combinations of indicators, where you can calculate some exact numbers. For example: how many issues are reported, what are their status, how many are not fixable, etc.

**JM 69 (ME):** Also another indicator is the backlog: how many items are in it; how many items will be implemented and how many are already in progress. If no. of backlogs to be implemented is divided by the no. of hours needed to implement them, we get a number and that is interesting to me.

**MM 33 (ME):** We agreed on projects about its duration. For example: 3-person-month. We look at the tool entries to determine how much of the time has been spent and how much is left. Based on that, we decide if something needs to be done and react.

**JM 68 (ME):** Also another consideration can be the development work progress vs. no of issues reported and their status.

**MM 37 (ME):** We consider availability of resources (e.g. unexpected leaves, paid leaves, amount of holidays, etc.) while calculating person-month hours.

**MM 36 (ME):** The basic unit that we register and report is an hour.

**MM 32 (ME):** The most important aspect of every project is “Persons-hours-done”- work.

**MM 34 (ME):** We calculate a person-month as certain number of hours. It doesn't depend on weekdays or weekends. (e.g. when you have done 157 hours, it means 1-person-month).

**MM 35 (ME):** We agree on the company level to determine how many hours should be considered 1-person-month.

**MM 38 (ME):** We need to consider different calculation for yearly timeline and monthly timeline when calculating person-month. This are ruled given to us by our owner and other companies.

**MM 45 (ME):** We need to distinguish between the allocated time and the time used in real life. This needs to be done in order to address software maintenance work and fixing discovered bugs.

**MM 47 (ME):** The allocation of time is done in agreement. Then we need to estimate about how much work will be needed to do.

**MM 48 (ME):** When we are estimating the work to be done within allocated time, we need to consider the risk of big bugs.

**TuH 17 (ME):** For me it is important to see number of recorded issues vs. number of solved issues in GitHub.

**MM 72 (ME):** For Mpass project, we need to provide report to the ministry of education every month. We need to mention about the amount of money spent, amount of work man hour spent and what was achieved. The report templates are given by the ministry.

**VK 29 (ME):** Most of the radiators make sense to me: [Planned Development Hours Vs. Spent Development Hours; Man Hours, Hour log sheets; Allotted Time Vs. Spent Time; Invested Money Vs. Spent Money; No. of User Stories Vs. Implemented Stories; State of a Task Breakdown (Not Started - In Progress - Ready for Testing - Done - Delivered - Start of Usage); Number of reported Issues vs. Number of solved Issues; Milestones; Team Size and Skills]

**MM 59 (ME):** For working with subcontractors, I also need to monitor about used money, not only hours.

**TuH 10 (ME):** When I am following a project to monitor its progress, I want to know how many people are participating. When a project gets bigger, I know somehow it has to be good. I know that since a lot of people are now involved there, so there must be something positive and significant. And I want to know the details.

**TuH 35 (ME):** When the barn down chart is not often fine grain, then it would be good to see additional information about task labels, status and time.

**TH 22 (ME):** It will be really useful information if I get to know how issues are changing their status over time to determine development condition of the pro-

**TuH 20 (ME):** Milestones will always be interesting.

**TuH 34 (ME):** I think a barn down chart is good for showing project progress.

**MM 60 (ME):** In most of the projects, man-hours is the most important measurement.

**VK 30 (ME):** We have some specific ways to measure the Demola Performance.

**VK 40 (ME):** If it is a software related project, it is good idea to somehow build the indicators within. For example: repository being passive. So you can get a timeline, and you can see where a lot of activity has happened and where the amount of activity has been little in certain times. This might be a signal to project status. It may not be problematic, but something worthy to have a discussion about.

**TuH 33 (ME):** If I can point out state of development tasks with time and how they are connected with different user stories, I can know about project progress

**TH 24 (ME):** If ratio of open issue vs. closed issues is shown, they may somehow indicate progress status of development.

**TH 11 (ME):** I like to get notifications when new projects are stated and when it goes to a new phase. Let's say at first, this is just an idea and maybe I like the idea from the beginning and wish to follow it. Or maybe I don't understand the concept behind the idea, so I wait for later times. Then when I hear that there is this prototype of that idea and then I want to know about it again. If the prototype looks interesting to me, then I might start following the project.

**TH 32 (ME):** I think I would get an overall idea about how the project progress, if the development is going in a good way or negative way if I see **user stories and their completion based on number of issues created and closed along with different issue labels/severity.**

**TH 27 (ME):** If we can **associate Milestones with issues** that will be useful because then we would know what issues are people working on and if they are part of current Milestones. But this is somehow secondary information.

