Aalto University
School of Science
Computer, Communication and Information Sciences

Toni Orpana

Data-driven feedback for eHealth content management

Master's Thesis
Espoo, November 19, 2018

Supervisor: Prof. Marko Nieminen, Aalto University
Advisor: Aki Puustjärvi M.Sc. (Tech.), HUS

Aalto University
School of Science
Computer, Communication and Information Sciences

ABSTRACT OF
MASTER'S THESIS

Author: Toni Orpana

Title:
Data-driven feedback for eHealth content management

Date: November 19, 2018 **Pages:** xi + 82

Major: Computer Science **Code:** SCI3042

Supervisor: Prof. Marko Nieminen, Aalto University

Advisor: Aki Puustjärvi M.Sc. (Tech.), HUS

Web analytics has proven significant potential for constantly improving the provided web-based services and applications. By analyzing interaction data collected from web applications, it is possible to study how the applications are used in detail. The focus of this study is to analyze if interaction data collected with Piwik PRO web analytics platform using JavaScript tagging can provide sufficient detail about user behaviour and interaction in a modern single-page web application. Furthermore, the analysis seeks to answer if the collected data can be refined in a way that will help the content managers of the web application to continuously improve the content and to spot dysfunctional content.

The research is based on Omapolku, a Finnish public e-health service providing digital services for personalized healthcare. In this study, the analysis focuses on evaluating digital treatment pathways in Omapolku, which provides various types of information and utilities designed for the needs of specific patient groups. The evaluation is based on the graphical user interface of a treatment pathway view by analyzing a sample dataset consisting of actions performed by the users. The data is analyzed with general web analytics metrics and by applying statistical analyses of web usage mining.

The results show that the interaction data can provide necessary detail for evaluating general usage metrics and basic usage patterns. However, the results show that the data does not provide necessary information for identifying most actions performed by the users, which makes it practically impossible to link the data to the front-end components of the user interface.

As an outcome of this study, it is recommended that additional identifiers are added to the front-end components of the treatment path interface and that the JavaScript tagging script is modified to record the corresponding identifiers and the action context. In addition, a novel prototype was designed as a solution to the identified challenges and to support the work of the content managers.

Keywords: data-driven, content management, web analytics, interaction data, web usage mining, e-health

Language: English

Aalto-yliopisto
Perustieteiden korkeakoulu
Tieto-, tietoliikenne- ja informaatiotekniikka

DIPLOMITYÖN
TIIVISTELMÄ

Tekijä: Toni Orpana

Työn nimi:
Dataperusteinen palaute eTerveyspalveluiden sisällöntuotantoon

Päiväys:	19. Marraskuuta 2018	**Sivumäärä:**	xi + 82
Pääaine:	Tietotekniikka	**Koodi:**	SCI3042
Valvoja:	Prof. Marko Nieminen, Aalto-yliopisto		
Ohjaaja:	Diplomi-insinööri Aki Puustjärvi, HUS		

Web-analytiikka on osoittanut nykypäivänä potentiaalinsa osana web-pohjaisten sovellusten jatkuvaa kehitystä. Web-sovelluksista kerätyn interaktiodatan analysointi mahdollistaa sen, että sovellusten käyttöä voidaan tutkia yksityiskohtaisesti. Tämä työ keskittyy analysoimaan mikäli Piwik PRO analytiikkapalvelun JavaScript seurantakoodilla kerätty interaktiodata tarjoaa riittäviä yksityiskohtia käyttäjien käyttäytymisestä ja interaktiosta yksisivuisessa web-sovelluksessa. Tämän lisäksi työ keskittyy tutkimaan mikäli kerättyä dataa voidaan jalostaa siten, että sitä voi hyödyntää toimintahäiriöisten sisältöjen paikantamiseen sekä sisällön jatkuvaan kehittämiseen.

Tutkimus perustuu Omapolku-sovellukseen, joka on julkinen suomalainen eTerveyspalvelu. Omapolku tarjoaa digitaalisia palveluita henkilökohtaiseen terveydenhuoltoon. Tässä työssä analyysi perustuu Omapolun digitaalisien hoitopolkujen toimivuuden arvioimiseen. Digitaaliset hoitopolut tarjoavat monipuolista tietoa sekä työkaluja, jotka on suunniteltu potilasryhmäkohtaisesti tietyn hoitotarpeen mukaisesti. Hoitopolkujen toimivuuden arviointi toteutetaan tutkimalla digihoitopolkujen graafisesta käyttöliittymästä kerättyä interaktiodataa. Kerättyä dataa analysoidaan yleisillä web-analytiikan mittareilla sekä tilastollisilla web-tiedonlouhinnan menetelmillä.

Työn tulokset osoittavat, että interaktiodata voi tarjota tarpeellista tietoa yleisten mittareiden laskemiseksi sekä yksinkertaisten käyttäytymismallien selvittämiseksi. Tulokset myös osoittavat, että data ei tarjoa tietoa yksityiskohtaisten tapahtumien alkuperän selvittämiseksi käyttöliittymässä.

Työn tuloksena suositellaan, että digihoitopolkujen käyttöliittymän komponentteihin lisätään lisätunnisteita ja että JavaScript seurantakoodia muokataan siten, että tapahtuman konteksti ja siihen liittyvä komponenttitunniste tallennetaan tapahtumaan. Tämän lisäksi työssä esitetään prototyyppi ratkaisuna havaittuihin haasteisiin sekä tukemaan sisällöntuottajien työtä.

Asiasanat:	dataperusteinen, sisällöntuotanto, web-analytiikka, interaktiodata, eTerveyspalvelut
Kieli:	Englanti

Acknowledgements

This study was conducted in the IT Management of the Hospital District of Helsinki and Uusimaa (HUS) for the Health Village project. Already prior to the first steps of this study, I felt tremendous motivation to help support the development of the important services that Health Village provides. The joyrney was demanding yet rewarding and I feel privileged for the opportunity to be a part of the skilled and enthusiastic team producing high quality information and useful services.

First, I want to express my gratitude to Mikko Rotonen and Sirpa Arvonen at HUS for the opportunity to conduct my thesis at HUS and for the overall support during the thesis work. I also want to thank my thesis instructor Aki Puustjärvi at HUS for his active guidance, insight, suggestions, and overall support.

Special thanks go to my thesis supervisor Prof. Marko Nieminen for his guidance, constant feedback, motivation and domain knowledge. Finally, I want to thank my family and friends for the support during the thesis work and studies at Aalto University in general.

Espoo, November 19, 2018

Toni Orpana

Abbreviations and acronyms

AJAX	Asynchronous JavaScript and XML
CLF	Common Log Format
DOM	Document Object Model
ELF	Extended Log Format
GDPR	General Data Protection Regulation
GUI	Graphical User Interface
HCI	Human-Computer Interaction
HIPAA	Health Insurance Portability and Accountability Act
HUS	The Hospital District of Helsinki and Uusimaa
IS	Information System
IT	Information Technology
ICT	Information and Communication Technology
JSON	JavaScript Object Notation
MPA	Multi-Page Application
MVC	Model-View-Controller
PII	Personally Identifiable Information
RegEx	Regular Expression
SD	Standard Deviation
SPA	Single-Page Application
SQL	Structured Query Language
TWA	Template-based Web Application
URL	Uniform Resource Locator
WUM	Web Usage Mining
XML	Extensive Markup Language

Specific to this study

DTP	Digital Treatment Path/Pathway in Omapolku web application

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The Internet has been perhaps the most revolutionary technological innovation within the last two decades. Initially a research project in *packet switching*—a technique for data transmission between networked computers—the Internet has evolved into a global information infrastructure offering an increasing amount of tools for information acquisition (Leiner et al., 2009). In today's technological world, digitalization is transforming businesses and industries towards the digital world of networked devices. The term digitalization can be defined as the use of digital technologies in changing a business model and to provide new revenue-producing innovations in a competitive global marketplace (Collin et al., 2015). While digitalization has an obvious impact on businesses, it spans also the consumers, allowing better access to digital services. This introduces new market potential and novel opportunities for businesses to reach their customers (Collin et al., 2015).

Web 2.0 is a concept that is commonly used to describe the changing trends in the utilization of web technologies. Modern web applications have evolved from traditional hypertext read-only systems into dynamic media of user-created content and rich interaction (Mesbah, 2009). In the world of today, web applications are steering towards a desktop-style experience. These applications, also known as Rich Internet Applications (RIAs), share two common aspects of Web 2.0 including *collaboration* and *interaction*. Here, collaboration refers to the social aspect where people interact and share the same data. The aspect of interaction allows buildings web applications that behave much like native desktop applications. (Taivalsaari et al., 2008) In this study, the focus is in the context of e-health services—web-based services and applications that are implemented in the domain of healthcare. E-health has become an emerging field in the intersection of medical informatics, public health, and business, offering health services and information via the Internet (Eysenbach, 2001).

Understanding how users *interact* with a web application is essential for measuring the success of that application. Web analytics is an approach based on data collection and analysis of web data for identifying problems and improving the design, user experience, and performance of web applications (Zheng and Peltsverger, 2015). In the context of this study, the fundamental objective of utilizing web analytics is to gain understanding of user interaction and usage patterns. According to Atterer et al. (2006), modern web application interfaces have become more sophisticated and have moved towards the client-side, instead of utilizing the classic client-server request-response paradigm. As a result, it has become more difficult to obtain feedback about the usage of web applications since in the case of many applications, the data available from server logs is not sufficient for discovering detailed information about the usage of the web application. Since the emergence of Web 2.0, typical clickstream data does not necessarily provide desired information, because the traditional *page* paradigm has changed (Kaushik, 2007). In this study, this issue is studied by analyzing a sample dataset consisting of *clickstream data* that was collected from a single-page web application.

1.1 Problem statement

The objective of this study is to examine if the information extracted with web analytics methods and practices—practices that have been traditionally used extensively within the context of e-commerce—could be adopted to support the content management of digital treatment paths in Omapolku, a novel e-health single-page web application in Finland for public use. Digital treatment paths are treatment-specific implementations of tasks that are linked and organized into an intuitive user interface. Currently, there is only limited information of how the application serves the end-users' needs. In order to overcome this problem, it is useful to study how the end-users utilize and interact with the service and how effective the service is in terms of treatment quality. Designed for personal use via the Internet, insight of how Omapolku is perceived in terms of usability and user experience by the end-users is primarily based on free-form textual feedback with a varying level of quality in terms of accuracy and actionability. For this reason, it had become obvious that there is an evident need for researching this subject and to find a solution that can provide more accurate information about the usage of the application.

1.2 Research questions

In order to work with the research problem, the problem can be formulated into the following research questions:

- **RQ1:** Can web analytics provide detailed information about user behaviour and interaction?
- **RQ2:** What are the challenges of adopting web analytics on a single-page web application?
- **RQ3:** Can the output from analytics tools be refined in a way that it will aid Omapolku content managers in spotting dysfunctional content and improving it?
- **RQ4:** Does user interaction data reveal similar issues that have been reported via customer feedback?

Here, RQ1 tries to seek answers if traditional web analytics approaches and tools could provide information in sufficient detail to help analyze user interaction of a single-page application with a relatively complex design including many components subject to user interaction. RQ2 tries to discover what the challenges will be faced when integrating web analytics into a single-page web application. RQ3 focuses on studying if interaction data can be refined in a way that can provide essential knowledge for Omapolku content managers to help improve the content and spot dysfunctional content. Finally, RQ4 tries to discover whether issues reported via textual customer feedback can be revealed by utilizing web analytics.

1.3 Scope and limitations

In this study, web analytics methods are studied from the perspective of gaining insight from the end-users' point of view of how the service implementation performs in practice. The decision for choosing this viewpoint was found to be the most relevant and useful, establishing the grounds for a new prototype feature that was designed concurrently during the study to help provide valuable information for the content managers of Omapolku. In this study, the process is based on the information found in the literature, collected user interaction data, and customer feedback. This study will not

particularly focus on evaluating the technical performance or software implementation of Omapolku as such. Instead, means for gaining insight of possible issues with the application in terms of usability or user experience will be examined in the light of continuous data-driven improvement.

1.4 Structure of the study

The structure of this study is divided into eight chapters. Chapter 1 introduces the subject of this study, presents the purpose and motivation of this study, defines the research problem and related research questions, and describes the chosen scope and limitations. Chapter 2 presents the environment of this study. Chapter 3 provides the necessary concepts and theory as a basis to support the analysis of the results. Chapter 4 presents the used methods, the process of data collection and analysis, and justifies the utilization of prototyping as a part of the research approach of this study. Chapter 5 presents the results and works as a basis for justifying the need for a novel prototype. Chapter 6 introduces a protoype design of a novel software component for Omapolku content managers. Chapter 7 assesses the legibility and implications of the results, reflects on the proposed prototype and provides answers to the research questions. Finally, Chapter 8 concludes this study and provides suggestions for future research.

Chapter 2

Environment of the study

This chapter presents the project environment relevant to this study. First, Section 2.1 provides a short introduction to public healthcare in Finland. Next, Section 2.2 presents the organization in which this study was conducted. Section 2.3 lays down the foundations of this study by presenting the large-scale project framework, the Virtual Hospital 2.0 initiative. Section 2.4 introduces Health Village, a key project of Virtual Hospital 2.0. Finally, Section 2.5 presents the case web application Omapolku which was analyzed in this study.

2.1 Health services in Finland

Public healthcare in Finland is governed by the Finnish constitution. According to The Constitution of Finland (731/1999, Section 19), it is the responsibility of public authorities to promote the health of the population. Healthcare services are primarily financed from the tax revenue and provisioned by municipalities. In Finland, there are a total of 311 municipalities of which 295 are geographically part of Mainland Finland. Each municipality belongs to one of the 20 hospital districts in Finland.

On municipal level, primary health care services are provided by health centres. A municipality may either operate its own health centre or cooperate it together with other municipalities. Some municipalities have purchased nearly all of their health centre services from service providers in private sector. In contrast to primary health care services, specialized medical care is provided by hospital district hospitals. (Kuntaliitto, 2017; HUS, n.d.a)

2.2 HUS

The Hospital District of Helsinki and Uusimaa (HUS) is the largest hospital district in Finland. HUS is a Joint Authority formed by 24 municipalities which provides specialized medical care services for over 1.6 million residents within the associated municipalities. As a comparison, the second largest hospital district—the hospital district of Pirkanmaa—covers the health care of only a third that of HUS with around 0.5 million residents. (Kuntaliitto, 2017; HUS, n.d.a) HUS employs over 22,000 medical professionals from different branches of healthcare and other expertise (HUS, n.d.b).

2.3 Virtual Hospital 2.0

Virtual Hospital 2.0 ("Virtuaalisairaala 2.0") is a joint initiative of five Finnish hospital disctricts for producing healthcare services between 2016–2018. Finnish university hospital districts participating in the initiative are presented in Figure 2.1, based on the illustration by Kuntaliitto (2017), and include the following hospital districts:

- the Hospital District of Helsinki and Uusimaa
- the Pirkanmaa Hospital District
- the Northern Ostrobothnia Hospital District
- the Hospital District of Northern Savo
- the Hospital District of Southwest Finland

The vision of Virtuaalisairaala 2.0 is to provide quality treatment for all citizens regardless of their residence, age, and digital expertise (Virtuaalisairaala 2.0, n.d.). Laying down the foundations for virtual health services, HUS implemented Mental Health Hub ("Mielenterveystalo") in 2009, offering patients virtual mental health services, including special treatment like psychotherapy and addiction counselling. The experiment with Mental Health Hub was a success, which lead to the idea of expanding the program to include other medical specialties (Microsoft, 2017).

According to Virtuaalisairaala 2.0 (n.d.), by making healthcare services available to all Finnish citizens, regardless of their place of residence and income level, the project improves the equality of citizens. Digital services complement traditional healthcare before and during treatment and during

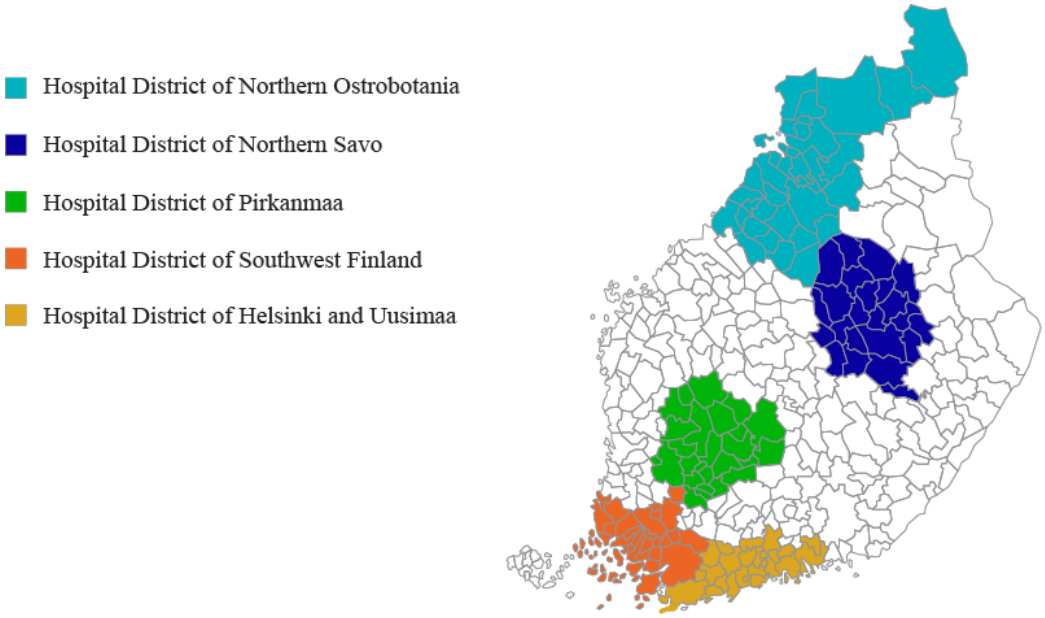

Figure 2.1: Hospital districts associated with Virtuaalisairaala 2.0.

the monitoring stage of a treatment. In addition, the role of proactive treatment increases and possibilities for monitoring the quality of life increase significantly. Digital healthcare services also streamline and enhance treatment processes while diversifying the work in the healthcare sector. The objective of digital services is to also increase the effectiveness and customer satisfaction and to continuously collect information for further development of the services. The goal for developing virtual e-services that intertwine with the treatment of patients in specialized medical care include:

- enhancing treatment and prevention of diseases
- enabling customers with timely admission for medical treatment
- decreasing the total time of finding and receiving an appropriate service
- producing treatment assessment and screening more personally
- reducing needless contacts and clinic visits
- increasing customer satisfaction
- improving the efficiency of healthcare professionals
- producing savings and improving operation processes
- developing novel, innovative, and competitive services

2.4 Health Village

Health Village ("Terveyskylä") is a web service for specialized medical care, developed in collaboration with patients. The service provides information and support for patients and citizens, and tools for medical professionals. The services provided by Health Village supplement traditional healthcare services and are suitable for monitoring quality of life, assessing symptoms, and living with a long-term illness. Along with the novel services, both the importance of preventive healthcare and possibilities for individuals to maintain personal health increase.

