Software Design of a Health BCSS: Case Onnikka

University of Oulu
Department of Information Processing
Science
Master’s Thesis
Tuomas Alahäivälä
5.5.2013
Abstract

The purpose of this thesis is to study the process of designing and developing a web-based application for supporting people struggling with overweight to change their behaviour in order to prevent health risks. Obesity and its co-morbidities represent one of the major public health problems globally. To prevent health problems caused by overweight, such as the metabolic syndrome, a sustainable change in an individual’s health behavior is required. Behavior Change Support Systems (BCSSs) have been introduced as objects of study in the field of persuasive technology, defined as information systems designed to form, alter or reinforce attitudes, behaviors or an act of complying without using deception, coercion or inducements. One of the most prominent domain areas for these systems is promoting behavior change for improved health and healthier lifestyles. Despite the importance of the research domain, descriptions from the systems in most cases have been presented in too general level to be benefited from. In the scope of this thesis, we built a full-fledge BCSS within the weight loss and maintenance domain to support users lifestyle change process during a 52-week long intervention. The system design process is studied here to find re-usable conceptual designs such as software architectures and design patterns for future BCSS development and research.

Keywords
persuasive systems design, behaviour change support systems, weight loss intervention
Foreword

I would like to thank Professor Harri Oinas-Kukkonen for his guidance with this thesis. I would also like to thank the members of the OASIS research group and the PrevMetSyn consortium for their contributions, especially Terhi Jokelainen, Anna-Maria Keränen, Pasi Karppinen, Professor Markku Savolainen and Tuire Salonurmi. I also pay gratitude to all the people involved in the testing phase and expert evaluations.

The nice people at Balsamiq Studios kindly provided a license for their software to create the beautiful mock-ups used in this thesis. The web development practices presented in the excellent Ruby on Rails Tutorial E-book by Michael Hartl were widely put into use in the development of Onnikka.

Tuomas Alahäivälä

Oulu, January 31, 2011
Contents

Abstract .......................................................................................................................... 2
Foreword ......................................................................................................................... 3
Contents......................................................................................................................... 4
1. Introduction ................................................................................................................ 5
2. Background ................................................................................................................ 7
   2.1 Behavior Change Support Systems ................................................................. 7
   2.2 Persuasive Systems Design ........................................................................... 9
   2.3 Web-based weight loss interventions ......................................................... 11
3. Research Method.................................................................................................... 13
   3.1 Design Science in Information Systems Research ....................................... 13
   3.2 Design Science Research Methodology ...................................................... 14
   3.3 Implemented DSRM process ...................................................................... 15
4. The Software Artifact............................................................................................ 17
   4.1 Persuasion Postulates .................................................................................. 17
   4.2 Persuasion Context .................................................................................... 17
       4.2.1 The Intent ......................................................................................... 17
       4.2.2 The Event ....................................................................................... 17
       4.2.3 The Strategy ................................................................................. 18
   4.3 Persuasive Software Features ...................................................................... 18
   4.4 The Development Process ........................................................................... 21
       4.4.1 Requirements specification ............................................................... 21
       4.4.2 Software Architecture Design ........................................................... 23
       4.4.3 User Interface Design ................................................................... 24
       4.4.4 Implementation of the Software Components .................................. 35
5. Evaluation ................................................................................................................ 45
   5.1 User testing ....................................................................................................... 45
   5.2 Evaluation of the Persuasive Features ........................................................... 47
       5.2.1 Primary Task Support ....................................................................... 47
       5.2.2 Dialogue Support ............................................................................ 48
       5.2.3 System Credibility Support .............................................................. 49
       5.2.4 Social Support ............................................................................... 50
6. Discussion ................................................................................................................ 51
7. Conclusion ............................................................................................................... 54
References ................................................................................................................... 56
1. Introduction

This thesis is conducted within the *PrevMetSyn* research project in the University of Oulu, which studies new counseling methods for the prevention of the metabolic syndrome. The goal of the project is to reduce risk factors leading to type 2 diabetes or cardiovascular diseases on people struggling with overweight, while simultaneously discovering the best methods for preventing the metabolic syndrome and overweight. As a part the study, a persuasive web service is built to support the weight losers in changing their lifestyles for the better in order the reduce health problems. This thesis addresses the design and development process of that service.