**TH 25 (ME):** **Planned development hours vs. spent development hours is also a good indication.** Because we allocate our budget based on how much hours would be allocated for the work. Within the spent hours, if we could see the number of fixed issues vs. number of open issues that can indicate how the project is going on, if there is some problem going on.

**TA 7 (ME):** there should be a **minimum no. of hours spent in a week** to get a predetermined project work hour.

**TA 6 (ME):** if somehow **total working hour of people** is shown.

**MS 4 (ME):** I am not so interested in commits in GitHub. I look things from end user POV and issues are most important to me

**MS 17 (ME):** I have not seen the options of "Traffic" or cloning in GitHub. I have only noticed "**Contributors**"

**TH 26 (ME):** Suppose we have allocated **20 hours** and a minimum number of **15 issues to be solved within this time.** If we see that **15 hours has passed and only two issues have been fixed, we can determine how the work is going on, how much contribution has been made, what is the situation,** etc.

**MS 16 (ME):** I only noticed the commits in GitHub. I didn't know that there is option to see addition and deletion in GitHub. I only need the active information.

**MS 21 (ME):** The **Issues** are also a good thing. Issue types also give good information.

**MS 20 (ME):** Its really good to see how active the project is (**commits in GitHub**)

## AFFINITY DIAGRAM A-8: VISUALIZATIONS

### Visualizations

**JM 45 (VIZ):** GitHub is now the easiest way to get started. So it the primary of source of getting information to know what is going on and to make some sort of visualization charts out of it.

**JM 71 (VIZ):** The **traffic light sign** interests me as well. In the morning when I open the dashboard, I see the overview of my day. Do I need to be busy with something or not, can I concentrate on invasion.

**MM 66 (VIZ):** I'd be really happy with **very simple visualizations**, let's say bar charts where I can see what has been planned and what has happened in real life.

**MM 67 (VIZ):** I'd like to see Information like amount of work, travel, other expenses, etc. as visualization.

**MM 68 (VIZ):** It would also be great to have possibility to drill down the depth of the information. For example, if you have a large expenditure in travelling, you should be able to

**VK 54 (VIZ):** For visualizations, the traffic light approach might be appropriate here, especially with repositories. If there are indicators like passivity, or activeness over a time - **traffic lights, weather forecast would be intuitive** to indicate if everything is going well or not. Especially **when we need to handle a good number of projects.**

**MS 24 (VIZ):** I like all of the visualizations that you have shown. Especially **cards and bar charts are simple and easier to understand.** Nothing extraordinary, but the view is really clear.

**JM 70 (VIZ):** The visualization of software progress depends on what I am looking.

**MM 84 (VIZ):** We want to see the **real time situation about resources being used.**

**JM 73 (VIZ):** I'd probably want to **see timelines**, not a snapshot. I want to see status at different times.

**VK 55 (VIZ):** I want visualizations that provide status as "**at a glance information**" as well as "**detailed information**"

**MS 26 (Viz):** It is ok to have **multiple visualization way to show single information**, but it is simply extra options. It's not something I must require.

**TH 29 (VIZ):** I often think that a **line chart** is good

**TH 30 (VIZ):** Under widgets if you show **spark line** (e.g. under the number mentioned in card, with sparkline how the numbers have been developed) I think it would have been a nice addition.

## AFFINITY DIAGRAM A-9: EXPECTATIONS

### Expectations

**JM 3 (EXPEC):** I think of the idea about developing a portal which gives me situational overview of the projects.

**JM 20 (EXPEC):** Instead the monthly report from my project managers regarding to project developments, I want to **see the progress in real time.**

**JM 25 (EXPEC):** We need the ideas to be available in a public space so that everyone can see them, like them and give their feedback. This is needed to know that if the idea is gathering enough interest or not. An Idea needs to have more approval and people (within Digipalve-lutehdas and outside) for making further process.

**JM 54 (EXPEC):** **Director Generals need to understand in a glance the condition of the project.**

**JM 19 (EXPEC):** We use tax-payers' money and I want to give people chance to see what we are using the money for.

**JM 42 (EXPEC):** We need to have provision of giving feedbacks within the system so that it can be collected frequently.

**JM 31 (EXPEC):** I expect Dipor Dashboard should show both the present situation of ongoing services (POCs) as well as the ideas suggested by citizens.