Health Village consists of thematic information"hubs", also known as "houses", each focusing on a certain healthcare theme, e.g., allergy and ashtma. At the time of writing this study, Health Village has opened 30 different hubs. In addition to hubs, Health Village's services include Own Path ("Omapolku"), Health Village PRO ("TerveyskyläPRO"), chatbots and other useful services. (Terveyskylä.fi, n.d.b)

2.5 Omapolku

Omapolku ("Own Path") is a digital service channel for personalized health care. Omapolku can be used for participating in remote appointments and as a secure messaging channel. Omapolku provides a platform for connecting *digital treatment paths* to patients' treatment plan via a referral made by a doctor. In addition to treatment-specific digital treatment pathways, Omapolku contains different self-care programs and other useful functionalities. (Terveyskylä.fi, n.d.a) Digital treatment paths are designed for patients and their main components include (Virtuaalisairaala 2.0, n.d.):

- **Registration and authentication**. By authenticating via Health Village, an individual can create an account to Omapolku. The services provided by Omapolku are available after an authentication. In order to see personal information related to a treatment, a patient is required to authenticate via strong authentication.

- **Digital treatment paths**. Digital treatment pathways (DTPs) provide personalized information, symptom assessment, exercises, self-monitoring, questionnaires and other activities that have been designed for the needs of specific patient groups.

- **Messaging**. Users can receive messages sent by medical professionals. In addition, users can choose to receive reminders via email or text

messages. Messaging provides a secure channel for patient-professional communication.

- **Research consent**. Patients may give consent for research, allowing the data collected from the patient or data provided by the patient to be used for research purposes. Given a consent, the data can be saved in a medical record system. However, conducting research on the data requires an accepted research plan and approval from an ethical committee.

- **Application catalogue**. A collection of mobile and Internet of Things (IoT) applications produced by Health Village or other vendors that follow the given requirements and guidelines. These applications can be integrated to digital treatment paths.

- **Meters and devices**. Omapolku supports integrating various devices and meters, such as weight scales and blood pressure meters. The data produced by the meters and devices can be integrated into digital treatment paths for data visualization.

- **Symptom assessment**. Patients can assess symptoms according to a schedule determined by a medical professional. Patients will receive different self-care programs or a recommendation for seeing a medical professional based on symptom assessment algorithms.

In many cases, symptom assessment will provide either treatment recommendations or service guidance. In both of these cases, the software used for the assesment will be treated as a medical device. The development of medical devices is heavily regulated and requires rigorous research. The usability of these services is of highest importance since there is no professional present to give guidance for the patient in case of misunderstandings. For this reason, the results of this study will be important also in this sense.

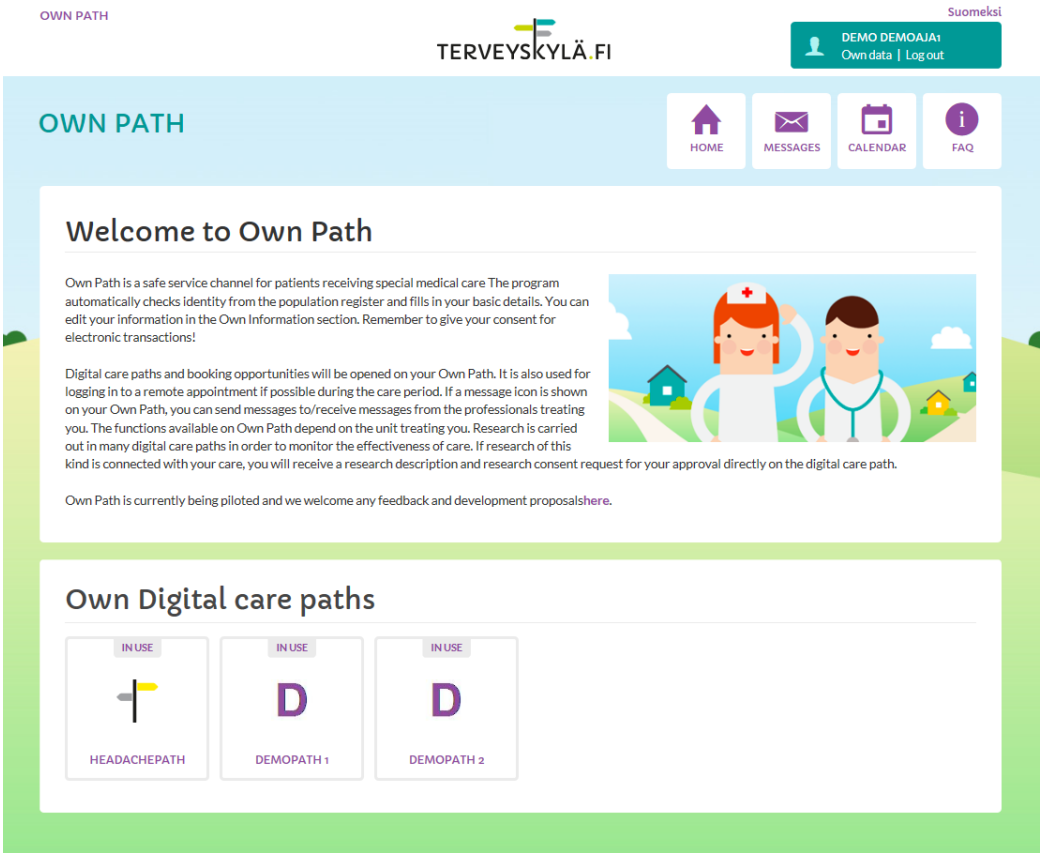

Figure 2.2: Example graphical user interface of Omapolku front page.

Chapter 3

Background concepts

This chapter introduces essential background concepts and theory relevant to this study. A fundamental part of the literature utilized in this study concerns the broad domain of web applications, web analytics, and data mining. Considering the broadness of these topics, only relevant part of the theory was included and presented in detail. Section 3.1 starts by introducing the concept of the Web as an application platform, followed by introducing the concepts of web applications and novel single-page applications. Section 3.2 introduces web analytics a web analytics process based on the current best practices found in the research ad literature. Section 3.3 describes web usage mining in the context of interaction analytics, providing an overview of essential analytics methods used for analyzing interaction patterns from web data.

3.1 Web applications

In the early days of the Internet, the purpose of websites of the time was to serve simple, static pages—pages with basic content such as images and text—to the client. Since then, after the introduction of e-commerce, businesses have started to require means for providing dynamic and up-to-date information for their clients. (Jadhav et al., 2015) Interaction in traditional multi-page web applications (MPAs) was based on a *multi-page model* where each web page was requested separately from the server. After the server response, the entire user interface of the web application was refreshed, resulting in poor interactivity and responsiveness. (Mesbah and van Deursen, 2007) Later, new web development techniques arose including the utilization of Asynchronous JavaScript and XML (AJAX) for implementing inteactive and dynamic web applications (Mesbah and van Deursen, 2007).

With this new approach, it was possible to create dynamic and interactive single-page web applications (SPAs) that are based on a *single-page model* Garrett (2005). Technologies, such as AJAX, which are used for implementing modern web applications are commonly referred collectively as *Web 2.0* technologies (Taivalsaari et al., 2008).

3.1.1 Asynchronous JavaScript and XML

Asynchronous JavaScript and XML (AJAX) was originally defined by Garrett (2005). AJAX allows developing more interactive web-based user interfaces by utilizing a combination of established technologies used in web development. The fundamental difference between classic web interfaces and AJAX interfaces is that the latter utilizes asynchronous communication between the client and the server (Mesbah and van Deursen, 2007). As mentioned earlier, traditional web interfaces rely on a multi-page model where client requests were handled syncronously. However, with AJAX, web developers can create web applications with a single-page model where requests can be made asynchronously without a need for updating the whole interface. (Mesbah and van Deursen, 2007)

As specified by to Garrett (2005), AJAX incorporates a combination of multiple existing web technologies, including

- standards-based presentation using XHTML and CSS,
- dynamic display and interaction using the Document Object Model,
- data transportation and manipulation XML and XSLT,
- asynchronous data retrieval via XMLHttpRequest,
- and JavaScript binding everything together.

According to Mesbah and van Deursen (2007), AJAX changes the way of approaching web application developement where instead of seeing interaction as a sequence of web pages, web applications can be developed using a more intuitive single-page model based on updating user interface components.

3.1.2 Multi-page application model

Web applications based on a multi-page application model consist of separate web pages, where each page has a unique Uniform Resource Locator (URL)

(Mesbah et al., 2012). Applications based on the multi-page paradigm are also called *multi-page applications* (MPAs) (Oh et al., 2013). In order to interact with a multi-page web application, the user has to send a request to the web server by clicking a hyperlink or by submitting a form (Zepeda and Chapa, 2007).

Figure 3.1 presents the request-response cycle of the multi-page application model based on the work by Jadhav et al. (2015). The request-response cycle consists of the following steps:

1. The client makes the initial request to open the web application.
2. The web server responds to the request with a HTML document and other necessary resources including as CSS stylesheets and JavaScript.
3. When the user interacts with the application, e.g., by clicking a hyperlink or a button, a new request is sent to the web server.
4. The web server processes the new request and responds with a new HTML document.
5. The client processes the response and reloads the whole web page with the new content.

The client-server communication previously described is completely *synchronous*, which is usually sufficient in most applications consisting of simple static pages. However, in applications requiring more interactivity, the synchronous client-server communication is does not necessarily provide sufficient response times since the client has to wait until the server-side processing of the request has been completed. Using a synchronous multi-page paradigm for interactive applications is not necessarily a good solution due to low performance, loss of application states, higher bandwidth usage, and redundant data transmission. (Zepeda and Chapa, 2007)

According to Oh et al. (2013), using a template-based approach to create multi-page web applications has become popular since a template system reduces code redundancy, improves productivity and modularity, therefore making the application more maintainable. Such applications are also known as template-based web applications (TWAs). According to Oh et al. (2013), while a template-based approach has obvious benefits, it does, however, suffer from the same issues regarding interactivity since a whole HTML document is still being requested from the server and will be rendered at client side according to the same request-response cycle that was previously described.

In order to overcome the limitations of the multi-page application model in terms of interactivity, previous research has proposed an approach that automatically migrates multi-page applications to single-page applications. Mesbah and van Deursen (2007) suggest a migration process consisting of the following steps:

1. Retrieving web pages.
2. Extracting navigational paths.
3. UI component model identification.
4. Single-page UI model definitions
5. Target UI model transformation.

In simple terms, the migration process utilizes an algorithm which crawls through the multi-page web application, creates a model of the current application structure and transforms it to a single-page model. It should be noted that this procedure is designed to transform an existing web application based on a multi-page model.

3.1.3 Single-page application model

Many modern web applications are implemented with a single-page application model which provides an experience much like a desktop application (Jadhav et al., 2015). Unlike applications designed with a multi-page model, single-page applications consist of a single web page interface. The interface of a single-page application is composed of individual components, which allows creating, deleting, editing, and updating each component individually. (Zepeda and Chapa, 2007)

Figure 3.2 illustrates the request-response cycle of a single-page application model based on the work by Jadhav et al. (2015). The request-response cycle has similar characteristics compared to a multi-page model, but the significant difference is in the data exchange approach when the use interacts with the application. The request-response cycle of a single-page model consists of the following steps:

1. The client makes the initial request to open the web application.
2. The web server responds to the request with a HTML document and other required resources including CSS stylesheets and JavaScript.

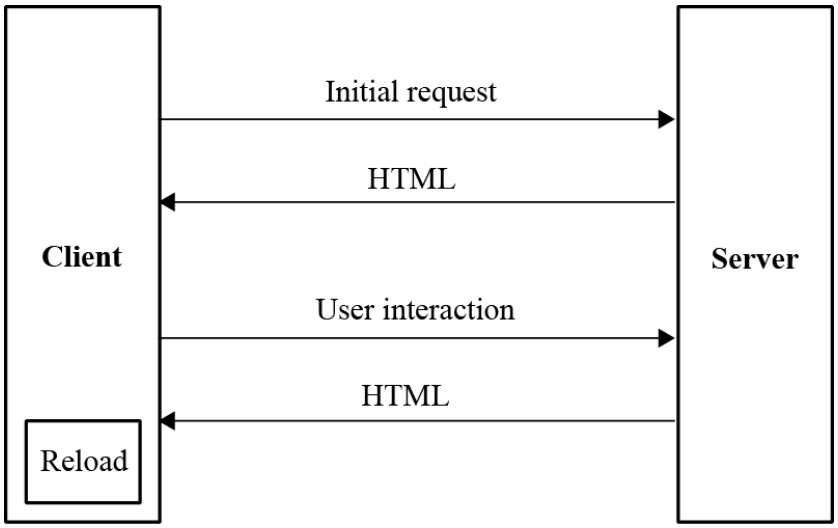

Figure 3.1: Multi-page application request-response cycle. When the user interacts with the web application, the server responds by returning a complete HTML document.

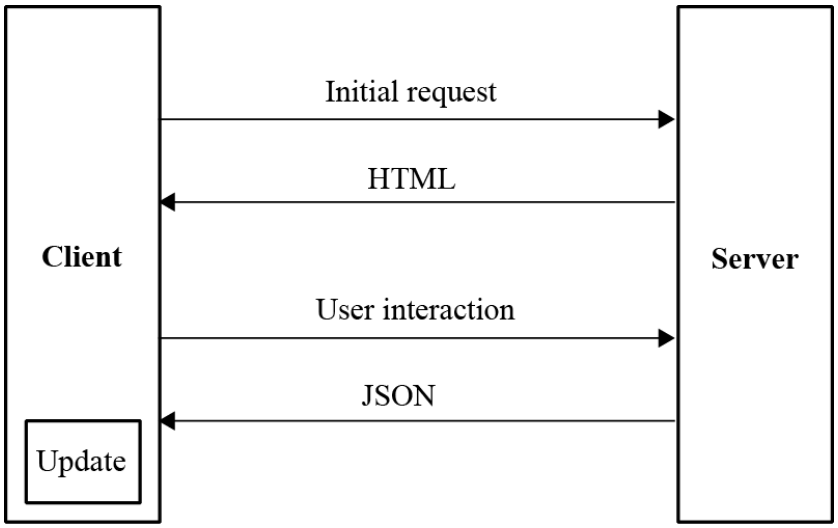

Figure 3.2: Single-page application request-response cycle. Unlike in the case of multi-page application model, the server responds with a JSON object containing only the relevant data.

3. When the user interacts with the application, e.g., by clicking a hyperlink or a button, a new request is sent to the web server.

4. The web server processes the new request and responds with JSON data including only relevant data regarding the performed action.

5. The client processes the response and replaces and/or updates only the required components of the HTML document with the new content.

The client-server communication in a single-page model utilizes AJAX and is therefore *asynchronous* in nature. Zepeda and Chapa (2007) states that while it would seem like adding a layer—the AJAX engine—to the application would make the application less responsive, AJAX actually eliminates the need for reloading complete HTML pages after each user interaction, resulting in more efficient data transmission.

From architectural point of view, implementing the front-end for a single-page application using JavaScript becomes quickly complicated. (Jadhav et al., 2015). In order to tackle with the complexity issue of maintaining the front-end JavaScript code, several front-end frameworks have been developed for implementing SPAs, including *AngularJS*, *React*, and *Vue.js*.

3.2 Web analytics

Waisberg and Kaushik (2009) define web analytics as "the science and the art of improving websites to increase their profitability by improving the customer's website experience." In simple terms, web analytics utilizes scientific methods, such as statistics and data mining, and is subject to many creative decisions that have a significant impact on the achieved results. Waisberg and Kaushik (2009) provide a brilliant illustration of the latter: where a painter has to choose from many colors and find a perfect color mix, an analyst or marketer has to find relevant data sources in order to provide actionable results. A more technical definition of web analytics was originally defined by the Web Analytics Association and was proposed as a standard definition. According to this definition, web analytics is the act of "objective tracking, collection, measurement, reporting, and analysis of quantitative Internet data to optimize websites and web marketing initiatives.". While web analytics as a term may sound purely technical, both scientific methods and artistic decisions have a significant role in the context of this study.

In the beginning of the Internet era, websites were much simpler compared to websites of today. A user would type in a website address and a

Uniform Resource Locator (URL) used to identify the file in the web server, after which the web server would process the request and return a file including simple text and hyperlinks. Eventually, it was noticed that errors occured, preventing the transmission of the requested file, or that some links were incorrect and therefore resulted in a failure. These requests were captured in a server-side access log. It was discovered that these entries could actually be used to find information about the users who had made requests ("hits") to the server. Server logs did not only contain information about the hit, but also other information such as the filename, time, referrer, IP addess, operating system, and a browser identifier. With this information, it was possible to get more in-depth information about users and where the hits came from. Eventually, when log files started to grow in size, a script that would automatically parse the log files and retun basic metrics was created. Analog is one of the first widely available log file analysis programs, implemented by Dr. Stephen Turner in 1995. (Kaushik, 2007)

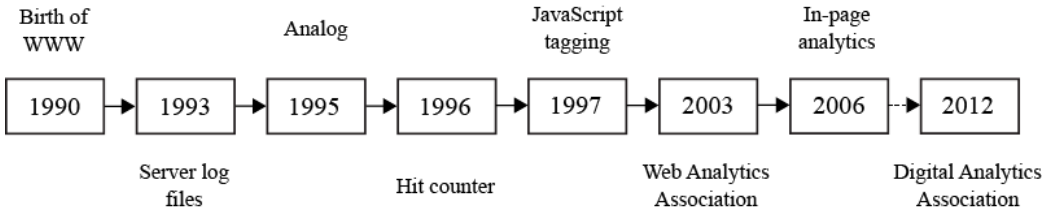

Figure 3.3: Evolution of web analytics.

Figure 3.3 illustrates the evolution of web analytics based on the illustration by (Singal et al., 2014). A practical example of how rapidly the discussion around the topic of web analytics has evolved is that on March 20th, 2007, a Google search with keywords

```
"web analytics" + definition
```

yielded *642,000 results* (Kaushik, 2007), whereas at the time of writing this study, roughly 11 years later, the same search yields *22,6 million* results, meaning an enormous 3420% increase in the search results. Without a doubt, the discussion and applications within the domain of web analytics has seen an impressive growth since the first applications of web analytics. As it can be seen in Figure 3.3, it took only around 7 years, after Internet was publicly available, when the first implementations of JavaScript tagging—a data collection method essential also in this study—already existed.

According to Waisberg and Kaushik (2009), a general framework for web analytics, based on the field's best practices, is presented in Figure 3.4. The main steps of the framework will be presented and discussed next in detail. Here, it is important to realise that the process does not assume immediate *success*; the results and outcomes of the analysis should be evaluated against the objectives and refined if necessary.