Recently, behavior change support systems (BCSSs) have been introduced as objects of study within the persuasive technology field (Oinas-Kukkonen, 2010). These systems are designed to form, alter, or reinforce attitudes, behaviors, or acts of complying without using deception or coercion (Fogg, 1998). The interests in BCSS research includes the approaches, methodologies, processes, and tools for developing such systems, as well as studying their possible impacts. One of the most prominent domain areas for BCSS applications is promoting behavior change for improved health and healthier lifestyles (Kraft et al., 2009).

Obesity and its co-morbidities represent one of the major public health problems globally. To prevent health problems caused by overweight, such as metabolic syndrome, a sustainable change in an individual’s health behavior is required (Keränen, 2011). The effectiveness of nutritional and weight counseling varies: highly effective personal counseling is too expensive for wide use, while distributing information might be cost-efficient but not effective enough for sustainable weight loss. Therefore, new cost-efficient counseling methods are needed. Because of this, web-based weight loss and maintenance software systems have become an important research area in health behavior change. These systems may utilize many different kinds of persuasive features (Lehto & Oinas-Kukkonen, 2010). However, there is only a little research on how they should be implemented.

Although BCSSs have emerged as an important research area, the descriptions of the systems in most cases have been presented in too general level (Lehto & Oinas-Kukkonen, 2010; Lehto & Oinas-Kukkonen, 2010; Segerståhl et al., 2010). When describing a BCSS, the persuasion context, containing many design issues related to the chosen technology, its use, and the users, should be covered comprehensively (Oinas-Kukkonen, 2012).

With black-box thinking of software systems without an actual description of the implementation it would be difficult to argue any generalizable results. Through conceptual designs such as software architectures, design patterns and heuristics, BCSSs and other persuasive systems research can be advanced from proof-of-concepts to actual software development guidelines. The purpose of this thesis is to study the process of designing and developing a web-based application for supporting people struggling with overweight to change their behavior in order to prevent health problems such as the metabolic syndrome. Theoretical constructs such as the Persuasive Systems Design (PSD) model (Oinas-Kukkonen & Harjumaa, 2009) are used in designing the system.
The study is carried out utilizing the design science research methodology, which includes an iterative process of designing and evaluating a functional IT artifact to produce a solution to the research problem. The Design Science Guidelines suggested by Hevner et al. (2004) and the Design Science Research Methodology framework presented by Peffers et al. (2007) are applied in conducting the research.

The main contribution of the thesis is the full-fledged BCSS produced, along with the documentation of its design and development process. The resulting software artifact can be utilized in the development of health care counseling methods, as well as in studying the application of the design methods for persuasive technology. The system design process can be studied to find re-usable conceptual designs such as software architectures and design patterns for future BCSS development and research.

The thesis contains seven chapters. Following this introduction, the second chapter describes our theoretical background from prior research on BCSSs and the PSD model. Chapter three explains the design science research method. Chapter four describes the design and development process and the characteristics of the artifact produced. Chapter five contains the evaluation of the artifact base on the conducted user testing and expert evaluations. Chapter six discusses the results of the study. Finally in chapter seven, summary and conclusions are given.
2. Background

In this chapter, the theoretical background of the study and the concepts of behavior change support system, persuasive systems design and web-based weight loss interventions are introduced. The PSD model, being the state-of-the-art construct considering BCSSs development (Oinas-Kukkonen, 2010), was chosen as primary approach to our systems design process.

2.1 Behavior Change Support Systems

A major factor in people’s attitudes or behaviors change is persuasion, which by definition means “an attempt to change attitudes or behaviors or both (without using coercion or deception)” (Fogg, 1998). Interactive information technology designed for promoting such change is respectively known as persuasive technology. Modern communication technologies such as the Web and mobile phones enable many possibilities for persuasion, since their users can be easily reached and they can combine means of interpersonal and mass communication. Welfare, commerce, education, safety, environmental preservation and occupational effectiveness among others are all viable application areas for persuasive technology (Fogg, 2003.)