**JM 49 (EXPEC):** Instead of waiting for a month to get a report, I want to **see the status instantly,** whenever I wish to.

**JM 38 (EXPEC):** The backlogs need to be always visible to everyone: the plans, actions, sprint planning (ideal cases).

**JM 39 (EXPEC):** Everyone should be able to follow what is going to happen in 2 weeks, is it going to happen or not

**JM 46 (EXPEC):** The dashboard also needs to have **link to directly connect to the production environment** of the service to be tested, in order to get feedback from there.

**JM 47 (EXPEC):** As a product owner, I require that teams update about their development information in GitHub. They need to use GitHub Issues.

**JM 37 (EXPEC):** A Software project needs to have a requirement management/ issue management system. It can be GitHub. Jira. etc.

**JM 50 (EXPEC):** I want to reduce the reporting responsibilities of the teams. I want the availability of information to be as much automated as possible, not to wait for some report.

**JM 65 (EXPEC):** I have a strong idea that basically we don't need to keep anything hidden. We are not going to put agreements with companies in the service. It's about development; it's not about management itself.

**JM 66 (EXPEC):** The money is not important for me to measure a development measure criterion. I am only concerned about the fact that companies don't go over the budget in implementing a service. If the expenditure is under the allotted budget and the results are good, I am satisfied.

**JM 78 (EXPEC):** And it's basically easier to develop something specific to our needs than to configure a complex system; to include organizations and its processes which are normally slow and rigid.

**JM 82 (EXPEC):** I require that the services need to have their own APIs. So anyone can build up their customized user interface. Anyone should be able to acquire the data and customize this according to different type of contexts. It needs to be understandable to their target audience. Not just for software engineers.

**JM 74 (EXPEC):** We need some new and different ideas regarding to visualizations and it is the right moment to test something like this. We need to find new ways.

**JM 90 (EXPEC):** The system needs to be flexible. It needs to be a micro service architecture so that I can put a new piece there, use the APIs and collect the information in the view and expand Dipor whenever I need it.

**JM 85 (EXPEC):** I also need to know information about people associated with the projects: who is involved with what work. That's why the team composition is within the requirements.

**JM 86 (EXPEC):** I also need to know how the team composition is being changed time to time

**JM 87 (EXPEC):** It would be great to have a digital Kanban view for the services that we are currently working for.

**MM 61 (EXPEC):** Logging man-hour spent can be done by Tiima, but I wish we had more developed tools.

**JM 81 (EXPEC):** I like the idea of weather forecasting because it is something the normal people would understand.



**MM 64 (EXPEC):** Sometimes monitoring too much time and it would be simpler if I could have web based tool where I can have a glimpse of all this information as graphs or reports.

**MM 76 (EXPEC):** It would be nice to have a tool where you can program in advanced about the report format.

**MM 81 (EXPEC):** It would be really great if I could access the system with my mobile/laptop at any time of the day to log about my work hours. If I do something, I should be able to register right away.

**VK 18 (EXPEC):** We need to have testable and viable concept, depending on the project itself. From Demola POV, we want the team to focus on the concept and values associated with the project.

**MM 70 (EXPEC):** I should be able to find information I need by 2 clicks.

**VK 32 (EXPEC):** We should be careful about what to measure and how to balance the measurements.

**VK 34 (EXPEC):** State of task breakdown combined with number of issues reported vs. no.

**VK 38 (EXPEC):** The easy indicators (for example: hours, invested money, and so on) they should be mentioned but with some disclaimer that you should not put too much focus on them.

**VK 23 (EXPEC):** We want to ensure that when the project starts, there is money reservation for getting license of the project results, the project itself is valuable for the individual person so that he is willing to pay for the license of the produced results.

**VK 27 (EXPEC):** It is important for us that the barriers to start a Demola project should be as low as possible.

**VK 33 (EXPEC):** The measurements should include a balanced combination of different radiators.

**VK 36 (EXPEC):** Regarding to no. of reported Issues vs. no. of solved issues, if an issue about “not going to the right direction” or “not delivering good value on the point when it becomes visible” - then this type of issues should also be pivoted and given significance as well. This makes much sense to me

**VK 47 (EXPEC):** Making comparison between projects would be done more if we are provided with some tools or systems for that.

**VK 42 (EXPEC):** If everyone is using some sort of repositories and a systematic way can be built to monitor the status, it would make much sense to me.