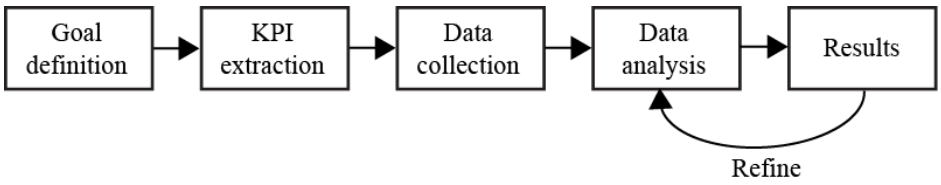

Figure 3.4: Web analytics process.

3.2.1 Goals and metrics

Every web-based application—be it a traditional website or a novel single-page web application—has a defined purpose and a reason why it exists. According to Waisberg and Kaushik (2009), answering this question is critical when defining the *goals* for a website. Defining the objectives of a website serves as a fundamental input in identifying the key metrics, namely *Key Performance Indicatiors (KPIs)*, which can be used to measure the success of the website with respect to the defined goals. Waisberg and Kaushik (2009) emphasizes that a website should be accounted like any other business expenses where investments should be measured against the returns.

According to Singal et al. (2014), web analytics metrics provide baseline information for analyzing data collected from web usage. Singal et al. (2014) suggests that these metrics can be divided into four categories: *site usage*, *referrers*, *content analysis*, and *quality assurance*. Table 3.1 presents the key metrics that have been divided into these four categories.

Usage	**Referrers**	**Content**	**Quality**
Number of visitors and sessions	Websites working as traffic sources	Top entry pages	Erronous pages
Number of returning visitors	Search terms used to find the website	Most popular pages	Visitor behaviour in case of an error

Geographic information	How many visitors bookmark the website	Top exit pages	
Search engine activity		Top clickpaths though the website	

Table 3.1: Types of web analytics metrics.

In this study, metrics concerning site usage and content analysis are particularly important and are utilized for analyzing the sample data.

3.2.2 Data collection

Two of the commonly used methods for collecting usage data in the domain of web analytics are *web server logging* and *page tagging* (Zheng and Peltsverger, 2015; Singal et al., 2014; Sen et al., 2006). However, other methods for collecting usage data exist, such as *network-pipe* based approaches relying on TCP/IP packet sniffing (Sen et al., 2006; Kaushik, 2007; Waisberg and Kaushik, 2009; Singal et al., 2014). In this study, only web server logging and page tagging will be presented in detail.

Web server logging

Web server logging is a classic method for collecting usage data in web analytics (Zheng and Peltsverger, 2015). Web server logs have served as the original source for data collection ever since the emergence of the Internet (Kaushik, 2007, p. 26). A server log file is generated by a web server which records server activities and HTTP request headers in a textual format. A commonly used, standardized text file format for server logs is known as the NCSA Common Log Format (CLF) (Zheng and Peltsverger, 2015). W3C (1995) defines the Common Log Format (CLF) as follows:

```
remotehost rfc931 authuser [date] "request" status bytes
```

Table 3.2 defines the attributes of the CLF. According to W3C (1995), the benefit of using a standardized format is that it provides a possibility to use generic statistics programs for analyzing the contents of the log file.

Attribute	Definition
`remotehost`	Remote hostname or IP number, if DNS hostname is not available
`rfc931`	The remote logname of the user
`authuser`	The username of the authenticated user
`[date]`	Date and time of the request
`"request"`	The request line in exact form that was received from the client
`status`	The *HTTP status code* that was returned to the client
`bytes`	The content-length of the transferred data in bytes

Table 3.2: NCSA Common Log Format.

A practical example illustrating the CLF can be seen by running a simple local Python server. In the following example print, a local Python server is run on Django framework v1.7.1:

```
[28/Oct/2018 19:20:45] "GET /en/ HTTP/1.1" 200 3749
```

As a side note, the first three attributes `remotehost`, `rfc931`, and `authuser` are not included in the example server log due to the logger configuration. That being said, however, the CLF structure is widely used for server-side logging with slightly varying formatting configurations. In addition to CLF, another standardized format is the Extended Log Format (ELF). The following output presents an example of the ELF as defined by W3C (n.d.):

```
#Version:  1.0
#Date:  12-Jan-1996 00:00:00
#Fields:  time cs-method cs-uri
00:34:23 GET foobar.html
12:21:16 GET foobar.html
12:45:52 GET foobar.html
12:57:34 GET foobar.html
```

According to W3C (n.d.), the ELF format is more extensible and permits a wider range of data to be captured compared to the CLF format. In general, Waisberg and Kaushik (2009) describes the process collecting data into a web server log includes the following steps:

1. Client makes the initial request to the web server by typing an URL.

2. The web server creates an entry in the log file including details about the request.

3. The web server responses with a HTML page.

JavaScript tagging

JavaScript tagging is currently a popular method for collecting web analytics data. The benefit of JavaScript tagging is that it allows collecting very detailed and accurate data (Kaushik, 2007). Currently, many vendors providing web analytics tools, such as Google Analytics and Piwik PRO, rely on JavaScript tagging. From organizational point of view, companies no are no longer required to build their own infrastructure to collect data since vendor-provided tools based on JavaScript tagging collect data on third-party servers (Kaushik, 2007). Singal et al. (2014) states that the benefits of page tagging over web server logging include, but are not limited to, near real-time reporting and the ease of recording additional information. Singal et al. (2014) also state, however, that collecting data with page tagging requires additional scripting on the client-side code and causes additional bandwidth usage on at page load when the JavaScript tag is loaded from the server. In addition, page tagging can only record *successful* page loads and therefore is not capable of recording failures like web server logs.

Kaushik (2007) and Waisberg and Kaushik (2009) describe the basic process of JavaScript tagging as follows:

1. Client makes the initial request to the web server by typing an URL.

2. The web server responses with a HTML page including a JavaScript code snippet embedded to the page.

3. When the page loads on client-side, the JavaScript code is executed which in turn records the page view and details about the session. This information is the sent to the data collection server.

In order to make it easier for the users to adopt JavaScript tagging in their web applications, web analytics platforms including Piwik PRO and Google Analytics have implemented a useful *tag manager* functionality which allows quick and easy updating of JavaScript tracking codes without a need for manually editing the tracking code. As a result, configuring tracking on a web-based application requires less technical expertise when a tag manager is used.

Both Google Analytics and Piwik PRO JavaScript tracking tags can be integrated into a web application either by using a *synchronous* or an *asynchronous* JavaScript tag. In a synhronous implementation, the JavaScript tag will be first loaded and executed prior to loading rest of the page content. In contrast, an asynchronous tag will be loaded and executed along with the rest of the page content. The main difference is that a synchronous tag prevents loading the rest of the page content until the tag has been loaded and executed, causing possible render blocking in the web application interface. A synchronous tag is usually placed in the `<head>` element of the HTML code. In the case of an asynchronous tag, the tag is placed at the end of the `<body>` element. The decision of using either a synchronous or an asynchronous tag depends on the tracking requirements. In general, a good practice is to use an asynchronous tag unless a synchronous tag is particularly required.

3.2.3 Data analysis

In order to extract information about user behaviour from the collected web analytics data, Waisberg and Kaushik (2009) suggests that specific tasks should be performed first. These initial tasks include

- extracting basic metrics with a web analytics tool, such as Google Analytics,
- analyzing traffic sources,
- studying pages with highest bounce rates,
- visualizing the data,
- focusing on the critical KPIs,
- implementing changes by applying the information provided by the findings.

In this study, the one of the main objectives of the analysis is to derive basic metrics from the collected data. For this reason, the other tasks previously mentioned will not be thoroughly discussed. Commonly used basic metrics, as described by Waisberg and Kaushik (2009), are presented in Table 3.3.

Metric	Definition
`Visits`	The number of distinct sessions on the website, i.e., the number of visitors that have interacted with the website.
`Bounce rate`	The percentage of sessions including only one page view.
`Page views`	The number of pages that have been visited in total.
`Pages per visit`	The average number of pages visited during each visit.
`Average time on site`	The average duration of a session.
`% of new visits`	The percentage of sessions where the user visited the website for the first time.

Table 3.3: Common metrics in the domain of web analytics.

According to Phippen et al. (2004), it is imperative to study and understand the factors that contribute to *dropouts*, i..e., users that leave the web content. In other words, it is essential to understand how the users can be kept instead of losing them. In the context of web analytics, the term *conversion* is usually applied to denote a user that has completed a targeted action (Zheng and Peltsverger, 2015). Here, combining general metrics such as *bounce rate*, *averate time on site*, and *pages per visit* can be used to derive measurements for evaluating the user experience. For instance, Phippen et al. (2004) provides two examples of such measurements including a *stickiness factor* (F_S) and *relevance factor* (F_R) as follows:

$$F_S = \frac{\textit{Time spent viewing all pages}}{\textit{Total number of unique visitors}} \tag{3.1}$$

$$F_R = \frac{\textit{Number of content pieces consumed by a visitor}}{\textit{Number of existing content pieces}} \tag{3.2}$$

However, Phippen et al. (2004) makes a remark that these example measurements should be considered only *illustrative* and that various other measurements exist, depending on the objectives of the analysis. According to Waisberg and Kaushik (2009), numbers and metrics as such can be overwhelming for many and therefore a visual presentation can deliver better

understanding of the data. Since there are various different ways to present data in a visual format, it is not reasonable to present all viable visualization techniques in this study. As an example of visualizing data, Figure 3.5 presents an example showing various common metrics in Google Analytics dashboard. Section 3.3 describes the process of *web usage mining* for analyzing and visualizing usage patterns and user behaviour. Since the practices of web usage mining are essential in this study, it will be used to extend the basic analysis process previously discussed.

Figure 3.5: Example dashboard of common metrics in Google Analytics.

3.3 Web usage mining

Thomas (2012) states that some attributes of users' experience on a web site can be deduced from their behaviour on the website; for instance, navigation patterns including "circling" back to previously visited pages and "swapping" to a search engine may indicate difficulty in navigating the website. By extracting these patterns from web log data, it is possible to gain knowledge about where and why the website is hard to navigate. For such information—information that is not knowingly provided by the users—Atterer et al. (2006) have applied the notion of *implicit interaction* as "the observable interaction behaviour of the user with an application that is not done consciously while focusing on reaching a goal".

Web usage mining (WUM), as defined by Srivastava et al. (2000), is the utilization of data mining techniques to discover usage patterns from the collected web data. This data can be collected from many sources, including server-side logs and client-side JavaScript. The patterns discovered from the data are usually presented as web pages, objects, or resources that are frequently accessed by users with common interests (Das and Turkoglu, 2009). Web usage mining has many application areas, including personalization, system improvement, website modification, business intelligence,

and usage characterization (Das and Turkoglu, 2009). A common web usage mining process is presented in Figure 3.6, consisting of three phases: *data pre-processing*, *pattern discovery*, and *pattern analysis* (Srivastava et al., 2000; Das and Turkoglu, 2009). Each step of the process will be discussed in detail in the following sections.

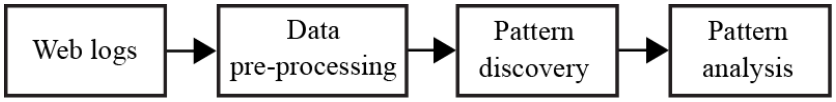

Figure 3.6: Web usage mining process.

3.3.1 Data pre-processing

The first step in web usage mining is data pre-processing. Pre-processing is particularly important and usually a complex process which has a critical role in the successful extraction of suitable data. The goal of data pre-processing is to offer a reliable and structural dataset for parttern discovery (Das and Turkoglu, 2009). As previously discussed in Section 3.2.2, web server log data can be presented in various formats. In addition to CLF and ELF log formats, other log formats such as the Microsoft log format exists (Das and Turkoglu, 2009).

According to Das and Turkoglu (2009), after the appropriate data source has been chosen, further pre-prosessing tasks include *data cleaning* and *session identification*. The objective of data cleaning is to remove irrelevant entries from the raw web log data that are not valid for pattern analysis. In order to accomplish this, Das and Turkoglu (2009) suggest identifying and removing

- unsuccessful entries that resulted in an error or failure,
- entries that were automatically generated by a search engine agent, and
- entries that concern requests to static assets, such as media files, JavaScript files, and stylesheet files.

After the raw log data has been cleaned, the next step is to identify user sessions from the data. Liu and Kešelj (2007) describe session identification as a "process of segmenting access log of each user into individual access sessions". Thomas (2014) define a *session* as a collection of all pages viewed by the same user that are not broken by inactivity of no more than 30 minutes. Cooley et al. (1999, p. 15) states that many commercial web analytics

platforms use *30 minutes* as a default timeout for session identification. According to Thomas (2014), constructing a session can be done by identifying distinct users by their IP-addresses and HTTP user-agent headers, and utilizing time stamps of the entries in the access log.

In the context of this study, web usage mining was utilized for analyzing interaction data. The utilized interaction data is not in the format of a server log file, i.e. CLF or ELF. Regardless of this, the principles of data pre-processing follow similar principles like previously discussed. The tasks concerning the pre-processing of the interaction data utilized in this study are presented Section 4.4.2.

3.3.2 Pattern discovery

Pattern discovery utilizes different methods from several fields including data mining, statistics, pattern recognition, and machine learning (Srivastava et al., 2000). Several techiques for performing pattern discovery exist in the domain literature, including *statistical analysis*, *associations rules*, *clustering*, *classification*, *sequential patterns*, and *dependency modeling* (Srivastava et al., 2000; Eirinaki and Vazirgiannis, 2003). In this study, statistical analysis was chosen as the primary approach for discovering interesting patterns from the used data. For this reason, other techiques are not presented in detail.

According to Srivastava et al. (2000), different statistical techniques are the most commonly used methods for extracting descriptive insights about individual sessions. For instance, analysis can be performed on the session data—e.g., page views, session duration, navigation path length—using statistical measuments such as *frequency*, *mean*, and *median*. Naturally, choosing the analysis measurements depend on the objectives of the analysis. In this study, frequency, mean, and standard deviation were seen particularly useful for analyzing the interaction data. Srivastava et al. (2000) makes a remark that whereas simple statistical analyses do not necessarily provide in-depth information, the knowledge can turn out to be useful for improving, e.g., the performance and security of the implementation. Furthermore, the information can be used to help planning changes in the current implementation and also provide support for marketing decisions.

3.3.3 Pattern analysis

Pattern analysis is the final step of the WUM process previously discussed. The motivation behind pattern analysis is to discard irrelevant and useless patterns and rules that were found during the pattern discovery phase. In

other words, the objective of pattern analysis is to extract only the interesting rules, patterns and statistics. In practice, conducting pattern analysis usually utilizes query mechanisms such as Structured Query Language (SQL). Different visualization techniques, such as graphical patterns and coloring, can be used to help visualize overall trends and patterns in the data. Srivastava et al. (2000); Das and Turkoglu (2009)

Chapter 4

Methods and data

This chapter presents the chosen methods and describes the process of data collection and analysis that was utilized in this study. Section 4.1 presents *design science* in information systems research that was seen practical and suitable research framework in this study. The section describes how prototyping was utilized as a part of this study to propose a new software component prototype for Omapolku content managers. Section 4.2 presents the essential tools that were utilized in this study. Section 4.3 presents the utilized data and describes the process and tools that were used to collect the data. Finally, Section 4.4 describes how the data was processed and analyzed.

4.1 Prototyping

Already during the first steps of this study, an idea for a dedicated analytics view that would support the content management in Omapolku emerged during the planning phase. According to the initial idea, having a dedicated analytics view for Omapolku content managers was seen as a prominent solution for providing actionable insights based on the collected interaction data.

Hevner et al. (2004) state that Information Systems (ISs) that are implemented within an organization have a purpose of improving the efficiency and effectiveness of the organization. Here, the capabilities of the implemented system and various different characteristics of the organization, its existing systems, development and implementation practices, and its people all together influence how this purpose is achieved in practice. Hevner et al. (2004) argue that the productive application of information technology to organizations and developing and communicating the knowledge concerning both the management of information technology and the application of

information technology for managerial and organizational purposes involves acquiring knowledge of both *behavioral science* and *design science*.

Behaviour science seeks to both develop and justify theories that explain organizational and human phenomena surrounding the design, implementation, management, and use of information systems. Such theories can be used to define such factors that need to be managed in order to achieve the purpose of the implemented information system. These theories both impact and are impacted by design decisions of the development methodology, functional capabilities, information content, and human interfaces of the system. In contrast to behaviour science, design science has its roots in engineering and is fundamentally a problem-solving paradigm. Design science seeks to create innovations by defining the ideas, practices and products though which the design, implementation, management, and use of information systems can be efficiently and effectively achieved. Designing such useful *artifacts* is complicated in practice since there is a need for creative advances in domain areas in which the existing theory is often insufficient; as technical knowledge advances, information technology is applied to novel application areas where adaptation of such technologies has not previously been believed amenable. (Hevner et al., 2004)

Hevner et al. (2004) propose a conceptual framework, as illustrated in Figure 4.1), to help understand, execute, and evaluate IS research by combining design science and behavioral science paradigms. In the framework, *environment* conceptualizes the problem space and consists of the people, organizations, and existing technologies. Furthermore, the environment describes the tasks, goals, problems and business needs perceived by the people within the organization. In the context of this study, the environment can be used to conceptualize the tasks and problems of a Omapolku content managers and the organizational need for improving the effectiveness of the system in general. *IS research* combines both behavioral and design science paradigms. Behavioral science is applied to develop and justify theories to explain phenomena related to the business needs. In contrast, design science approaches the research by buiding and evaluating the *artifacts* that were designed as a solution to the business needs. Here, IS research is accomplished by utilizing the *knowledge base* which consists of the foundations and methodologies. Hevner et al. (2004) also make a remark that it is essential to differentiate routine design—i.e., design based on applying existing knowledge to organizational problems—from design research which aims to address unsolved problems in novel and innovative ways. In this study, the approach of designing a novel prototype for Omapolku content managers steers more towards design research, although applying some of the best practices and models in the knowledge base.