Oinas-Kukkonen & Harjumaa (2008) have defined persuasive systems as “computerized software or information systems designed to reinforce, change or shape attitudes or behaviors or both without using coercion or deception”. Of those three incomes, reinforcing means the reinforcement of current attitudes that makes them more resistant to change. A changing outcome means changing a person’s response to an issue. Shaping means the formulation of a novel behavior pattern for a certain situation. In persuasive systems, either computer-human persuasion or computer-mediated persuasion may be used (Fogg, 1998). In computer-mediated persuasion, persuasion happens between people via communication technology. Computer-mediated persuasion instead happens when people interact with computer technology, in which case it may be challenging to identify the actual persuader. Based on several social psychological theories, Oinas-Kukkonen & Harjumaa have suggested using differing persuasion strategies and techniques for different persuasion goals.

The prior research on persuasion has provided multiple approaches and techniques for studying persuasive systems. According to information processing theory, the persuasive impact of messages derives from six steps a person goes through when persuaded: information presentation, attention, comprehension of the arguments, yielding to the position presented, retention for some time and action in compliance with the new position (McGuire, 1973). The cognitive consistency theory states that people like their views about the world to be organized and consistent, and work to keep their mental structures that way (Fraser et al., 2001). The elaboration likelihood model presented by Petty and Cacioppo (1986) states that people are different when it comes to being easily persuaded by either argumentative central route, or peripheral route relying on cues. Cialdini’s (1988) influence techniques approach presents that people may answer to triggers for certain behavior in either in an automatic or controlled way. The coactive approach to persuasion includes ways to move the persuaders psychologically closer to the persuadees (Simons et al., 2001). In the computer science discipline, theories such as Technology Acceptance Model (Davis, 1989) and the Unified Theory
of Use and Acceptance of Technology (Venkatesh et al., 2003) have been constructed to understand the factors affecting the people’s intention of use regarding information systems. The persuasive technology framework by Fogg (2003) also provides several means for understanding persuasive technology.

Whereas persuasive technology is the field of research, behavior change support systems (BCSSs) are the objects of study within the field. Oinas-Kukkonen (2010) has defined a BCSS as “an information system designed to form, alter or reinforce attitudes, behaviors or an act of complying without using deception, coercion or inducements”. The interests in BCSS research include the approaches, methodologies, processes and tools for developing such systems, as well as studying their possible impacts. One of the most prominent domain areas for BCSS applications is promoting behavior change for improved health and healthier lifestyles. A prime example of such system is the Nike+ web service, utilizing several persuasive elements to encourage increased exercise (Oinas-Kukkonen & Harjumaa, 2009). Although an important emerging research area, the elaborate details of BCSS development such as technical descriptions of the systems have not been widely covered in prior research (Oinas-Kukkonen, 2010).

Oinas-Kukkonen (2010) has stated that the types of behavioral change can be divided into three categories persuasive systems can aim for: C-, B-, and A-Change. Different means of persuasion should be used according to which kind of change is pursued. C-Change means change in the act of complying with the goals provided by the system. This can be achieved by providing triggers for users to take action and comply with given requests, although they might not necessarily be properly motivated to do so. Achieving a long-term behavior change is more challenging, though a successful C-Change may eventually lead up to B-Change, which ensues for a more enduring behavioral change. For a sustainable B-Change though, an A-Change is needed, meaning the influence to the users’ attitudes altogether. For example to successfully overcome addictions, the users need support for the both A- and B-Change.