**VK 58 (EXPEC):** The first phases of the projects are really important for us from the prospect of the teams as well. It is important for the teams to be clear about the mandatory parts of Demola projects. There is a lot of information, events nowadays. We want the teams to commit to specific process, getting to know each other, having a good starting.

# APPENDIX B: LOW FIDELITY PROTOTYPES AND USER FEEDBACK FROM PROTOTYPE EVALUATION

## Prototypes based on Sampo Software Oy's Defined Requirements

Figure 0-2 Organization view in Dipor Dashboard

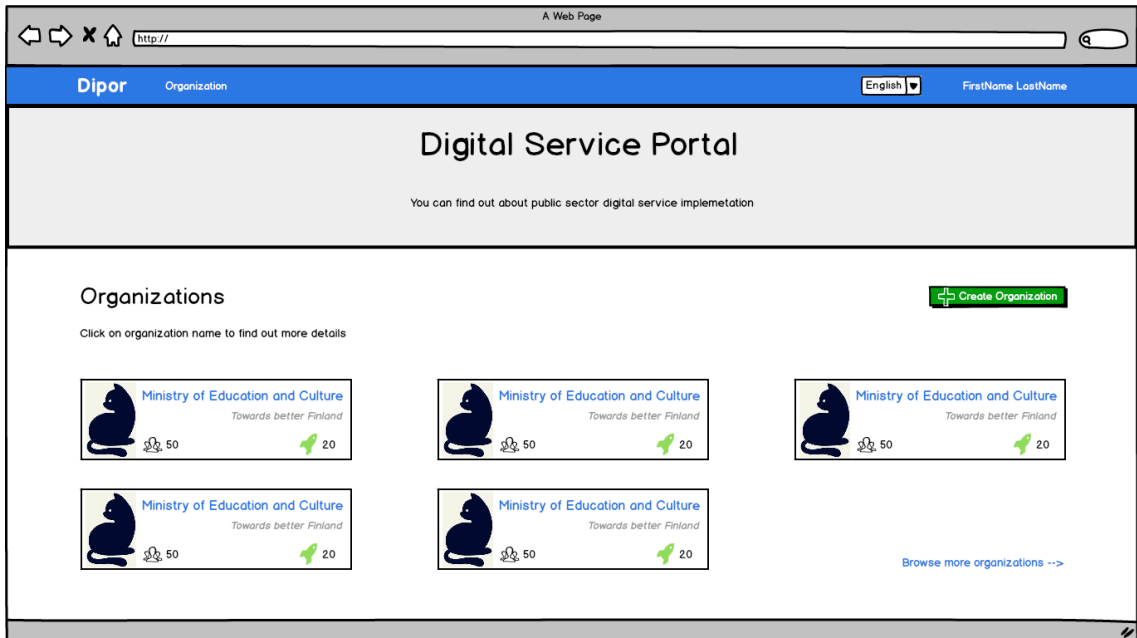


Figure 0-3 Organization profile in Dipor Dashboard

The screenshot displays the Dipor dashboard for the Ministry of Education and Culture. The interface includes a navigation bar at the top with the Dipor logo, the word "Organization", a language dropdown set to "English", and a user profile icon labeled "FirstName LastName". Below the navigation bar, the breadcrumb "Organizations > Ministry of Education and Culture" is visible. The main content area features the organization's logo (a blue cat) and name "Ministry of Education and Culture" with the tagline "Towards a better Finland". To the right of the logo are "Settings" and "Quick Add" buttons, with a dropdown menu for "Quick Add" showing "Department Service".

The dashboard is divided into several sections:

- Overview:** A summary box containing three metrics: 5 Departments, 20 Public Sector Digital Services (with 10 Ongoing and 10 in Production), and 50 Members.
- Admins:** A table listing administrators with columns for a profile icon, "FirstName LastName", "Designation", and an email icon.
- Departments (5):** A list of departments. The first entry is "Department of Basic and Early Education" with the tagline "Ensuring happy learning" and icons for 10 members and 10 ongoing services. Below it are three placeholder entries for "Department Name".