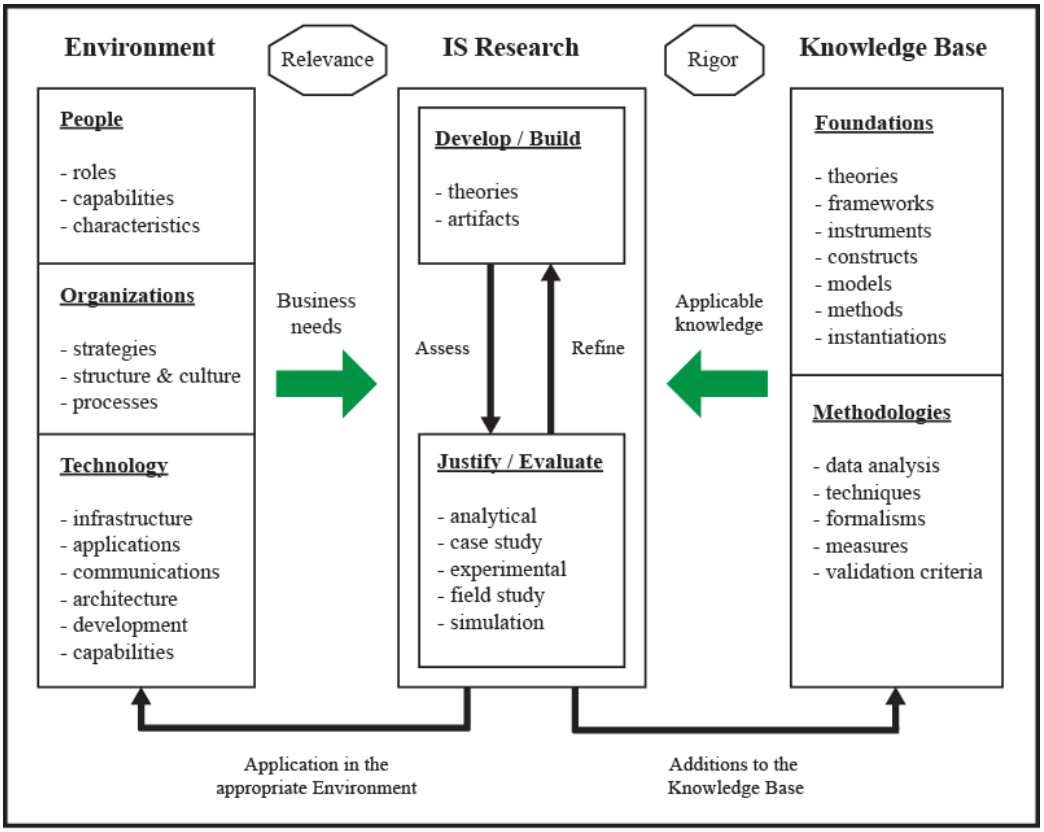

Figure 4.1: Information Systems Research Framework.

4.2 Tools

4.2.1 Piwik PRO

Piwik PRO is an enterprise-level web analytics platform for organizations with high standards for data privacy and security, offering companies a complete ownership over the data. Piwik PRO offers both a cloud-hosted and an on-premises deployment. The on-premises installation allows deploying Piwik PRO on the company's own infrastructure, which helps in mitigating concerns regarding data privacy and security, while allowing full control over the gathered data. Piwik PRO is compilant with international privacy laws, including the General Data Protection Regulation (GDPR) and the Health Insurance Portability and Accountability Act (HIPAA). (Piwik PRO, n.d.) As a side note, considering the introduction of the GDPR on 25th May 2018, sanctions for non-compliance with the regulation have increased significantly (Tankard, 2016). One implication of the considerably increased sanctions is that companies will be driven to utilize data collection solutions that are GDPR-compilant.

4.2.2 ATLAS.ti

ATLAS.ti is a software tool for qualitative data analysis of large quantities of textual, graphical, audio and video data (ATLAS.ti, n.d.). ATLAS.ti provides support for a wide range of different data formats, allowing coding, annotating, linking and visualizing findings with a systematic approach.

4.2.3 Node.js

Node.js is an asynchronous, event driven JavaScript run-time environment designed for building scalable web applications. (Node.js, n.d.) With Node.js, it is possible to run JavaScript code outside of a web browser in a local environment or on a web server. Utilizing Node.js was seen as a particularly convenient approach in this study, because the algorithm used for interaction data analysis was implemented in JavaScript to allow adopting the algorithm into a web application with minimal effort.

4.2.4 Google Charts

Google Charts is a data visualization library for creating interactive graphical charts from user-supplied data (Google Developers, n.d.). Google Charts is

commonly used via JavaScript embedded in a web page. The library supports various chart types depending on the visualization needs.

4.2.5 Visual Studio Code

Visual Studio Code is a lightweight source code editor for desktop computers running Windows, macOS, or Linux operating system. The editor has built-in support for JavaScript, TypeScript and Node.js and it provides various extensions for other languages. (Microsoft, n.d.)

4.3 Data collection

4.3.1 Customer feedback

In order to take into account the current opinions and suggestions for improving the service, existing customer feedback of Health Village was collected from the feedback system provided by Analystica. Analystica is a Finnish company that provides a continuous feedback handling system for collecting and analysing customer feedback data (Analystica, n.d.). Analystica feedback system is implemented and delivered as a web application and is licenced as Software as a Service (SaaS). Most of the feedback data was collected prior to this study with an online feedback form. The feedback form was provided by the Analystica feedback system, functioning as an interface between a user and the feedback system.

For this study, a sample feedback dataset was exported from the system into an Excel file. Each feedback instance is presented as a single row in the Excel file, including of various types of feedback attributes presented as columns in the Excel file. In the Excel file, for each feedback record, the relevant information chosen in this study included

- subject of the feedback,
- type of the feedback,
- and textual feedback content.

The raw data of each feedback instance consisted of 16 attributes, i.e. colums in the Excel file, such as the used device, web browser, contact information, and region. The original file format reference is presented in Appendix A.1. Attributes such as those previously mentioned were omitted since the objective of this study regarding customer feedback is to analyze the

textual content of the feedback. Since the objective of analyzing feedback in this study, as defined in RQ4, is to analyze feedback concerning Omapolku, only feedback instances targeted to Omapolku were collected for analysis in this study. As a result of the filtering, the final feedback dataset utilized in this study consisted of *39* feedback instances.

4.3.2 Interaction data

In order to gain better understanding of how patients interact with Omapolku, user interaction data was collected in the form of raw web session data. The source dataset consists of entries accessible via *Visitor Log* reporting dashboard in Piwik PRO. The source dataset consisted of data from *10* distinct user sessions. Since the focus of this study was not to perform statistical analysis on the session data, a small data sample was seen sufficient for this study since the data represents the *properties* and *level of detail* of the session data. The source data was exported to JSON file format, based on the fact that JSON has great support in major web browsers and is included in the latest ECMAScript standards. For this reason, parsing JSON data with custom algorithms is relatively simple and straight-forward.

In the raw JSON data, each visit was presented as an object including *90* different attributes, including attributes conserning *time*, *referrer*, *geographical location*, *device*, and other attributes used in the domain of e-commerce. The total amount of attributes depends on which type of a report format is used in Piwik PRO. In this study, only attributes relevant to the research goals were included.

```
1 {
2     "actionDetails": object,
3     "actions": <string>
4 }
```

Listing 4.1: A visit object in JSON format

The chosen attributes for this study are presented in Listing 4.1 and their corresponding data types are presented in Table 4.1. As mentioned earlier, the original visit objects contained significantly more information, however, most of the information was not seen relevant in this study.

Attribute	Type	Description
`actionDetails`	object	Array of action objects of those actions performed during the visit
`actions`	string	Total amount of actions performed during the visit. The value equals the amount of actions in `actionDetails`

Table 4.1: Attributes of a visit object in Piwik PRO.

In the data, the attribute `actionDetails` contains `action` objects. In Piwik PRO, an action stands for an event generated by the visitor. In the context of this study, the analysis of the data is based on the data within `actionDetails`, which contains all actions performed by the user during a single session. Listing 4.2 presents the data contained in an `action` object.

```
{
    "type": <string>,
    "url": <string>,
    "pageTitle": <string>,
    "pageIdAction": <string>,
    "idpageview": <string>,
    "serverTimePretty": <string>,
    "pageId": <string>,
    "timeSpent": <string>,
    "timeSpentPretty": <string>,
    "generationTimeMilliseconds": <string>,
    "generationTime": <string>,
    "interactionPosition": <string>,
    "icon": <string>,
    "timestamp": <int>,
    "customDimensions": <object>
}
```

Listing 4.2: A complete action object in JSON format

From the attributes presented in Listing 4.2, attributes `type`, `url`, `idpageview`, `pageId`, `serverTimePretty`, `timeSpentPretty`, `generationTime`, `icon`, `timestamp` and `customDimensions` were omitted due to their redundancy in the context of this study. At first, the `url` was included, but it became obvious that it was constructed using the attribute `pageTitle` in most of the events and was therefore omitted. Furthermore, it seemed that the `url` attribute was formed by appending the `pageTitle` attribute after the domain string. The remaining attributes that were utilized in this study and their descriptions

are presented in Table 4.2. As a side node, the utilized `pageTitle` values are in Finnish in the example dataset.

Attribute	Type	Description
`pageTitle`	string	Page title defined in the HTML `<title>` tag of the page.
`pageIdAction`	string	Unique identifier of the visited page where the action occured.
`timeSpent`	string	Total duration of the action in seconds.
`generationTimeMilliseconds`	string	Total generation time for the page in milliseconds.
`interactionPosition`	string	Position of the action within the action sequence of the visit.

Table 4.2: Utilized attributes of an action object.

4.4 Data analysis

4.4.1 Customer feedback

In this study, the primary interest regarding customer feedback was to analyze the level of detail and actionability of freeform textual feedback concerning Omapolku. At the initial stages of this study, there were not any other sources of information for gaining insight about the current opinions and open issues regarding Omapolku. For this reason, it was intriguing to utilize the information provided by customer feedback as a baseline for understanding the level of detail and actionability of the feedback in general. Furthermore, it was seen worth studying if user interaction data can reveal identical issues that have already been reported via customer feedback. This question was encapsulated as research question RQ4 that was introduced in Section 1.1.

A qualitative research method was chosen to analyze the customer feedback. The goal of qualitative analysis was to derive dominant themes from the feedback by using a *general inductive approach*. According to Thomas (2006), the general inductive approach can be used to derive frequent and dominant themes from detailed readings of raw data. In the context of this study, the goal of utilizing the general inductive approach was to extract the dominant themes and relevant categories from the raw feedback data.

Data pre-processing

The first step prior analyzing the customer feedback was *data pre-processing*. The goal of data pre-processing was to decrease information redundancy and take necessary actions from data privacy point of view by data anonymization. Here, the goal of data anonymization was to discard attributes of Personally Identifiable Information (PII). Such attributes usually include, e.g., *social security number*, *name*, textitphone number and *street address*. After total anonymization, the data pre-processing procedure was performed within the exported Microsoft Excel spreadsheet, including

- discarding columns including irrelevant feedback attributes,
- discarding empty and erronous feedback,
- discarding ambiguous and unactionable feedback.

Data coding

In order to categorize and draw conclusions from the feedback, the pre-processed feedback text was exported to ATLAS.ti. ATLAS.ti was chosen for feeback processing due to its efficient workflow which provides tools for efficient data *coding*. The first step after importing the text was to visually refactor the text into segments where each feedback instance is separated by two empty lines. This was seen helpful in order to differentiate the feedback texts from each other. Now, in order to categorize the issues reported in the feedback, each feedback instance was coded with a descriptive category code, such as "Media playback". Building the coding scheme, i.e., deciding the level of granularity of the categories was not straightforward since the coding scheme is always case-dependent. Since the feedback concerns technical issues, the goal was to categorize such issues in a meaningful way and in sufficient detail. In the beginning, the feedback was run though a *word cloud* functionality which, however, did not provide actionable results since the term frequencies were insufficient. Table 4.3 presents the coding scheme including only the relevant codes concerning technical issues.

Code	**Description**
Responsiveness	Issues with application responsiveness.
Media playback	Issues with audio and video playback.
Navigation	Issues with navigation in the application.
Exercise task	Issues with exercise tasks.

Table 4.3: ATLAS.ti coding scheme for feedback data.

For each feedback text, a certain code was used only once in order to prevent bias in the statistics. For example, if a certain keyword or a conceptually similar term or sentence appeared several times in the feedback data, it was coded only once.

4.4.2 Interaction data

An essential part of this study is to analyze user interaction by utilizing statistical techniques of web usage mining introduced in Section 3.3. The objective of the analysis was to understand how patients navigate in digital treatment paths and how much time is spent in performing different tasks. In order to accomplish this, the sampe dataset collected with Piwik PRO was analyzed by using a proprietary script designed and developed for this study. The script was implemented in JavaScript on Node.js using the EcmaScript 6 (ES6) standard. The algorithm was executed on a local development environment using Node.js, which allows rapid prototyping and code execution without a web browser. The implemented script is presented in Appendix C.

Data pre-processing

The first step of processing the raw session dataset was to remove all identifying attributes from the data. Here, two attributes, `visitIp` and `visitorId`, were discarded from each visit object. The IP-addresses were removed with a "Replace" function in Visual Studio Code with the following regular expression (RegEx):

```
"visitIp":"([0-9]{1,3}[ .]){3}[0-9]{1,3}"
```

Likewise, the visitor identifier, `visitorId`, was removed with a RegEx as follows:

```
"visitorId":"[a-Z0-9]{16}"
```

In order to remove the IP-addresses from the source data, the patterns found with the RegExes were then replaced with blank characters. The purpose of this was to both consider data privacy and to reduce redundant information. Removing the IP-address introduces a possible issue with identifying unique visitors since `visitorId` is based on a browser cookie and/or the fingerprint obtained from the IP-address and device information. In the context of this study, this issue is not relevant because sessions are analyzed

from interaction point of view. Here, lacking `visitIp` and `visitorId` do not prevent the analysis of the data since each visit in the JSON file is unique and structured such that each visit is contained in a separate JSON object. In this study, the essential task of session identification, as presented in Section 3.3, was straightforward since Piwik PRO automatically groups the distinct sessions as separated JSON objects, as previously mentioned.

General interaction metrics

Several basic metrics were calculated from the source data using statistical techniques of web usage mining in order to better understand the nature and proportion of each visit. The following metrics were extracted for each visit instance from the sample data using the custom script:

- visit duration
- total actions
- mean time per action
- unique pages
- mean page generation time
- standard deviation of page generation time

The mean time per action was calculated using the following standard equation of an arithmetic mean:

$$\bar{x} = \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right) = \frac{x_1 + x_2 + \cdots + x_n}{n} \tag{4.1}$$

The purpose of calculating the arithmetic mean of page generation time was to get an overview of how responsive the web application is. Furthermore, prior to deeper analysis, it became evident that in many recorded visits, the page generation time was equal between all actions and was worth further investigation. For this reason, standard deviation was used for calculating the variation of generation times using the following standard equation:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \overline{x})^2}{n}} \tag{4.2}$$

In addition, standard deviation was used to help find those visits that have this issue, which could possibly be caused by an issue with the Piwik PRO configuration. In both of the equations, x_i is the i:th value, $\overline{x}$ is the mean, and n is the number of values.

Path analysis

In order to better understand the relationships and the path of the actions, it was useful to visualize the performed actions in a way that would instantly show the order of actions and the time spent per action. As a result, a timeline visualization was chosen to illustrate the sequence and length of the individual actions performed during a visit. The timeline visualization was implemented with Google Charts using a script presented in Appendix D.

The main objective of path analysis in this study was to find a way to *link* the metrics extracted from the interaction data to the individual front-end components of treatment paths in Omapolku. The motivation behind this is that in order to truly understand what the data suggests, the data must be directly *identifiable*. For instance, if an action object recorded by Piwik PRO does not contain the necessary *context* of that action—i.e., in which treatment path the action occured and in which task, it is practically impossible to draw any conclusions from the data. By linking the data to individual front-end components in a treatment pathway interface, it is possible to gain knowledge which tasks seem to take more time than others, where the visits end, for how long content videos are being watched, and are content hyperlinks clicked. With this information, it is possible to prioritize and refine the current implementation both in the way of information structuring and presentation. For instance, if it seems that content videos are not very popular—considering that planning, scripting, recording, editing, and producing a video takes time and introduces expenses—it is worth thinking if videos are an efficient way to deliver relevant information. In addition, if certain tasks take significantly longer that other tasks, it is easier to study if there is a need for refactoring the information into smaller pieces. These examples are just a part of the possibilities that can be done by analyzing the interaction paths from the clickstream data.

Chapter 5

Results and analysis

This chapter presents the results of the suggested web analytics metrics and both the analyzed customer feedback and user interaction data that were processed according to the methods described in Section 4.4. First, Section 5.1 presents the key findings of the customer feedback. Section 5.2 presents the general web analytics metrics that can be used to analyze user interaction data of digital treatment paths in a general level. Finally, Section 5.3 introduces the results of the path analysis of interaction data that was processed and analyzed with the implemented script.

5.1 Customer feedback

Already during the first steps of analyzing the feedback data, it became evident that the level of detail of the information provided in the feedback text is not sufficient in most cases in terms of actionability. In practice, this introduces significant issues in feedback handling and actionability since the feedback has to be processed manually by a person, while trying to understand what issue the feedback practically concerns. Practical examples of this phenomenon include the following feedback instances that were translated from the original Finnish versions presented in Appendix B:

(1) *"I am using only ipad and iphone, the program does not work!"*

(2) *"Jamming all the time, and does not open programs, no sound in the films and really slow if it even does something"*

(3) *"[...] Whole service is jamming, regardless of what you do. [...]"*

Utilizing feedback with a high level of ambiguousness like in the previous three examples is not actionable to any extent. The only concusions that can be made from the example feedback is that there could be an issue with the service when using an iPad or iPhone. Since there are no details provided about the problem in the example case, it is practically impossible to process the feedback further for making improvements. Regardless of the fact that these examples do not seem provide very useful information due to lacking detail, the overarching theme here is that there is most likely a responsivity issue in the web application implementation.

Code	**Grounded**	**% of all feedback**
Responsiveness	6	**15.4**
Media playback	4	**10.3**
Navigation	2	5.1
Exercise task	1	2.6

Table 5.1: Coding results of feedback data.

Table 5.1 presents the results of the coded feedback. Here, *grounded* equals the frequency, i.e., how many feedback instances were coded with the corresponding code. The column *% of all feedback* presents the percentage of feedback that were coded with the corresponding code with respect to the total amount of 39 feedback instances in the source data. According to the results, it seems that the frequencies of *relevant* feedback are relatively low with respect to the objectives of this study. Furthermore, it seems that technical feedback in the sample dataset regarding Omapolku does not provide very granular and actionable information about the reported issues. However, regardless of the lack of detail, the results show that the most reported issues concern *responsiveness* and *navigation* of the web application. Interestingly, these issues can be examined from the interaction data. For instance, navigational issues can be extracted from the clickstream data by finding "circling" and "swapping" sessions as discussed in Section 3.3.

5.2 General metrics

General metrics include metrics for both analyzing visitor engagement and measuring the usability and user experience of the Omapolku web application in a general level. Table 5.2 presents such metrics and their descriptions that can be used to measure visitor engagement of patients' digital treatment paths.

ID	Metric	Description
A1	Pages per visit	Total amount pages that were visited during a browser session.
A2	Visits	Total amount of visits on a certain digital treatment path.
A3	Users	Total amount of patient users that have visited a certain digital treatment path.
A4	Avg. visit duration	The average duration of a visit on a certain digital treatment path.
A5	Avg. page load time	The average duration after which the page is completely loaded from the server.
A6	Error pages	Page requests that resulted an error (i.e., HTTP 404).
A7	Most popular pages	Pages that have been visited the most.
A8	Exit pages	Pages to which the broweser session was ended (i.e., pages that were last in the sequence of visited pages).
A9	Exit rate	Percentage of visits that ended the browser session on a certain page.
A10	Bounce rate	Percentage of visits that started and ended the browser session on a certain page.