**Table 1. Outcome/Change design matrix**

<table>
<thead>
<tr>
<th></th>
<th>C-Change</th>
<th>B-Change</th>
<th>A-Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>F-Outcome</td>
<td>Forming an act of complying (F/C)</td>
<td>Forming a behavior (F/B)</td>
<td>Forming an attitude (F/A)</td>
</tr>
<tr>
<td>A-Outcome</td>
<td>Altering an act of complying (A/C)</td>
<td>Altering behavior (A/B)</td>
<td>Altering an attitude (A/A)</td>
</tr>
<tr>
<td>R-Outcome</td>
<td>Reinforcing an act of complying (R/C)</td>
<td>Reinforcing a behavior (R/B)</td>
<td>Reinforcing an attitude (R/A)</td>
</tr>
</tbody>
</table>

From each type of behavioral change there are three potential, successful outcomes: formation (F-Outcome), altering (A-Outcome), and reinforcement (R-outcome). F-Outcome equals the formulation of a novel behavioral pattern for a situation. A-Outcome means change in an individual’s response to an issue. R-outcome stands for the reinforcement of existing attitudes or behaviors. From the intended outcomes and the change categories, a design matrix (see Table 1) can be built to help decide if the BCSS in question should aim for the goal of forming, altering, or reinforcing an act of complying, a behavior, or an attitude.
2.2 Persuasive Systems Design

According to Oinas-Kukkonen & Harjumaa (2009), the development of persuasive systems requires three steps: understanding key issues behind persuasive systems, analyzing the persuasion context, and the design of system qualities. Based on empirical studies and the prior research mentioned above, the fundamental issues concerning persuasive systems have been summarized in seven postulates:

1. Information technology is never neutral.
2. People like their views about the world to be organized and consistent.
3. Direct and indirect routes are key persuasion strategies.
4. Persuasion is often incremental.
5. Persuasion through persuasive systems should always be open.
6. Persuasive systems should aim at unobtrusiveness.
7. Persuasive systems should aim at being both useful and easy to use.

Postulate 1 means information technology constantly influences people’s attitudes and behavior in a multi-phased process, during which different factors such as goals may change. Postulate 2 is based on cognitive consistency theory, suggesting that the system should support for users to make commitments. According to postulate 3, people who make careful decisions based on given knowledge should be persuaded using direct routes, while those less thoughtful should be approached with more indirect cues. Postulate 4 states people are easier to persuade through series of incremental suggestions rather than aim for a big one-time change. Postulate 5 suggests revealing the designer bias behind the system openly. The systems should also avoid disturbing users performing their primary tasks, as suggested by postulate 6. Finally, postulate 7 reminds the importance of likable system qualities concerning successful persuasion (Oinas-Kukkonen & Harjumaa, 2009).

The next step in designing persuasive systems is analyzing the persuasion context. According to Oinas-Kukkonen & Harjumaa (2009), this analysis includes “recognizing the intent of persuasion, understanding the persuasion event, and defining and/or recognizing the strategies in use”. Acknowledging the intent includes determining who is the actual persuader. Since computers don’t have any intentions of their own, the source of persuasion in a system is always one of those who create, distribute, or adopt the persuasive technology. Analyzing the intent also covers defining the change type, intended for either attitude or behavior change, or both. As for understanding the persuasion event, the contexts of use, user and technology should be recognized. The use context covers the features of the problem domain question, while the user context minds the differences between individuals’ motivations, lifestyles, goals, etc. The technology context of the specific system used also has an impact on the persuasion event. Finally, identifying the persuasion strategies in use includes analyzing the message that is attempted to convey and the route, whether direct or indirect, that is used to reach the persuadee.

The final phase in the persuasive systems development is the actual design of persuasive features. Oinas-Kukkonen & Harjumaa (2009) have stated that the creation of such system includes the generic steps of persuasion context analysis and selection of persuasive design principles, requirement definition of software qualities, and the actual software implementation. The requirement definition of a software product covers gathering all the functional and nonfunctional requirements the implemented system should meet. As the postulates mentioned above are not comprehensive enough for such a task, Oinas-Kukkonen & Harjumaa (2009) have proposed a set of design principles for
persuasive systems distributed to four categories: primary task, dialogue, system credibility and social support. These principles may function as guidelines for software quality requirements as well as an evaluation tool for persuasive systems.

The design principles for supporting the user’s primary task include reduction, tunneling, tailoring, personalization, self-monitoring, simulation, and rehearsal. Reduction means distributing complex behavior into simpler tasks to accomplish goals. Tunneling is using the system to guide users in attitude change by persuading them along the way. Tailoring is personalizing the system according to certain user needs. By self-monitoring the users can track their progress within the system. Simulation means providing ways to observe the link between cause and effect with regard to certain behavior. The system should also allow rehearsal for the desired behavior (Oinas-Kukkonen & Harjumaa, 2009).