Figure 0-4 Department view in Dipor Dashboard

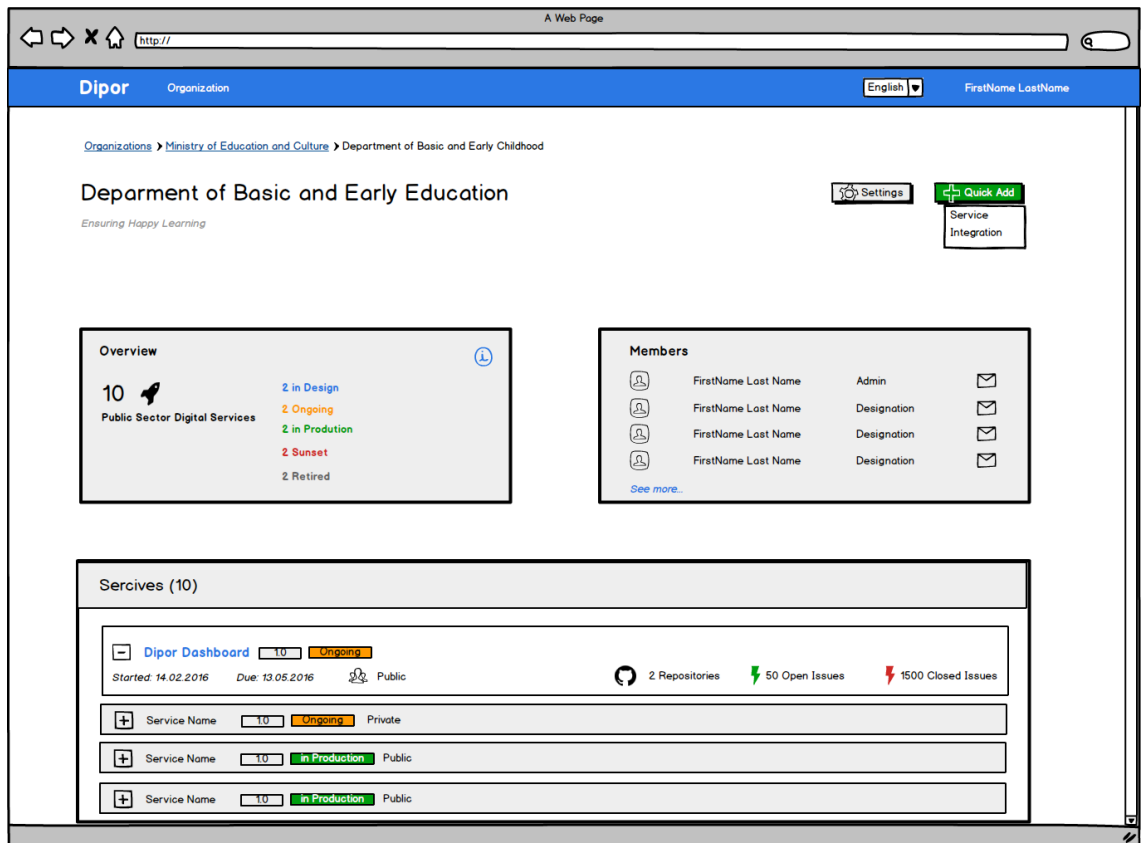
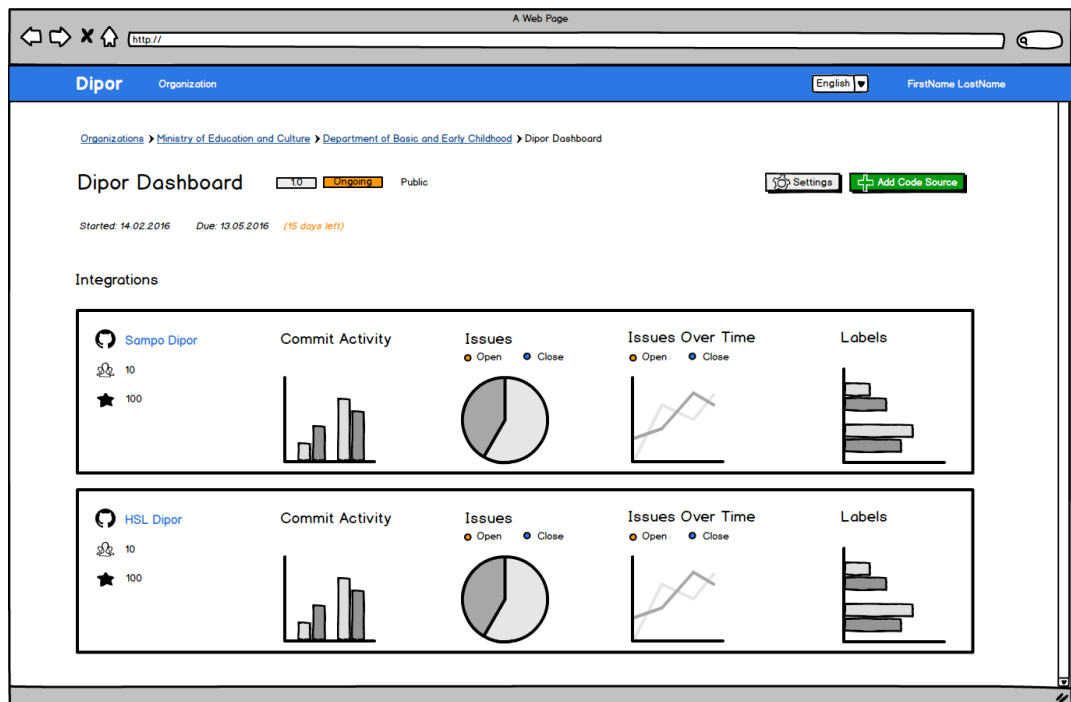