Table 5.2: Results of general web analytics metrics.

Most of the metrics presented in Table 5.2 are very common in web analytics literature, as introduced in Section 3.2. In fact, many of the commonly used web analytics platforms, such as Piwik PRO and Google Analytics, provide ready-made tools for reporting basic metrics including A1–A4 and A10. Figures 5.1 and 5.2 present examples of such dashboards in Piwik PRO and Google Analytics, showing some of the basic metrics that were listed in Table 5.2.

When assessing the reliability of metrics A1, A4 and A5 in a single-page application like Omapolku, there is a possibility that these metrics do not necessarily provide expected results unless correctly configured in the Piwik PRO *JavaScript tag* used for data collection. In contrast to traditional multi-page websites and applications, single-page applications do not necessarily trigger *page loads* other than the first page load. The problem with web analytics platforms like Piwik PRO is that they commonly rely on new page load events. Here, in the case of single-page applications, the analytics platform does not automatically know when a new page is viewed since

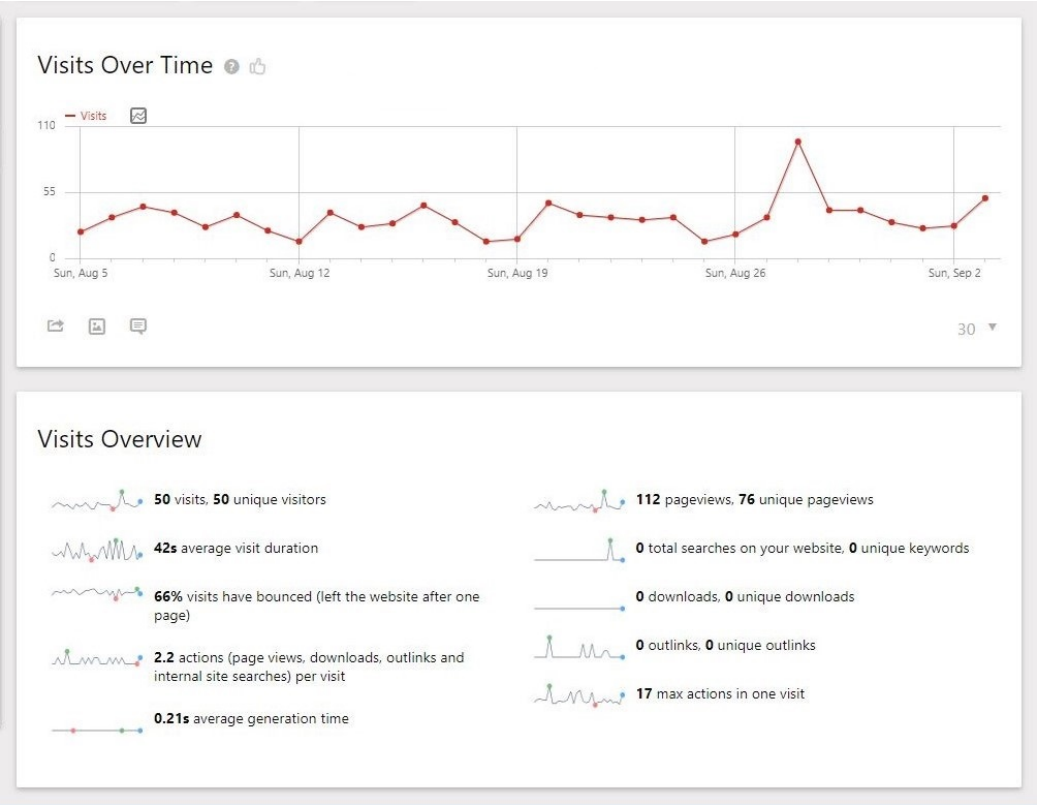

Figure 5.1: Basic metrics in example Piwik PRO dashboard.

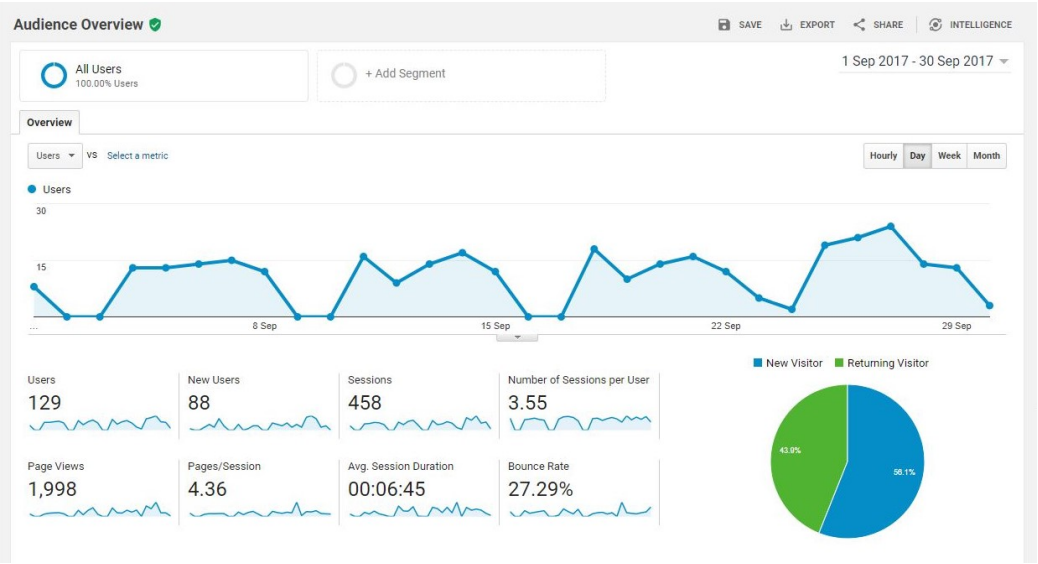

Figure 5.2: Basic metrics in example Google Analytics dashboard.

the page data is fetched from the server using AJAX. For this reason, extra attention should be paid to dealing with metrics regarding *pages* and *page loads* in single-page applications. In other words, since the user interaction occurs on the same *physical* page, but the contents of that particular page are dynamically requested from the server and the corresponding front-end components are updated and re-drawn, a *page load* event is not triggered. For this reason, it is crucial to configure the *JavaScript tag* of Piwik PRO to know when a new page has been visited. In this case, all tracking should be configured by utilizing *Virtual Page View* tracking tags in Piwik PRO Tag Manager. This issue will be discussed more in Section 5.3.

In addition to basic, page-level metrics, proprietary *content metrics* can be used to measure the usage of the content in digital treatment paths. Such content includes various front-end elements, e.g., buttons, videos, and hyperlinks in the digital treatment pathway view. Most importantly, content metrics regarding the treatment pathway *task* components, as illustrated in Figure 5.3, can be used to gain insight of how the path is navigated. Table 5.3 presents several content metrics and their descriptions that can be used for measuring the interaction in patients' digital treatment paths in a more granular level.

ID	Metric	Description
B1	Avg. task completion time	Average time of completing a task.
B2	Task exit rate	Percentage of sessions that ended on a certain task.
B3	Avg. viewing duration of videos in a task	Average viewing time of task-specific videos of completed tasks.
B4	Avg. path completion time	Average time from first path visit to path completion.

Table 5.3: Results of digital treatment path metrics.

Metrics B1–B4 can provide most of the key insight that can be extracted from the usage data. However, in order to implement these metrics in practice, the interaction data must be *identifiable*, i.e., the front-end component and the data should contain the same unique identifiers to allow linking the data to the corresponding front-end components. Otherwise, it is impossible to implement these metrics. This issue will be further analysed in Section 5.3.

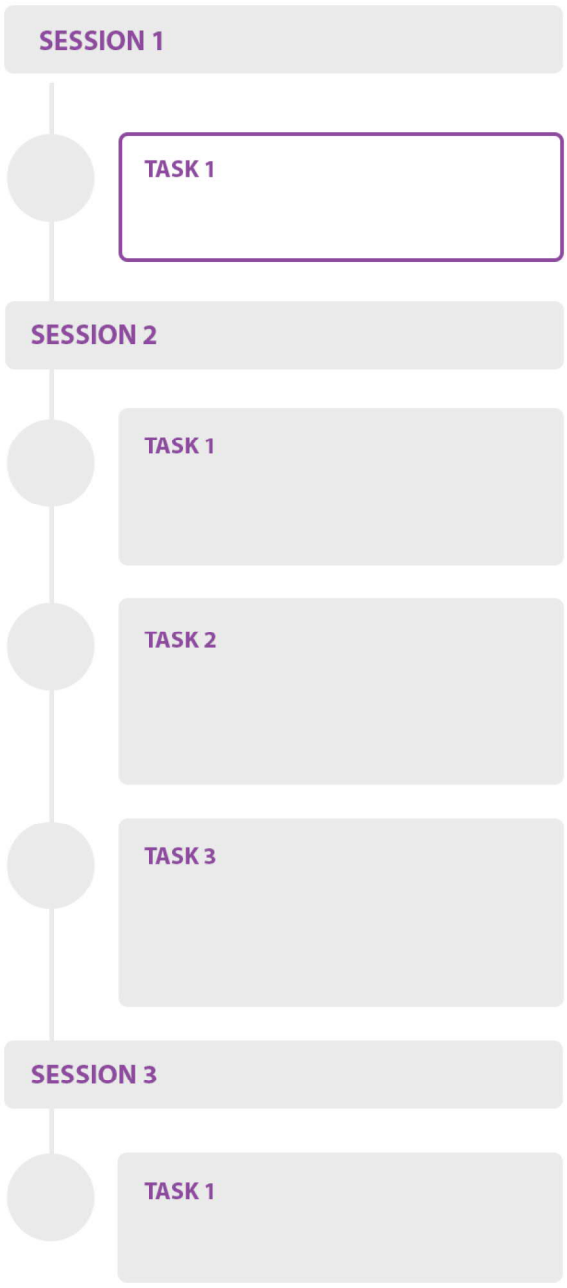

Figure 5.3: Front-end layout illustration of a digital treatment pathway. Here, the highlighted front-end component is a single task grouped under a session component in the treatment pathway.

5.3 Interaction data

5.3.1 General interaction metrics

Table 5.4 presents general metrics of visits derived from the interaction data according to the analysis process presented in Section 4.4. The results show significant variation in *mean time per action* between individual sessions. Furthermore, there does not seem to be any correlation between the total amount of actions performed during the visit and the visit duration. To some extent, this is understandable and expected considering human factors. For instance, when comparing the mean time per action between `Visit 2` and `Visit 6`, the latter with only *7* actions has a significantly greater mean time per action of *65.7* seconds, compared the former with *49* actions and *42.1* seconds mean time per action.

Although the sample dataset is not large enough for a valid statistical analysis—which was not considered relevant in this study—the results introduce a practical issue in web analytics; user behaviour is not predictable and safe assumptions cannot be made from the data of individual visits. Here, it is possible, for instance, that the visitor of `Visit 6` was either doing something else at the same time or had encountered a piece of content that took significantly longer to complete. For this reason, it is useful to analyze the visit in detail by examining and visualizing the visit path.

Visit #	Actions	Duration	Mean time per action (s)
1	7	35s	5.0
2	**49**	34m 22s	**42.1**
3	21	30m 37s	87.5
4	42	11m 41s	16.7
5	70	27m 10s	23.3
6	**7**	7m 40s	**65.7**
7	6	1m 1s	10.2
8	11	2m 27s	13.4
9	20	4m 59s	14.9
10	51	39m 24s	46.4

Table 5.4: General metrics calculated from sample data.

The results show that the current configuration of Piwik PRO may have an issue regarding the calculation of `generationTimeMilliseconds` value for an action performed during a visit. Table 5.5 presents the mean page

generation times and corresponding standard deviations (SD).

Visit #	**Unique pages**	**Mean generation time (ms)**	**SD of generation time (ms)**
1	2	147.0	**0.0**
2	23	92.8	11.2
3	13	121.1	14.1
4	19	232.9	11.3
5	16	130.0	**0.0**
6	6	279.0	**0.0**
7	4	102.0	**0.0**
8	7	136.0	**0.0**
9	13	75.0	**0.0**
10	28	209.8	79.2

Table 5.5: Page generation time metrics calculated from sample data.

Here, the main goal of using standard deviation was to help revealing such visits with zero variability between the page generation times of all actions performed during the visit. Surprisingly, 6 out of 10 visits, i.e., 60% have a standard deviation of 0. When standard deviation equals 0, it implicates that the values are all equal. The practical implication of this is that in those 6 cases, Piwik PRO used the same page generation time for all subsequent events. The most likely reason is that since Omapolku is a single-page appication, the tracking script did not send an updated page generation time for each of the subsequent actions. With the current implementation of the tracking script, the `generationTimeMilliseconds` provided by Piwik PRO is not necessarily reliable and it should not be used solely for determining page performance. However, in theory, it is possible that all of those requests are processed and rendered in the same time, but the probability for that is practically nonexistent considering the asynchronous nature of the requests.

Considering the results regarding *unique pages*, it became evident that in relatively many actions recorded as *virtual page views*, the attribute `pageTitle` has a value of *null*. The implication of this is that there is no way to determine which page was visited. In addition, the issue causes bias in the calculation of unique pages since all actions with `pageTitle` of *null* are considered as one.

According to the results of the feedback analysis, it became obvious that issues regarding *responsiveness* and *navigation* were reported frequently in

Omapolku feedback. Reflecting that on the results of the general metrics privously presented, the average page generations times in the sample dataset fall under the threshold suggested by Nielsen (1999); a respond time less than *one second* is sufficient for not distracting the user. However, considering the small sample size of the dataset utilized in this study, it is not safe to draw reliable conclusions that all other visits fall constantly unders the suggested response threshold.

5.3.2 Path analysis

In order to illustrate the issues with the collected interaction data, four different examples of click paths were analyzed and visualized. In the visualizations, each timeline block in the sequence is labelled with the `pageTitle` attribute of the action where the length of the block represents the time spent for completing the action, i.e., viewing the content on that page. The exact values of the data used to visualize the example timelines is presented in Appendix E.

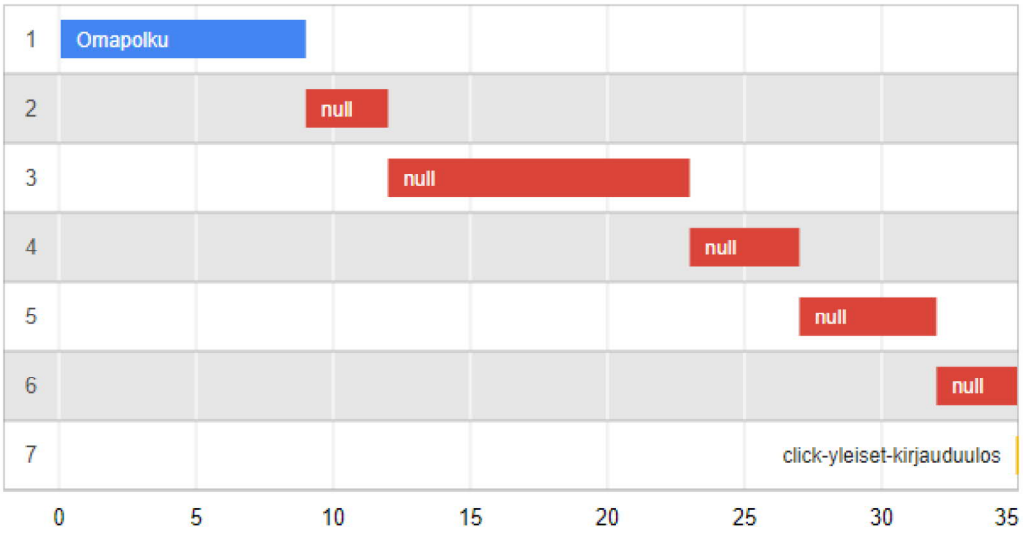

Figure 5.4: Visit 1, complete path timeline consisting of 7 actions performed within 0m 35s.

Figure 5.4 presents an example of a complete click path perfomed within one user session, consisting of 7 actions performed by the user. In the figure, each row represents a single click interaction in the front-end of Omapolku.

In this example, the click path from the main page (row 1) to the logout page (row 7) includes actions with a `pageTitle` value of *null* (rows 2–6). As a result, it is impossible to tell which page the user was viewing during the five consecutive actions performed between the entry and exit pages. In other words, it is impossible to know what the user was clicking. Here, it is possible that the click was performed to a virtual page with no defined `pageTitle`. With this information, a definite answer to this question cannot be given, although that is what the results suggest by looking at the time difference between the clicks.

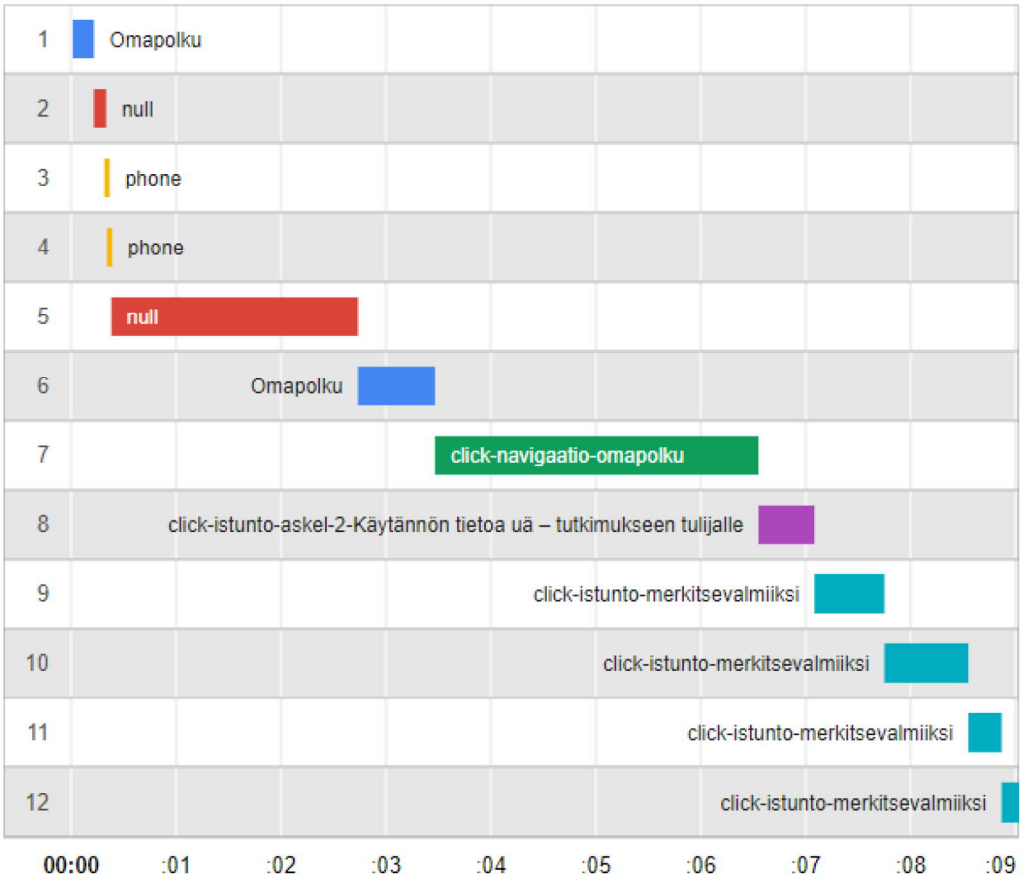

Figure 5.5: Visit 2, partial path timeline including first 12 actions performed within 9m 2s.