Dialogue support category of design principles includes praise, rewards, reminders, suggestion, similarity, liking and social role. They are related to the dialogue between the system and its users. By offering praise for the target behavior, users can be made more prone to persuasion. Providing rewards also works as a motivational tool. A system reminding users of their target behavior will improve their performance. Also, appropriate suggestions should be made along the way for increased persuasion. Similarity between users and the systems makes the user more persuadable. Liking means making the system’s look & feel pleasurable to use. The System could also adopt a social role for users more likely to use it (Oinas-Kukkonen & Harjumaa, 2009).

The system credibility support category contains trustworthiness, expertise, surface credibility, real-world feel, authority, third-party endorsements and verifiability. Trustworthiness can be achieved by providing truthful, unbiased information. Expertise on the subject matter should also be implied. Surface credibility means making competent firsthand experience. Real-world feel can be achieved by making the organization behind the system more visible. System should also refer to an authority for increased credibility. Third-party endorsements from well-known sources are also beneficial. Verifiability of the system can be expressed by referring to outside sources (Oinas-Kukkonen & Harjumaa, 2009).

Design principles for social support are social learning, social comparison, normative influence, social facilitation, cooperation, competition and recognition. Social learning means a person can learn from observing other people’s behavior. Social comparison is about comparing an individual’s performance with the performance of others. Normative influence aims for users to adopt the target behavior from peer pressure. Social facilitation happens as a result from the possibility to discern other people performing same tasks. A chance for actual cooperation with other users is also a strong motivating factor. Competition within the users can also motivate for reaching the target behavior. Public recognition when reaching their goals is also motivating for users (Oinas-Kukkonen & Harjumaa, 2009).

In designing and developing BCSSs, the research knowledge on persuasive systems described above should be utilized. The seven postulates behind persuasive systems need to be addressed, and the persuasion context of the intent, the event, and the strategy should be thoroughly analyzed. The general software design issues such as usefulness, ease of use, ease of access, high information quality, simplicity, convenience, attractiveness, lack of errors, responsiveness, positive user experience, and user loyalty should be considered (Oinas-Kukkonen, 2010). Special attention should be to paid to utilizing the design principles in the PSD model, concerning the support for
the primary task, the human-computer dialogue, the perceived system credibility, and the social influence.

According to a publications review by Tørning & Oinas-Kukkonen (2009), the most utilized PSD features on BCSS studies have been tailoring, tunneling, reduction, and self-monitoring of the primary task category, suggestion relating to dialogue support, surface credibility of the perceived system credibility category, and social comparison, normative influence, and social learning from the social influence category. The study also showed that most of the previous studies on persuasive systems didn’t achieve their intended goals, and didn’t utilize any persuasive design frameworks. This suggests that the PSD model should be more widely utilized when designing such systems. Other things to consider understand the user segments, clear description of the technology, the message, and the route, revealing the designer bias, and the possible ethical issues. Empirical BCSS research provides the ways for quantified measuring of the system success, provided that the goals, features, and the outcomes of persuasive systems are explicitly defined (Oinas-Kukkonen, 2010).

2.3 Web-based weight loss interventions

Health behavior change is one of the most prominent areas of persuasive systems, and specifically weight loss and maintenance supporting systems have proven a viable application domain. Health behavior change support systems have also been successfully utilized in the domains such as smoking cessation, hazardous drinking, diabetes, asthma, tinnitus, stress, anxiety and depression, complicated grief, and insomnia (Strecher, 2007). Information systems that promote improved health and healthier lifestyles have been stated one of the most prominent areas for future healthcare improvement (Kraft et al. 2009).