Figure 0-5 Service Integration View in Dipor Dashboard



## User Feedback from Low Fidelity Prototype Evaluation in User Interviews

Session	Feedback on Design Sketches from Sampo Software Oy	Feedback on Dashboard Sketches for Concept Development
User Interview 3	<p><b>Positive Findings:</b></p> <ul style="list-style-type: none"> <li>• I like the information hierarchy in the site.</li> <li>• The Quick Add button looks helpful</li> <li>• It's good to see that a service can add multiple repositories in it.</li> </ul> <p><b>Improvement Suggestions:</b></p> <ul style="list-style-type: none"> <li>• There should be some difference from home screen and organization catalog. Perhaps separate catalog?</li> <li>• The visualizations look a bit smaller to me. They might be good for comparison, but would I be able to interact with them?</li> </ul>	<p><b>Positive Findings:</b></p> <ul style="list-style-type: none"> <li>• "It's totally different from the 1<sup>st</sup> version. But I liked the idea of having my own home screen."</li> <li>• "It's good to see I can access my services directly from home screen. I don't have to go through several clicks to access them"</li> <li>• "Having a separate page for the repository looks well. I can see more information here."</li> </ul> <p><b>Improvements Suggestions:</b></p> <ul style="list-style-type: none"> <li>• How about the option of adding the GitHub Repository itself along with adding a service from Dipor?</li> <li>• I want to see all attributes available from GitHub Repository</li> <li>• Is there any ways to see open and closed issues on a bigger view against time?</li> </ul>
User Interview 4	<p><b>Positive Findings:</b></p> <ul style="list-style-type: none"> <li>• The hierarchy actually maintains how the ministry and departments actually function. I like this.</li> <li>• I like the idea that members of a ministry or department are visible</li> <li>• The dashboard looks good for comparing service development.</li> </ul> <p><b>Improvement Suggestions:</b></p> <ul style="list-style-type: none"> <li>• Can the organization card show information about number of followers and services?</li> <li>• I'd like to see some organization / department specific overview in their profile page. For Organizations, this could be number of On-going and In production services. For department services in different phases would be beneficial.</li> <li>• Associate the service pages with life cycle phases.</li> <li>• Only visualizations are not enough for me, I'd like to see numeric values to show the sum of an attribute and visualizations should show the change trend.</li> <li>• How I can determine if a specific integration needs my attention? Is it system configured or I can customize myself?</li> </ul>	<p><b>Positive Findings:</b></p> <ul style="list-style-type: none"> <li>• "It's good to see the Dashboard. This works perfectly if I want to compare two separate development works."</li> <li>• "It's good to see the option of separate catalogs for Organization and Services. It would be easier to access them than going through deep navigation."</li> <li>• I like the idea of keeping the completion date for a service and showing notification when the deadline is near.</li> <li>• I like the details view for a repository.</li> </ul> <p><b>Improvement Suggestions:</b></p> <ul style="list-style-type: none"> <li>• Can I get some way to identify which services need my attentions? Like if I have more opened issue than closed ones, I need to talk with the development team.</li> <li>• Favorite organization on home page</li> <li>• Life cycle phases on the services</li> <li>• How is the data being filtered? For week or month?</li> <li>• Statistics on development work for 30 days basis</li> </ul>