Figure 5.5 presents an example of a partial click path extracted from a longer visit consisting of 49 actions. In this example, the user navigated to the treatment path view after six clicks. Rows 7–12 show that the user clicked on the second task (labelled with `click-istunto-askel-2...`) of a session, after which the user clicked four times on a button to mark the session complete (rows 9–12, labelled with `click-istunto-merkitsevalmiiksi`). The problem here is that it is not possible to be sure if the user clicked the same button four times since the `pageTitle` does not include an exact identifier that could linked to the corresponding front-end component. Like in the previous example, there is a clear issue with the lacking *context* of the action, i.e., identification of the actions is not possible for many actions.

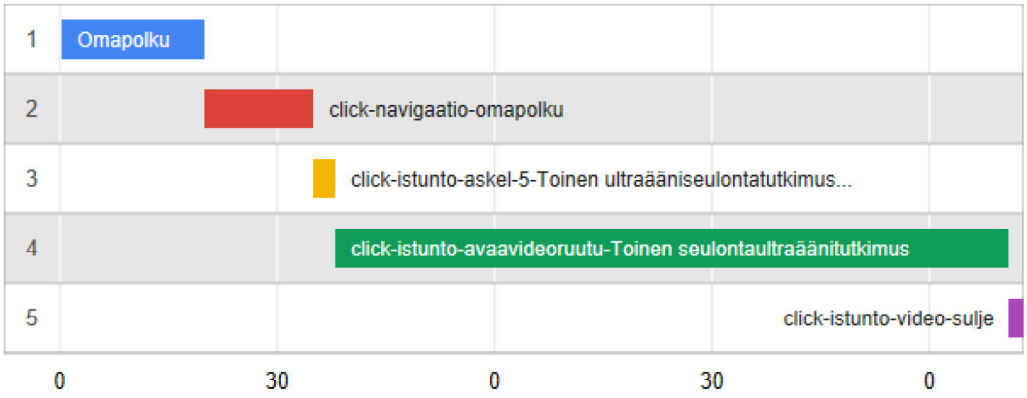

Figure 5.6: Visit 9, partial path timeline including first 5 actions performed within 2m 13s.

Figure 5.6 presents another example of a partial click path consisting of first five actions from the total of 20 actions. In this particular example, the interaction data provides particularly useful information that can be used to calculate the average viewing duration of a content videos, that is, metric B3 presented in Table 5.3. In this example, the results show that the video was viewed a total of *1m 33s*. However, the only way to identify which video was watched is to use the value of the `pageTitle` attribute of that virtual page. Regardless of this issue, the results show that it is possible to extract this information from the interaction data.

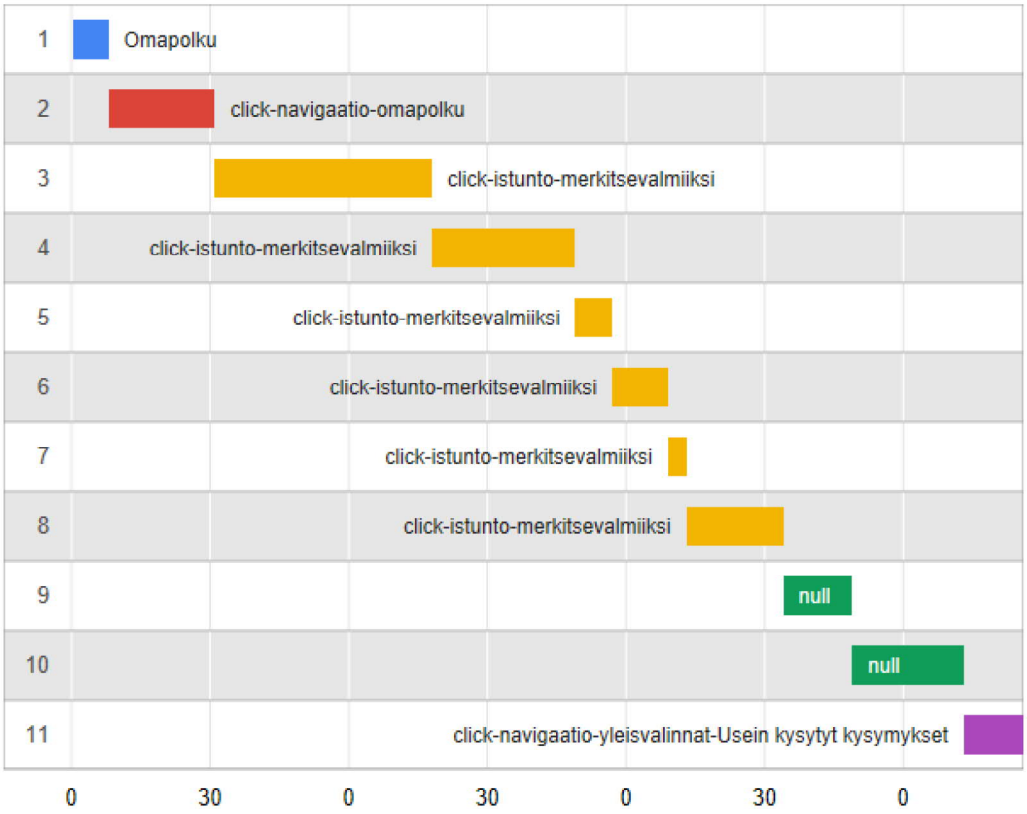

Figure 5.7: Visit 10, partial path timeline including first 11 actions performed within 3m 26s.

Finally, Figure 5.7 presents final example of a partial click path consisting of first 11 actions of the total of 51 actions. In this example, the concept of *swapping*, as discussed in 3.3, becomes evident. Here, the user first completes tasks (rows 3–8, labelled with `click-istunto-merkitsevalmiiksi`), after which the user jumps to a *frequently asked questions* page (row 11). However, since the two previous actions (rows 9–10) have a `pageTitle` of *null*, it is not possible to determine the exact page that caused the user to swap to the page.

Chapter 6

Solution to identified challenges: the prototype

This chapter presents prototype of a novel front-end component for Omapolku content managers. The prototype is proposed as a viable solution to the identified challenges presented in Chapter 5. First, Section 6.1 introduces the process of content management of treatment paths in Omapolku. Section 6.2 provides an overview of the prototype implementation and motivation for implementing a novel solution to support the work of content managers with the help of interaction data. Section 6.3 describes and illustrates the design of the prototype. Finally, Section 6.4 presents the main functional requirements of the prototype.

6.1 Introduction

The primary task of Omapolku content managers is to implement treatment pathways by utilizing a proprietary tool implemented for content managers. Implementing a digital treatment path consists of

- planning and choosing the relevant content to be published,
- building a hierarchical *session–task* structure,
- adding the chosen content (e.g., text, images, videos, questionnaires, external links) into the session–task structure,
- publishing the path.

Since the tools provided for content managers are fairly flexible in terms of usage, there are many ways to use the provided tools. For instance, implementing a session–task structure is not bound to certain structural decisions

and therefore it can be utilized in many different ways. Furthermore, there are many ways to present the chosen information. As an example, an external link to other sources, such as web pages including supporting information concerning a treatment or other useful information, can be presented either as a proprietary hyperlink content component or as a traditional hyperlink embedded into the text content. Such decisions can eventually have an effect on the user experience of the chosen implementation.

Currently, there are no simple means for content managers to receive practical feedback and insight of how the published content is perceived and experienced, apart from textual customer feedback. In order to continuously improve the efficiency and effectiveness of content management, along with better user experience, there is a clear need for a novel solution that can provide the necessary information more conveniently.

6.2 Overview

The purpose of the prototype is to provide a solution to overcoming the previously discussed limitations. According to the results presented in Chapter 5, the current way of utilizing web analytics data does not provide sufficient insight of the usage of digital treatment paths in a format that could be directly adopted to support content management. The prototype aims to provide actionable insight for content managers in a significantly more understandable format to support the continuous improvement of the existing implementations of treatment pathways. Utilizing web analytics data directly within the same context of the treatment path front-end implementation can help content managers to understand how patients interact with the provided content. Since the decisions regarding the content and implementation of each digital treatment path are made by a designated content manager, there is a possibility that certain implementations may introduce issues in usability and user experience. By utilizing the information provided by interaction data, it is be possible to discover some these issues—issues that could probably otherwise be left unaccounted for.

Currently, the approach of utilizing interaction data is cumbersome since the data needs to be manually analyzed for extracting the necessary details of interaction events. The process of manually analyzing large datasets with this level of granularity in real use cases is time-consuming and requires sufficient domain knowledge. In practice, this can be an issue. The current process of utilizing the data is neither optimal nor efficient in the long term. Since the standard reports of page-level metrics available in Piwik PRO do not provide sufficient detail for analyzing interaction in treatment pathways,

the *Visitor Log* in particular which provides most detailed information about visits, it is evident that the configuration of Piwik PRO should be improved to allow better identification of visit actions with respect to the front-end components of treatment pathways.

Given that the identification of individual actions is possible and that the actions can be linked to the front-end components of the treatment pathway, it is possible to implement a proprietary analytics component that displays the relevant component-level metrics in the same graphical interface. Figure 6.1 presents the ideology of the prototype. Both the design and general functionality of the prototype is based on the existing design of the front-end components and general navigation patterns of Omapolku.

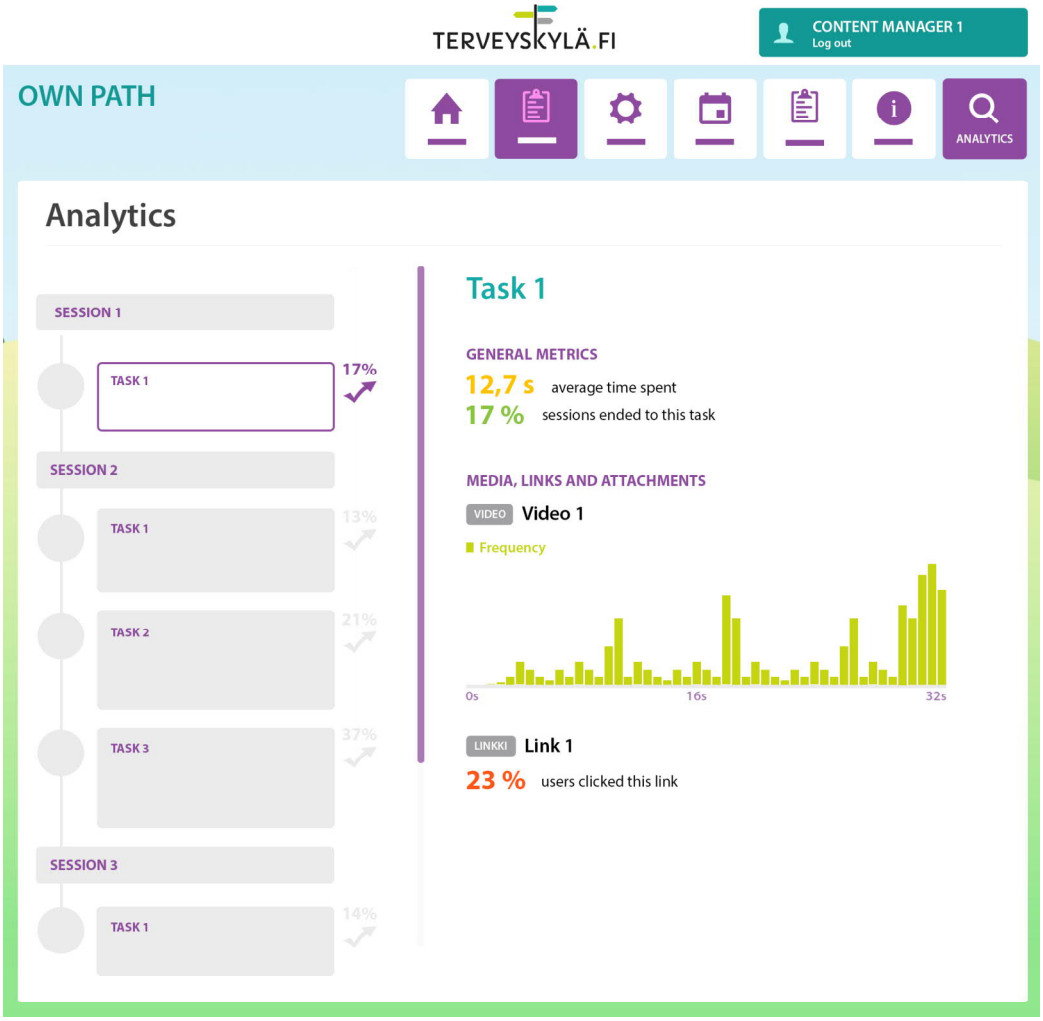

Figure 6.1: Prototype layout for content managers' analytics view.

6.3 Design

6.3.1 Navigation

From the software architectural point of view, the analytics component is a separate front-end view. This implementation is convenient for many reasons, including:

- **Modularity**. The component can be developed and tested separately from the main application. This is convenient and supports the implementation of a component-based service architecture.
- **User experience**. The component does not introduce significant changes to the main design of the application. The component offers useful information, but using it is optional and does not introduce changes in the basic workflow of a content manager.
- **Access control**. Only an authenticated content manager user is be able to access the component.

In order to navigate to the analytics view, a new navigation button will be added to the navigation panel of Omapolku, as presented in Figure 6.2. The navigation button is visible only for Omapolku users that are authorized for content management.

Figure 6.2: Navigation button to access the analytics view.

6.3.2 Task-specific performance metrics

The fundamental idea of the prototype is to integrate the collected interaction data dynamically to the context of Omapolku. This requires that the interaction data contains the necessary identifiers and context so that the data can be linked to individual front-end components of the prototype treatment pathway view. Figure 6.3 presents a proposed desing. Here, when

the content manager clicks any task component in the left side pathway, the content area will be updated with the data including general metrics relevant to the chosen task.

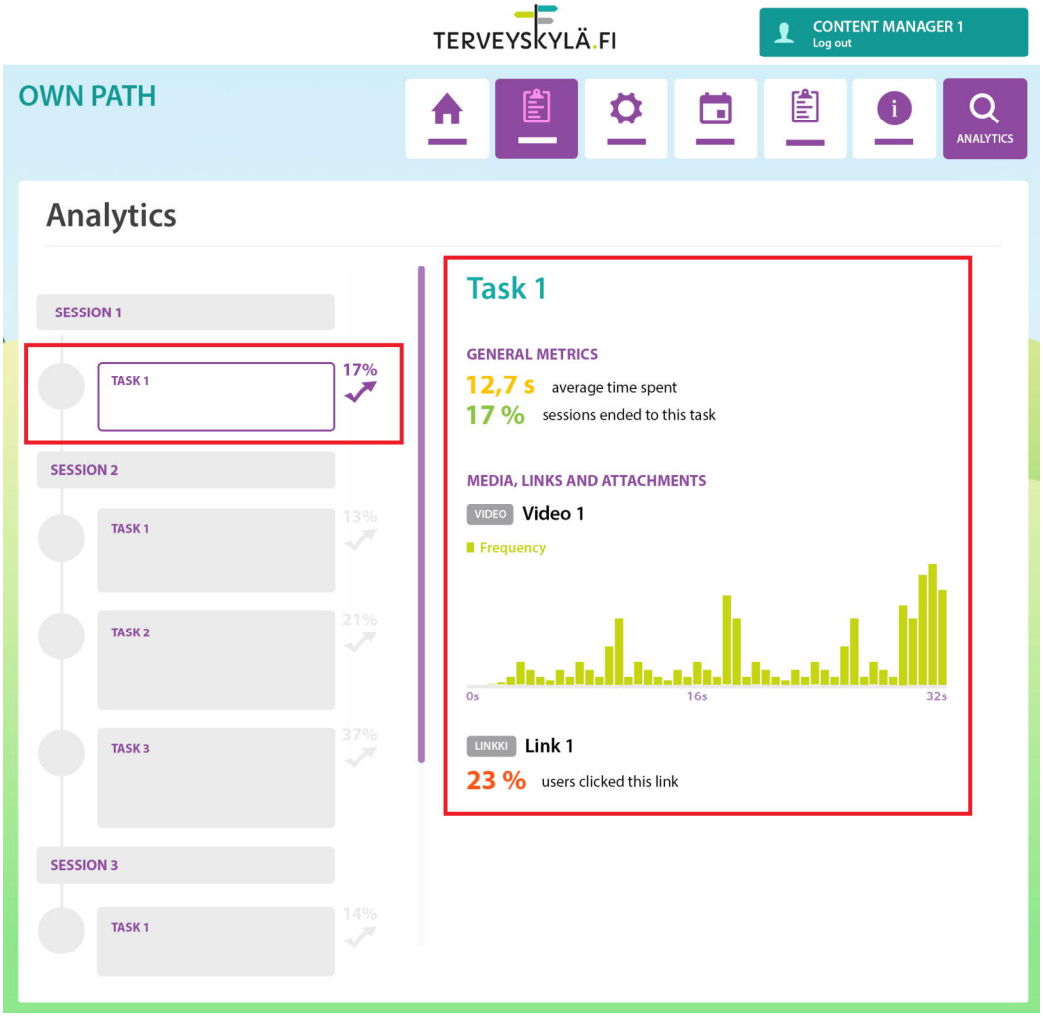

Figure 6.3: Task-specific metrics in the content managers' analytics view.

Figure 6.4 presents the task-specific component in the treatment pathway structure. Here, the value on the right side of the element presents the average *exit rate* of that task. In practice, the purpose of showing exit rate is show how many sessions where ended on each task on average. This information can reveal possible issues with dysfunctional content. Dysfunctional

content can include, e.g., tasks with too much content, erronous content, or other techical issues.