Web-based interventions can be defined as a mainly self-guided online programs that are operated through a website in order to receive health-related assistance (Barak, Klein & Proudfoot, 2009). Their purpose is to result in positive change in behaviors and improve knowledge, understanding and awareness in their users by providing trustworthy and unbiased information contents and utilizing interactive functionalities. Web-based weight loss interventions have been proven to have potential to accomplish similar outcomes as in other lifestyle treatment alternatives (Neve et al., 2009). Software features utilized in the existing systems include self-monitoring functionalities, food diaries, body mass index (BMI) calculators, support forums and coach messaging (Bennett & Glasgow, 2009). Significant results in weight loss have been perceived in weight loss interventions that are highly structured, provide a possibility for support from a counselor and for synchronous communication, contain behavior therapy components, provide tailored information contents, include interactive and dynamic functionalities, and encourage for frequent logins (Bennett & Glasgow, 2009; Krukovski et al., 2008).

Lehto and Oinas-Kukkonen (2010) have studied the utilization of persuasive features on six current web-based weight loss intervention systems. The persuasive context of the websites was also analyzed. The background organization was clearly stated only on two of the services, and none explicitly stated whether they pursued for attitude change or behavior change only. The use context was described only on a general level in all of the sites, and half of them didn’t target any different user groups. The technologies used by the sites included videos on every site and mobile functionalities on half of them. Podcasts, downloadable tools and social media connectivity were also offered. The
messages intended for persuasion in the sites tended to use a set of arguments instead of just one strong statement.

Considering the persuasive features, the sites were relatively strong on the primary task support. Self-monitoring, reduction and tunneling were widely used. Tailoring was not very commonly used, with half of the sites not utilizing any tailoring at all. Personalization, simulation and rehearsal were quite common instead. Utilizing dialogue support features was weaker: half of the sites offered praise and a few offered virtual rewards. Half of the sites also used reminders and three of them gave suggestions for action. System credibility was accomplished reasonably well, all of the sites successfully appearing trustworthy and presenting expertise. Real-world feel, authority, third-party endorsements and verifiability instead were not widely used. All of the sites provided some sort of social support: techniques such as blogs, discussion forums and groups, chat rooms and instant messaging were utilized. Lehto and Oinas-Kukkonen (2010) conclude that the web-based interventions should be considered as supplements to traditional treatment and peer support instead of replacements to them. Overall, the results suggested that the current weight loss sites might not actually appear very persuasive, and there is room left for improvement in the design and implementation process of such services.
3. Research Method

This chapter describes the design science research methodology used to carry out the study.

3.1 Design Science in Information Systems Research

The research in Information Systems (IS) discipline addresses problems in the environment of people, organizations and technologies. It much relies on two paradigms: behavioral science and design science. While the behavioral science paradigm focuses on producing theories explaining or predicting the behavior of individuals or organizations, design science aims on developing new innovative artifacts to extend the boundaries of human problem solving and organizational capabilities. Behavioral science has it origins in natural sciences, while design science’s roots are in engineering research. It is argued that for IS research to make significant contributions, behavioral science and design science should be used complementarily to solve fundamentals problems in productive applications of information technology (Hevner et al., 2004).

According to Hevner et al. (2004), in behavioral science research on IS the object of study is often an Information Technology (IT) artifact already implemented in an organizational context. Design science instead aims to create and evaluate these artifacts to solve certain organizational problems, exploiting the knowledge based on prior research. The IT artifacts can include constructs (vocabulary; symbols), models (abstractions; representations), methods (algorithms; practices) and instantiations (implemented systems). They can also include social innovations or some properties of technical, social, or informational resources (Peffers et al., 2008). The artifacts are represented in structured form that can then be evaluated using quantitative metrics or qualitative comparisons with other designs. The cycle of the design process and the evaluation of the artifact is iterated to fully understand the problem and improve the outcome (Hevner et al., 2004).