<p><b>User Interview 5</b></p>	<p><b>Positive Findings</b></p> <ul style="list-style-type: none"> <li>• General structure of the website looks simple and easy to understand.</li> <li>• Like the idea of adding GitHub repositories in the integration view.</li> </ul> <p><b>Improvement Suggestions</b></p> <ul style="list-style-type: none"> <li>• I'd like to have direct link to service pages. Would decrease number of clicks.</li> <li>• The visualizations appear too small to interact. If they are static, it is ok. But for filtering, seeing data, they should be bigger.</li> <li>• Is there any way to filter the data for specific time interval? Like last 24 hours, last 7 days?</li> <li>• It would be good to know how the attributes have changed over a specific time period. In texts. From visualization it is often difficult to interpret.</li> <li>• Can I customize what attributes I want to appear in the integration view? If not for individual repositories, combined customization will be ok</li> </ul>	<p><b>Positive Findings:</b></p> <ul style="list-style-type: none"> <li>• Good to see a separate Dashboard and details view for individual service</li> <li>• Keeping GitHub repository link is good</li> <li>• The idea of setting thresholds to identify problematic services is interesting. This can have some more ideation and improvement</li> </ul> <p><b>Improvement Suggestions:</b></p> <ul style="list-style-type: none"> <li>• Consider limiting number of visualizations appearing on dashboard view?</li> <li>• Can we set some default visualizations and threshold values for a first time user? S/he can customize it later.</li> <li>• Any means to show burn-down/burn-down chart? Kanban board?</li> <li>• Details on commits for individual contributors</li> </ul>
<p><b>User Interview 6 (Marko)</b></p>	<p><b>Positive Finding</b></p> <ul style="list-style-type: none"> <li>• I like the idea of monitoring service development within organizations and departments.</li> <li>• The overview statistics for organization and department would be helpful for at-a-glance information.</li> <li>• I like that charts are of different type. Gives me some variation in the UI.</li> </ul> <p><b>Improvement Suggestions</b></p> <ul style="list-style-type: none"> <li>• Can I see completion days for a service? If I have my own, some alert when the date approaches would be helpful.</li> </ul>	<p><b>Positive Findings:</b></p> <ul style="list-style-type: none"> <li>• "This is totally different from the previous idea. I like the simple UI style very much."</li> <li>• A dashboard having all my services and a separate detailed view for each service looks good. I can check the separate view if I need more information.</li> <li>• It's good to see the dashboard lets me know if I need to pay more attention to a service. Setting thresholds for this is something new for me, but I think I can learn it quickly.</li> </ul> <p><b>Improvement Suggestions</b></p> <ul style="list-style-type: none"> <li>• Is there any way I can select my preferred chart types? Sometimes even number is enough for me.</li> </ul>

## **APPENDIX C: FORMS USED IN USABILITY TESTING AND TEST TASK TEMPLATE**

### **CONSENT TO RECORD A USABILITY TEST**

I request you to participate in a usability test that is part of my thesis work on **Concept Development and User Experience Measurement of Dipor Dashboard** at the Tampere University of Technology. By participating in the usability test you will help us to evaluate the usability of the service to monitor ongoing digital public services.

You will be asked to perform different tasks using the service and to think out loud while doing the tasks. In addition, I will request you to fill in questionnaires and I will interview you about the use of the service. The test will be recorded.

During the test, I will record the computer screen and its events, a video image of your face, and audio. The materials recorded during the test will be used to evaluate the usability of the service in my thesis work. In addition to me, my supervisors for the thesis work will view the video and other materials from the test. The recordings will be destroyed after the work is over.

The results of the test will be reported anonymously.

You can stop participating in the usability test at any point.

**By signing this form, you will accept the above terms.**

Date and place: \_\_\_\_\_

Signature: \_\_\_\_\_

Name clarification: \_\_\_\_\_



## BACKGROUND QUESTIONNAIRE

Age: \_\_\_\_\_

Gender:  Male     Female     Other

### Occupation:

- Entrepreneur
- Employer
- Student
- Service Holder
- Unemployed or in a leave

### Education:

- Comprehensive or elementary school
- High School
- College University Degree
- Else:  
\_\_\_\_\_

What is your field expertise: \_\_\_\_\_

## USE OF COMPUTERS AND KNOWLEDGE IN SOFTWARE DEVELOPMENT PROJECT

### How do you evaluate your computer skills?

- Excellent, I understand how computers function
- Good, I use computers often and fluently
- I can use basic functions such as email
- I am a novice in computer use
- I don't use computers at all

### Are you familiar with software development work?

- Yes, I work as product owner in any project and provide and negotiate feature requirements.
- Yes, I am involved with designing and/or coding the features in a software project
- Yes, but I only follow some projects to know how the development work is going on
- Yes, I know basic terminology about this but have never participated in any development work
- No, I don't know anything about software development work