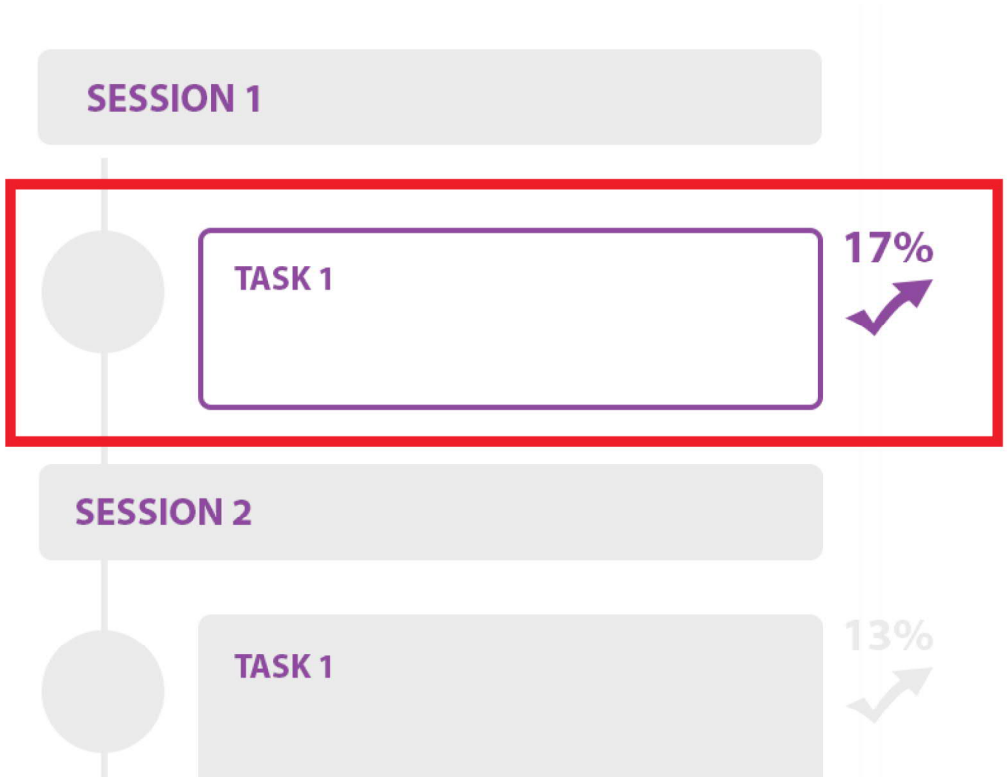

Figure 6.4: Task component in the pathway of the analytics view.

Figure 6.5 presents the task-specific general metrics and relevant information about the task completion, including metrics concerning the usage of content videos and hyperlinks. The metrics displayed in this component depend on the task content; if no videos and/or hyperlinks exist in the task, metrics regarding these are not shown. Metrics displayed by the task-specific component include

- average time spent in the task,
- exit rate, i.e., percentage of sessions that were ended to the task,
- histogram of viewing times of a certain video,
- perentage of users that clicked a certain link.

These metrics can provide useful, baseline information for Omapolku content managers about the functionality of a certain task. The idea behind the prototype is to provide an implementation which allows presenting also other relevant metrics within the same component. The long-term objective of the prototype is to show metrics of all content that have been configured to collect interaction data. For instance, if a new hyperlink is added to a task, the task-specific analytics can automatically show the corresponding metrics of the hyperlink without a need for front-end code manipulation.

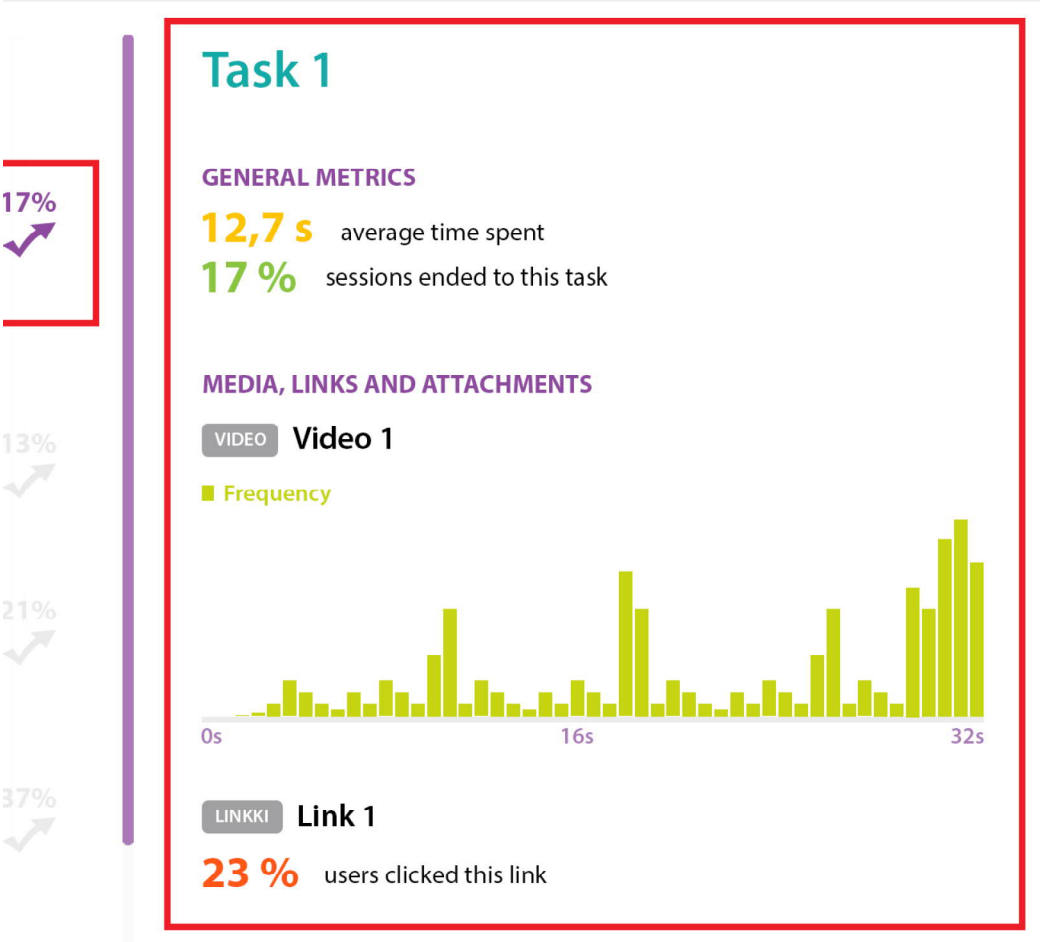

Figure 6.5: Content metrics component of the analytics view. The component includes general task-specific metrics in addition to metrics concerning the task contents, e.g., videos and hyperlinks.

6.4 Functional requirements

User story
As a content manager, I can see a button labeled `Analytics` in the navigation bar after having navigated to any digital treatment path view.
As a content manager, I can click a button labeled with `Analytics` in the navigation bar after having navigated to any digital treatment path.
As a content manager, I can see an analytics component of the current digital treatment path after having clicked the a button labeled with `Analytics` in the navigation bar.
As a content manager, after having clicked a button labeled with `Analytics`, I can see task-specific exit rate visualization components on the right side of a each task component.
As a content manager, after having clicked a task component in the patway structure, I can see task-specific metrics in the content area of the analytics component.

Table 6.1: Essential functional requirements of the prototype.

The essential functional requirements for implementing the basic structure and logic of the front-end of the prototype are presented as user stories in Table 6.1. More detailed user stories are not included in this study, since many decisions regarding the granular functionality of the prototype require further planning and evaluation.

Chapter 7

Discussion

In this chapter, the results presented in Chapter 5 are discussed and evaluated. First, Section 7.1 assesses the legibility of the presented results, followed by discussion on the implications of the results in Section 7.2. Section 7.3 reflects on the proposed prototype that was designed as a solution to the issues and limitations discovered in the analysis and to support the content management process of Omapolku. Finally, Section 7.4 provides answers to the four research questions of this study.

7.1 Reliability of the results

When assessing the legibility of the results presented in this study, it is essential to take into consideration that the results were based on statistically insufficient amount of data. The reason behind this, as stated in Section 4.3, is that the objective of this study was not to perform statistical analysis as such. Rather, a fundamental part of this study was to analyze if interaction data can provide information in sufficient detail to help understand user interaction and if that information can be refined in a way that can be used to spot dysfunctional content in the application. In addition, the study was based on the *design science* approach with a focus in providing a novel solution to an existing problem.

Regarding the customer feedback, conclusions were based on a very limited amount of feedback. For this reason, it is not safe to conclude that customer feedback can not provide enough detail and actionability to help support further development of Omapolku and other applications in general. Furthermore, it is possible that some feedback was unintentionally omitted during the data collection provided that the feedback was not targeted correctly to Omapolku. In other words, it is plausible that more feedback could

have been analyzed. This, however, was noticed and taken into account during the data collection phase.

Assessing the legibility of the results regarding analyzed the interaction data, it is clear that the dataset was statistically nonexistent. However, similarly like in the case of customer feedback, only a minimal dataset was chosen for analyzing the *sufficiency* and *level of detail* of the data in general. Significantly more data could have been utilized in the analysis, however, the utilized data represents the characteristics and level of detail of the interaction data. Therefore, the small dataset was seen suitable in this study.

Overall, the results comply with the research objectives of this study and provide a great baseline for further development of interaction analytics in Omapolku. However, it should be noted that since information about the web application architecture, i.e., code base and application logic, was not conveniently available for this study due to various factors, some assumptions were made regarding the Piwik PRO configuration. For instance, it was assumed that the `pageTitle` attribute of the visit actions contained the title of a *virtual page*—a technique used to track interaction in single-page applications. Eventually, it was left unclear which value was recorded to the `pageTitle` attribute. Assuming that this assumption turns out to be invalid, it would have an effect on the analysis of the results.

7.2 Implications of the results

As mentioned before, the primary objective of this study was to analyze if web analytics tools and methods could provide sufficient information and if the information can be refined in a way to allow spotting dysfunctional content. The results suggest that the current configuration of the Piwik PRO analytics platform does not record interaction events in sufficient detail. In practice, for most visit actions in the dataset, `pageTitle` is the only attribute that can be used to identify which element was clicked by the user. Furthermore, the results suggest that there are situations where the `pageTitle` does not record a value at all, resulting in an ambiguous and practically unactionable *null* value. In simple terms, it is difficult and even impossible in some cases to know where an action happened and what front-end component was being clicked. As a recommendation, the *context* and an appropriate *identifier* should be recorded to visit actions. With this modification, it is possible to directly link the interaction data with the corresponding front-end components.

Assessing the results of the general metrics of interaction data, the results show that the visit duration does not directly correlate with the number of

actions performed. Here, the recommendation is that the visits should be analyzed action by action since the general metrics do not provide reliable information for determining if a treatment pathway has dysfunctional content. It should be noted, however, that using a significantly larger dataset could improve the reliability of the results regarding the general metrics.

Overall, the results show that the interaction data collected with Piwik PRO can provide useful information for understanding user behaviour, given that the context is recorded to each visit action. However, already with the current implementation, it was possible to check how long a content video was watched. In addition, it was possible to spot a visit that *swapped* from a treatment path to the frequently asked questions page. Given the very limited dataset, these results already provide a great baseline, proving that interesting patterns can be extracted from the interaction data. However, to further improve the implementation, capturing the context of the action requires adding additional identifiers to the front-end components via, e.g., `data-*` attributes introduced in HTML5 standard. For instance, given an example treatment pathway including a task element labeled as `Task 1` grouped under a session element labeled as `Session 1`. Here, for instance, the `Task 1` component is added with `data-pathId='...'`, `data-sessionId='...'`, and `data-taskId='...'` attributes that can be used to identify and trace the component in the front-end of the web application. In practice, though, the only requirement is that both the front-end component and the corresponding visit action object in Piwik PRO data use the same unique identifier, such as a unique `id` attribute. There are several different approaches to solving the identification and data linkin issue. In the described example, the `data-*` attributes can contain an arbitrary value given that it is recognized in both ends.

7.3 Reflections on the solution

The objective of the prototype design was to propose a viable *conceptual* approach for integrating the interaction data—currently available only via the Piwik PRO dashboard—and presenting it in a more understandable format for Omapolku content managers in the same application context. At this point, the prototype is not finished to any extent and the technical implementation was omitted in this study. In practice, the implementation depends on many factors, including the used frameworks and other architectural decisions in the application logic.

Assessing the feasibility of implementing the prototype in practice, there are a few complex issues, in addition to the component identification issue

previously discussed, that need to be solved prior to the implementation. First, the algorithm for calculating a realiable task-specific *exit rate* needs to be carefully designed by taking into account the treatment path logic. Regarding the *exit rate*, a problem occurs in sessions with only one task since the exit rate is most likely close to 100% since such sessions always start and end to the same task. In addition, being able to calculate and visualize task-specific content metrics conserning, e.g., videos and hyperlinks, requires that the treatment path data structure contains required information about these components and that they are identifiable in the data.

7.4 Answers to the research questions

> ***RQ1**: Can web analytics provide detailed information about user behaviour and interaction?*

The results show that web analytics *can* provide sufficient information, provided that the analytics tool used for the data collection is correctly configured. Furthermore, it became evident that it is crucial to record the *context* of the interaction event and a unique *identifier* of the corresponding front-end component that was clicked. As mentioned earlier in this study, currently the only way to identify the front-end components from the Piwik PRO data is to use the `pageTitle` attribute value, which is insufficient for reliable component identification due to varying level of ambiguousness.

> ***RQ2**: What are the challenges of adopting web analytics on a single-page web application?*

The underlying problem of adopting traditional web analytics is that in general, the paradigm of web application design has shifted from static multi-page model to more complex, dynamic single-page model. Since traditional web analytics commonly relies on the *page load* paradigm, tracking dynamic interactions introduces new requirements for the web analytics implementation. In this study, the current configuration of Piwik PRO requires further development in the front-end code of Omapolku to allow tracking and identifying user interactions in sufficient detail. In general, the main issue is the lacking identifiers in the front-end components, which makes it practically impossible to link the interaction data later on with the corresponding front-end components.

> ***RQ3**: Can the output from analytics tools be refined in a way that it will aid Omapolku content managers in spotting dysfunctional content and improving it?*

Related to the anwers of RQ1 and RQ2, the output of web analytics tool—namely, the interaction data—can be refined in a way that can provide valuable insight about the content usage. Furthermore, spotting *dysfunctional* content is possible to some extent, as explained in the example timeline presented in Figure 5.7. In addition, it was possible to calculate useful metrics about the visits, including visit duration, *mean time per action*, *mean page generation time*, and *unique pages*. These metrics can be combined when analyzing the data for dysfunctional content. Furthermore, these metrics can be presented in the analytics component of the proposed prototype.

> ***RQ4**: Does user interaction data reveal similar issues that have been reported via customer feedback?*

The results show that the sample feedback data does not provide sufficient detail for defining exact technical issues. However, it became evident that the most commonly reported issues regarding *responsiveness* and *navigation* can be assessed by examining and visualizing the click path timeline and by utilizing the general path metrics. However, the sample dataset was not large enough for making a statistically valid conclusion that the feedback is generally too ambigous or lacking in detail.

Chapter 8

Conclusions and future work

The objective of this study was to analyze if web analytics can provide data in sufficient detail for analyzing user interaction in Omapolku. Furthermore, the analysis focused on analyzing if the collected interaction data can be refining in a meaningful way to help support the content management of Omapolku and to help spotting dysfumctional content. In addition, challenges of adopting web analytics in a web application using a single-page model were analyzed. While the focus of this study was in the analysis of interaction data, it was also studied if interaction data can reveal similar issues already reported via textual customer feedback.

The results show that the interaction data can provide necessary information for evaluating general metrics which can be used to assess the functionality of treatment pathways in Omapolku. The results also proved that the interaction data can also be utilized to track the usage of treatment path content, such as videos, and to track swapping navigation patterns. However, it became evident that there is am evident need for additional, unique component identifiers in the front-end components of the treatment path interface. In addition, the results show that the interaction data is missing required identifiers and a context, which makes it impossible to trace the action back to the individual components of the application. The results concerning general interaction metrics also revealed an issue in *page generation time* of certain visit actions, possibly implicating an issue with the current configuration of the Piwik PRO analytics platform.

As a solution to the identified issues, supported by the approach of the *desig science* framework, a new prototype was designed. The prototype was seen as a viable solution, providing a conceptual approach for utilizing interaction data in providing useful and actionable insight for Omapolku content managers.

Based on the results, it is recommended to further develop the JavaScript

tag of the Piwik PRO analytics platform and the front-end code of Omapolku by adding unique idenfiers that can be used to identify the interaction data. Provided that the required identifiers are added, it is possible to implement the suggested analytics component prototype for the content managers.

Towards the end of this study, an idea for researching if the development backlog of Omapolku—or any other web application in general—could be more data-driven and prioritized based on findings from customer feedback and interaction data. In addition, another interesting topic for research and experimentation is to see whether the interaction analysis on certain front-end components—treatment pathway tasks in this study—could be achieved by defining *goals* in a web analytics platform. This way, in theory, it could be possible to utilize *funnel reporting* to see how the goal was achieved. In this study, a goal could be seen as a collection of completed tasks. Most importantly, more research should be conducted in order to see if component *identification*, including *context* capturing, can be implemented in a feasible manner so that the interaction data can be linked to the front-end components of the web application.

References

Analystica. SaaS-application for gathering and analysing data, n.d. `http://www.analystica.fi/OhjelmistoENG.aspx` [Accessed on 25 September 2018].

ATLAS.ti. What is ATLAS.ti?, n.d. `https://atlasti.com/product/what-is-atlas-ti/` [Accessed on 17 October 2018].

Richard Atterer, Monika Wnuk, and Albrecht Schmidt. Knowing the User's Every Move: User Activity Tracking for Website Usability Evaluation and Implicit Interaction. In *Proceedings of the 15th International Conference on World Wide Web*, WWW '06, pages 203–212, New York, NY, USA, 2006. ACM. ISBN 1-59593-323-9. doi: 10.1145/1135777.1135811.

Jari Collin, Kari Hiekkanen, Janne J. Korhonen, Marco Halén, Timo Itälä, and Mika Helenius. IT Leadership in Transition - The Impact of Digitalization on Finnish Organizations. Technical report, Aalto University; Aalto-yliopisto, 2015.

Robert Cooley, Bamshad Mobasher, and Jaideep Srivastava. Data preparation for mining world wide web browsing patterns. *Journal of Knowledge and Information Systems*, 1, 04 1999. doi: 10.1007/BF03325089.

Resul Das and Ibrahim Turkoglu. Creating meaningful data from web logs for improving the impressiveness of a website by using path analysis method, April 2009 2009. ID: 271506.

Magdalini Eirinaki and Michalis Vazirgiannis. Web Mining for Web Personalization. *ACM Trans.Internet Technol.*, 3(1):1–27, feb 2003. doi: 10.1145/643477.643478.

Gunther Eysenbach. What is e-health? *Journal of Medical Internet Research*, 3(2):e20, 06/18 2001. doi: 10.2196/jmir.3.2.e20. J1: J Med Internet Res.

Jesse James Garrett. Ajax: A new approach to web applications. *http://www.adaptivepath.com/publications/essays/archives/000385.php*, 2005.

Google Developers. Display live data on your site, n.d. `https://developers.google.com/chart/` [Accessed on 24 October 2018].

Alan Hevner, Alan R, Salvatore March, Salvatore T, Park, Jinsoo Park, Ram, and Sudha. Design Science in Information Systems Research. 28:75, 03 2004.

HUS. Health services in Finland, n.d.a. `http://www.hus.fi/en/patients/Pages/Health-services-in-Finland.aspx/` [Accessed on 8 January 2018].

HUS. Avainlukuja henkilöstöstä, n.d.b. `http://www.hus.fi/hus-tietoa/henkilosto/tietoa-henkilostosta/Sivut/default.aspx/` [Accessed on 5 November 2018].

Madhuri A. Jadhav, Balkrishna R. Sawant, and Anushree Deshmukh. Single Page Application using AngularJS. *International Journal of Computer Science and Information Technologies*, 6(3):2876–2879, 2015.

Avinash Kaushik. *Web analytics: An hour a day.* John Wiley and Sons, 2007.

Kuntaliitto. Sairaanhoitopiirit 2017, 2017. `https://www.kuntaliitto.fi/sites/default/files/media/file/Ervat_Sairaanhoitopiirit2017_0.pdf/` [Accessed on 5 November 2018].

Barry M. Leiner, Vinton G. Cerf, David D. Clark, Robert E. Kahn, Leonard Kleinrock, Daniel C. Lynch, Jon Postel, Larry G. Roberts, and Stephen Wolff. A Brief History of the Internet. *SIGCOMM Comput.Commun.Rev.*, 39(5):22–31, oct 2009. doi: 10.1145/1629607.1629613.

Haibin Liu and Vlado Kešelj. Combined mining of Web server logs and web contents for classifying user navigation patterns and predicting usersfuture requests. *Data & Knowledge Engineering*, 61(2):304–330, May 2007 2007.

Ali Mesbah. Analysis and Testing of Ajax-based Single-page Web Applications. 2009.

Ali Mesbah and Arie van Deursen. Migrating Multi-page Web Applications to Single-page AJAX Interfaces. In *11th European Conference on Software Maintenance and Reengineering (CSMR'07)*, pages 181–190, March 2007. ISBN 1534-5351. doi: 10.1109/CSMR.2007.33.

Ali Mesbah, Arie van Deursen, and Stefan Lenselink. Crawling Ajax-Based Web Applications Through Dynamic Analysis of User Interface

State Changes. *ACM Trans.Web*, 6(1):3:1–3:30, mar 2012. doi: 10.1145/2109205.2109208. URL `http://doi.acm.org/10.1145/2109205.2109208`.

Microsoft. Virtual Hospital improves patients healthcare access, dramatically cuts costs, 2017. `https://customers.microsoft.com/en-us/story/helsinki-university-hospital-health-office-365` [Accessed on 8 September 2018].

Microsoft. Getting started, n.d. `https://code.visualstudio.com/docs` [Accessed on 28 October 2018].

Jakob Nielsen. User interface directions for the web. *Commun.ACM*, 42(1): 65–72, jan 1999. doi: 10.1145/291469.291470.

Node.js. About Node.js®, n.d. `https://nodejs.org/en/about/` [Accessed on 17 October 2018].

J. Oh, W. H. Ahn, S. Jeong, J. Lim, and T. Kim. Automated Transformation of Template-Based Web Applications into Single-Page Applications. In *2013 IEEE 37th Annual Computer Software and Applications Conference*, pages 292–302, July 2013. ISBN 0730-3157. doi: 10.1109/COMPSAC.2013.54.

A. Phippen, L. Sheppard, and S. Furnell. A practical evaluation of Web analytics. *Internet Research*, 14(4):284–293, 2004. https://doi.org/10.1108/10662240410555306.

Piwik PRO. Data-Sensitive Industries Choose Piwik PRO to Track the Customer Journey Across Desktop, Mobile and Secure Member Areas, n.d. `https://piwik.pro/why-piwik-pro/` [Accessed on 27 September 2018].

Arun Sen, Peter A. Dacin, and Christos Pattichis. Current Trends in Web Data Analysis. *Commun.ACM*, 49(11):85–91, nov 2006. doi: 10.1145/1167838.1167842.

H. Singal, S. Kohli, and A. K. Sharma. Web analytics: State-of-art amp;amp; literature assessment. In *2014 5th International Conference - Confluence The Next Generation Information Technology Summit (Confluence)*, pages 24–29, Sept 2014. doi: 10.1109/CONFLUENCE.2014.6949041.

Jaideep Srivastava, Robert Cooley, Mukund Deshpande, and Pang-Ning Tan. Web usage mining: Discovery and applications of usage patterns from web data. *Acm Sigkdd Explorations Newsletter*, 1(2):12–23, 2000.

Antero Taivalsaari, Tommi Mikkonen, Dan Ingalls, and Krzysztof Palacz. Web Browser As an Application Platform: The Lively Kernel Experience. Technical report, Sun Microsystems, Inc, 2008. SMLI TR-2008-175.

Colin Tankard. What the GDPR means for businesses. *Network Security*, 2016(6):5, 2016. doi: https://doi.org/10.1016/S1353-4858(16)30056-3".

Terveyskylä.fi. Omapolku-palvelukanava ja digihoitopolut, n.d.a. `https://www.terveyskyla.fi/palvelut/omapolku-palvelukanava-ja-digihoitopolut` [Accessed on 27 October 2018].

Terveyskylä.fi. Mikä on Terveyskylä.fi?, n.d.b. `https://www.terveyskyla.fi/tietoa-terveyskyl%C3%A4st%C3%A4/mik%C3%A4-on-terveyskyl%C3%A4-fi` [Accessed on 17 October 2018].

David R. Thomas. A General Inductive Approach for Analyzing Qualitative Evaluation Data. *American Journal of Evaluation*, 27(2):237–246, 06/01; 2018/08 2006. doi: 10.1177/1098214005283748. doi: 10.1177/1098214005283748; 21.

Paul Thomas. Explaining Difficulty Navigating a Website Using Page View Data. In *Proceedings of the Seventeenth Australasian Document Computing Symposium*, ADCS '12, pages 31–38, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1411-4. doi: 10.1145/2407085.2407090.

Paul Thomas. Using Interaction Data to Explain Difficulty Navigating Online. *ACM Trans.Web*, 8(4):24:1–24:41, nov 2014. doi: 10.1145/2656343.

Virtuaalisairaala 2.0. Laadukasta hoitoa kaikille asuinpaikasta riippumatta, n.d. `http://www.virtuaalisairaala2.fi/fi/esittely` [Accessed on 8 September 2018].

W3C. Logging Control In W3C httpd, 1995. `https://www.w3.org/Daemon/User/Config/Logging.html#common-logfile-format` [Accessed on 8 October 2018].

W3C. Extended log file format, n.d. `https://www.w3.org/TR/WD-logfile.html` [Accessed on 31 October 2018].

Daniel Waisberg and Avinash Kaushik. Web Analytics 2.0: empowering customer centricity. *The original Search Engine Marketing Journal*, 2(1): 5–11, 2009.

J. S. Zepeda and S. V. Chapa. From Desktop Applications Towards Ajax Web Applications. In *2007 4th International Conference on Electrical and Electronics Engineering*, pages 193–196, Sept 2007. doi: 10.1109/ICEEE.2007.4345005.

Guangzhi Zheng and Svetlana Peltsverger. *Web Analytics Overview*, pages 7674–7683. Encyclopedia of Information Science and Technology, Third Edition. IGI Global, 2015.

Appendix A

Customer feedback data format

Original (in Finnish)	**Translated (in English)**
ID	ID
Valitse palvelu, josta haluat antaa palautetta (pakollinen)	Select a service for which you want to give feedback (required)
Valitse, mistä aiheesta haluat antaa palautetta (pakollinen)	Select a subject of which you want to give feedback (required)
Apumuuttuja	Helper variable
Mitä sisältöä palautteesi koskee?	What content does your feedback concern?
Kirjoita palautteesi (pakollinen)	Write your feedback (required)
Millä laitteella käytät/käytit palvelua?	Which device you use/used the service with?
Millä selaimella käytät/käytit palvelua?	Which browser you use/used the service with?
Haluatko, että sinuun otetaan yhteyttä?	Would you like to be contacted?
Kirjoita lyhyt kuvaus yhteydenottopyynnöstäsi (pakollinen)	Write a short description about your request for contact (required)
Nimesi:	Your name:
Valitse ensisijainen yhteydenottotapa:	Select the primary contact method:
Puhelinnumero:	Phone number:
Sähköpostiosoite:	Email address:
Valitse alue, jossa asut:	Select the region where you live:

Table A.1: Original feedback columns in the Excel file.

Appendix B

Original examples of ambiguous customer feedback

(1) *"Minulla käytössä vain ipad ja iphone, ohjelma ei toim!"*

(2) *"Jumittaa koko ajan, ja ei aukaise ohjelmia, ei ääntä filmeissä ja todella hidas jos edes tekee jotain"*

(3) *"[...] Koko palvelu tökkii, teki mitä vain. [...]"*

Appendix C

Script for analyzing visit data

```
1  var fs = require("fs");
2
3  // Main script
4  try {
5    // Read the JSON file asynchronously
6    fs.readFile("MASTER_DATA.json", function(e, data) {
7      // Parse JSON data
8      var pData = JSON.parse(data);
9
10     let o, actionsCount, actions, pageTitle, pageIdAction,
           generationTimeMillis, timeSpent, interactionPosition;
11
12     // Processed data from all visits
13     let visitsData = [];
14
15     for (let i=0; i<pData.length; i++) {
16
17       // Visit object
18       o = pData[i];
19
20       actions = o['actionDetails'];
21       actionsCount = o['actions'];
22
23       // Array of generation times (in milliseconds)
24       let gtms = [];
25
26       // Array of unique page views
27       let upvs = [];
28
29       // Processed data of a single visit
30       let visitMetrics = [];
31
32       let totalTimeSpent = 0;
```

```
33
34        for (let j=0; j<actions.length; j++) {
35          action = actions[j];
36
37          pageTitle = action['pageTitle'];
38          pageIdAction = action['pageIdAction'];
39          generationTimeMillis = action['
                generationTimeMilliseconds'];
40          timeSpent = action['timeSpent'];
41          interactionPosition = action['interactionPosition'];
42
43          // Check if timeSpent exists
44          if (timeSpent) {
45            timeSpent =  parseInt(timeSpent);
46          }
47
48          // Store in timeline format
49          if (j > 0) {
50            totalTimeSpent += visitMetrics[j-1].timeSpent;
51          }
52
53          // Store visit action values as an object into visit
                data array
54          visitMetrics.push({pageTitle, timeSpent,
                totalTimeSpent, generationTimeMillis,
                interactionPosition});
55
56          // Store generation time into an array
57          gtms.push(parseInt(generationTimeMillis));
58
59          // Store unique page identifiers into an array
60          if (!upvs.includes(pageIdAction)) {
61            upvs.push(pageIdAction);
62          }
63        }
64
65        // Calculate mean page generation time of this visit
66        let meanPGT = calculateMean(gtms);
67
68        // Calculate sample standard deviation of page
              generation time of this visit
69        let ssdPGT = calculateSigma(gtms, meanPGT);
70
71        // Construct visit data object containing all required
              data
72        let data = {visitId, actionsCount, visitMetrics,
              meanPGT, ssdPGT};
73
74        // Store visit data array into array containing data of
```

```
            all visits
75        visitsData.push(data);
76      }
77
78      printVisitsData(visitsData);
79    });
80  } catch(e) {
81    console.error(e);
82  }
83
84  // Print visit data into console
85  let printVisitsData = (visitsData) => {
86    for (let i=0; i < visitsData.length; i++) {
87      let visitData = visitsData[i];
88
89      let visitMetrics = visitData.visitMetrics;
90      let meanPGT = visitData.meanPGT;
91      let ssdPGT = visitData.ssdPGT;
92      let totalActions = visitData.actionsCount;
93
94      console.log('========================================');
95      console.log('#'+ (i+1) + " (" + totalActions + " actions)
            ");
96      console.log('========================================\n')
            ;
97
98      for (let j = 0; j < visitMetrics.length; j++) {
99        let d = visitMetrics[j];
100       console.log(
101         (d.interactionPosition + ". ") +
102         (d.totalTimeSpent !== undefined ? "[" + (Math.floor(d
              .totalTimeSpent / 60)) + "m " +
103         (d.totalTimeSpent % 60) + "s" + "] ": "") +
104         d.pageTitle +
105         (d.timeSpent !== undefined ? " [" + d.
              generationTimeMillis + "ms, " +
106         (Math.floor(d.timeSpent / 60)) + "m " + (d.timeSpent
              % 60) + "s" + "]" : ""));
107
108       // Print total duration by including the timeSpent of
            the last action (if exists)
109       if (j === visitMetrics.length-1) {
110         let totalVisitDuration = d.totalTimeSpent;
111
112         if (d.timeSpent !== undefined) {
113             totalVisitDuration += d.timeSpent;
114         }
115         console.log('\nVisit duration: ' + (Math.floor(
              totalVisitDuration / 60)) + "m " +
```

```
116         (totalVisitDuration % 60) + "s");
117         console.log('Avg. time per action: ' + (
               totalVisitDuration / totalActions).toFixed(1));
118         console.log('Mean PGT: ' + (meanPGT).toFixed(1) + "ms
               ");
119         console.log('sigma of PGT: ' + (ssdPGT).toFixed(1) +
               "ms");
120       }
121     }
122     console.log('\n');
123   }
124 }
125
126 // Calculate mean
127 let calculateMean = (values) => {
128   let n = values.length;
129   if (n == 0) return 0;
130   let v = 0;
131   for (let i=0; i<values.length; i++) {
132       v += values[i];
133   }
134   return v/n;
135 }
136
137 // Calculate standard deviation
138 let calculateSigma = (values, mean) => {
139   let n = values.length;
140   if (n == 0) return 0;
141   let v = 0;
142   for (let i=0; i<values.length; i++) {
143       v += Math.pow(values[i]-mean,2);
144   }
145   return Math.sqrt(v/n);
146 }
```

Appendix D

Example timeline implementation in Google Charts

```
1  google.charts.load("current", {packages:["timeline"]});
2  google.charts.setOnLoadCallback(drawChart);
3  function drawChart() {
4    var container = document.getElementById('chart');
5    var chart = new google.visualization.Timeline(container);
6    var dataTable = new google.visualization.DataTable();
7
8    // Define columns
9    dataTable.addColumn({ type: 'string', id: 'Position' });
10   dataTable.addColumn({ type: 'string', id: 'Title' });
11   dataTable.addColumn({ type: 'date', id: 'Start' });
12   dataTable.addColumn({ type: 'date', id: 'End' });
13
14   // Example data, used in one of the examples in this study
15   dataTable.addRows([
16     [ '1', 'Omapolku', new Date(0,0,0,0,0,0), new Date
         (0,0,0,0,0,13) ],
17     [ '2', 'null', new Date(0,0,0,0,0,13), new Date
         (0,0,0,0,0,20) ],
18     [ '3', 'phone', new Date(0,0,0,0,0,20), new Date
         (0,0,0,0,0,21) ],
19     [ '4', 'phone', new Date(0,0,0,0,0,21), new Date
         (0,0,0,0,0,23) ],
20     [ '5', 'null', new Date(0,0,0,0,0,23), new Date
         (0,0,0,0,2,44) ],
21     [ '6', 'Omapolku', new Date(0,0,0,0,2,44), new Date
         (0,0,0,0,3,28) ],
22     [ '7', 'click-navigaatio-omapolku', new Date
         (0,0,0,0,3,28), new Date(0,0,0,0,6,33) ],
23     [ '8', 'click-istunto-askel-2-K\"ayt\"ann\"on tietoa u\"a
          -- tutkimukseen tulijalle ', new Date(0,0,0,0,6,33),
```

```
        new Date(0,0,0,0,7,5) ],
    [ '9', 'click-istunto-merkitsevalmiiksi', new Date
        (0,0,0,0,7,5), new Date(0,0,0,0,7,45) ],
    [ '10', 'click-istunto-merkitsevalmiiksi', new Date
        (0,0,0,0,7,45), new Date(0,0,0,0,8,33) ],
    [ '11', 'click-istunto-merkitsevalmiiksi', new Date
        (0,0,0,0,8,33), new Date(0,0,0,0,8,52) ],
    [ '12', 'click-istunto-merkitsevalmiiksi', new Date
        (0,0,0,0,8,52), new Date(0,0,0,0,9,2) ]
  ]);
  chart.draw(dataTable);
}
```

Appendix E

Values used in the example timelines

Action #	Elapsed time	Page title	Time spent
1	0m 0s	Omapolku	0m 9s
2	0m 9s	null	0m 3s
3	0m 12s	null	0m 11s
4	0m 23s	null	0m 4s
5	0m 27s	null	0m 5s
6	0m 32s	null	0m 3s
7	0m 35s	click-yleiset-kirjauduulos	-

Table E.1: Data of Visit 1 used for Figure 5.4 timeline.

Action #	Elapsed time	Page title	Time spent
1	0m 0s	Omapolku	0m 13s
2	0m 13s	null	0m 7s
3	0m 20s	phone	0m 1s
4	0m 21s	phone	0m 2s
5	0m 23s	null	2m 21s
6	2m 44s	Omapolku	0m 44s
7	3m 28s	click-navigaatio-omapolku	3m 5s
8	6m 33s	click-istunto-askel-2-Käytännön tietoa uä – tutkimukseen tulijalle	0m 32s
9	7m 5s	click-istunto-merkitsevalmiiksi	0m 40s
10	7m 45s	click-istunto-merkitsevalmiiksi	0m 48s
11	8m 33s	click-istunto-merkitsevalmiiksi	0m 19s
12	8m 52s	click-istunto-merkitsevalmiiksi	0m 10s

Table E.2: Data of Visit 2 used for Figure 5.5 timeline.

Action #	Elapsed time	Page title	Time spent
1	0m 0s	Omapolku	0m 20s
2	0m 20s	click-navigaatio-omapolku	0m 15s
3	0m 35s	click-istunto-askel-5-Toinen ultraääniseulontatutkimus eli Rakenneultraääni-tutkimus - info	0m 3s
4	0m 38s	click-istunto-avaavideoruutu-Toinen seulontaultraäänitutkimus	1m 33s
5	2m 11s	click-istunto-video-sulje	0m 2s

Table E.3: Data of Visit 9 used for Figure 5.6 timeline.

Action #	**Elapsed time**	**Page title**	**Time spent**
1	0m 0s	Omapolku	0m 8s
2	0m 8s	click-navigaatio-omapolku	0m 23s
3	0m 31s	click-istunto-merkitsevalmiiksi	0m 47s
4	1m 18s	click-istunto-merkitsevalmiiksi	0m 31s
5	1m 49s	click-istunto-merkitsevalmiiksi	0m 8s
6	1m 57s	click-istunto-merkitsevalmiiksi	0m 12s
7	2m 9s	click-istunto-merkitsevalmiiksi	0m 4s
8	2m 13s	click-istunto-merkitsevalmiiksi	0m 21s
9	2m 34s	null	0m 15s
10	2m 49s	null	0m 24s
11	3m 13s	click-navigaatio-yleisvalinnat-Usein kysytyt kysymykset	0m 13s

Table E.4: Data of Visit 10 used for Figure 5.7 timeline.