### If you are associated with any software development work, what is the purpose?

- As a client, I need to monitor the development progress and to make decisions about its continuation
- As a team manager, I need to make sure team is working efficiently and the development work is on schedule
- As a developer/designer, I need to make sure a feature is working properly and satisfies the requirements.
- I am simply interested on the type of the software and wish to know how the work would continue in future

Other, if any \_\_\_\_\_

**Are you familiar with the following platforms for monitoring and visualizing software development data?**

- GitHub
- Jira
- Agilefant
- Trello
- Waffle
- Google Analytics
- Something not mentioned above, if any \_\_\_\_\_
- None

**If you know any of the platforms above, how often have you used them?**

- Daily or nearly daily
- Few times a week
- Few times a month
- More rarely than few times a month
- Never

**Have you ever used Dipor Dashboard (<http://dashboard.digipalvelutehdas.fi/>) before this test?**

- Yes
- No
- I'm not sure

**If you have used the Dipor Dashboard before, when you used it last?**

- Less than a week ago
- Less than a month ago
- Less than 3 months ago

**If you have used the Dipor Dashboard before, how often do you use it?**

- Daily
- Several times during the week, but not daily
- About once a week
- Couple times in a month

**What kind of activities do you perform in Dipor Dashboard?**







---

---

---







---

## USABILITY TASKS<sup>1</sup>

	<p>Look for a yummy brownie recipe using Google. Find the list of ingredients.<sup>2</sup></p>
	<p>Open a browser and go to Dipor Dashboard's page. <a href="https://dashboard.digipalvelutehdas.fi">https://dashboard.digipalvelutehdas.fi</a> Login with the credential given credentials.</p>
	<p>You are really interested on works done by the Ministry of Education and Culture. You want to know more details about it. Find this organization and go to its page.</p>
	<p>There are several departments under this ministry and each department hosts a number of services. Find out the list of services under Department for Basic and Early Childhood Education.</p>
	<p>You wish to know about one of the services and associated information about its development. Go to the service page and find out its status, version and how it is visible to everyone.</p>
	<p>You are interested to know which people are working under this service. Find out some of their names.</p>

<sup>1</sup> Image courtesy: <https://www.vectorstock.com/royalty-free-vector/animal-icons-vector-165003>

<sup>2</sup> Pilot test task to demonstrate think aloud method

	<p>You want to know how much active this service has been over time. Find out the number of commits from two consecutive entries and tell how the dates in these entries relate to each other.</p>
	<p>You are interested to see how many issues have been reported for this service. Find out the number of open issues and the number of closed issues.</p>
	<p>You want more detailed information about open issues over a certain period of time. Find out about how many open issues were there between the times August 31, 2015 to December 23, 2015.</p>
	<p>It is easier for you to track issues if they are somehow categorized. Find out what are the different labels used for issues.</p>
	<p>You are interested about bugs that are produced when a service is developed. Find out the number of bugs under this service.</p>
	<p>You suddenly remember about a service that has similar development work. Add the given data source to get different information about that service development:  <b>User: nrel</b>  <b>Repository: api-umbrella 11</b></p>

---

## USER SATISFACTION QUESTIONNAIRE

<b>Evaluate the following statements by checking the correct answer</b>	<b>Strongly Disagree</b>	<b>Disagree</b>	<b>I don't know</b>	<b>Agree</b>	<b>Strongly Agree</b>
It was easy to learn to use the service	[ ]	[ ]	[ ]	[ ]	[ ]
I found the information I needed easily	[ ]	[ ]	[ ]	[ ]	[ ]
The appearance of the service was pleasant	[ ]	[ ]	[ ]	[ ]	[ ]
I am satisfied with the fluency of the use of the service	[ ]	[ ]	[ ]	[ ]	[ ]
The service included terms and words that were unfamiliar to me	[ ]	[ ]	[ ]	[ ]	[ ]
It was easy to perform the given tasks	[ ]	[ ]	[ ]	[ ]	[ ]
Using the service was frustrating	[ ]	[ ]	[ ]	[ ]	[ ]
I am also going to use the service later	[ ]	[ ]	[ ]	[ ]	[ ]

**What overall grade would you give to the service?  
(on a scale from 1 = poor to 5 = very good): \_\_\_\_\_**

**Thank you! Your response will be processed confidentially**