The Information Systems research framework presented by Hevner et al. (2004) examines the relations between environment, IS research and knowledge base. The environment defines the problem space, consisting of people, organizations and technology, which addresses the business needs that ensure the relevance for the conducted research. The knowledge base contains the foundations and methodologies, based on which the research is conducted. The foundations include the theories, frameworks and tools which the previous research in IS and other disciplines have provided, while the methodologies provide guidelines to be used in justifying and evaluating the artifact. Research rigor is then accomplished by using this pool of previous knowledge and methodologies. The IS research consists of two phases: development and justification of theories explaining or predicting the target phenomena and building and evaluation of the artifact needed to solve certain problem. The research makes continuing additions to the knowledge base and provides applications to satisfy the business needs of the environment.
Based on the fact that the building and the application of an artifact provide knowledge and understanding of a design problem and its solution, Hevner et al. (2004) have suggested seven guidelines for design science in IS research:

1. Design as an Artifact: Design science research should produce a viable artifact, be it a construct, a model, a method or an instantiation.
2. Problem Relevance: The solutions brought by design science should address an import and relevant problems in the domain area.
3. Design Evaluation: The design artifact’s utility, quality and efficacy should be rigorously evaluated.
4. Research contributions: Design science research should represent verifiable contributions in the design artifact, foundations, and methodologies areas.
5. Research rigor: The research should rely on rigorous methods in building and evaluating the artifact.
6. Design as a Search Process: Design should be seen as an iterative process that produces solutions via a recurring cycle of generating and testing.
7. Communication of Research: The outcomes of the research should be effectively communicated to both technology-oriented and management-oriented audiences.

3.2 Design Science Research Methodology

Peffers et al. (2006) have presented a framework for conducting research based on the previous design science principles. Their Design Science Research Methodology (DSRM) incorporates principles, practices and procedures to carry out such research. To build a consensus of the different components of the DSRM process, seven representative papers and presentations on the subject were reviewed. The synthesis resulted in a model of six consecutive activities of the DS process: problem identification and motivation, defining the objectives for a solution, design and development, demonstration of the artifact, its evaluation, and communicating the results. See Figure 1 for visual representation of the process.

![Diagram of the DSRM process](image)

Possible entry points for research

**Figure 1. DSRM process presented by Peffers et al. (2006)**

The first activity of problem identification and motivation includes defining the specific problem of research and justifying the value of its solution. Knowledge of the state of
the problem and the importance of the solution are needed resources of the activity. The second activity, defining the objectives for a solution, includes inducing the objectives from the problem definition and from the knowledge of what is doable. The objectives can be either quantitative or qualitative. The activity requires knowledge of the state of the problems and possible current solutions. Third activity is the design and development of the artifact, whether construct, model, method or instantiation. The artifact’s desired functionality and architecture is defined and the actual artifact is created, considering the theory basis for solving the problem is known. The fourth activity of demonstration includes demonstrating the artifact use for solving the problem at hand via experimentation, simulation, case study or such. Effective knowledge of the artifact use is required. The fifth, evaluation activity includes the actions to observe and measure how well the solution to the problem is supported by the artifact. The evaluation methods can differentiate from comparison with the objectives of solution to quantitative satisfaction surveys and qualitative measures of system performance. Based on the results of the evaluation, the process can revert back to design and development phase, or continue to the sixth activity of communication. It includes communicating the results of the study to appropriate audiences, such as other researchers in the discipline or practicing professionals (Peffers et al., 2006; Peffers et al., 2008).

The DSRM process does not have to be carried out in the exact sequential order of the model. The research can be started from a certain step, moving outward. The model includes several possible entry points for starting the research. If the research originated from observation of the problem or suggested future research from an earlier paper, a problem-centered approach starting from activity 1 would be suitable. An objective-centered solution derived from business needs for an artifact would instead start from activity 2. An existing artifact, possibly from another domain, might function as a basis for a design- and development-centered approach starting from activity 3. Finally, observing a successful practical solution could result in a client- or context-initiated approach from activity 4, in which the research rigor would be applied to the process retroactively (Peffers et al., 2008).

3.3 Implemented DSRM process

The DSRM framework was implemented to carry out the system design and development process (see Fig. 2). The entry point in the research was the need for implementation details stated in the BCSS research agenda (Oinas-Kukkonen, 2010).
**Figure 2. Applied DSRM process**

*Problem centered approach.* The basis of this study comes from the fact that the details of BCSS development are not widely covered in the prior research. The research agenda for BCSS research (Oinas-Kukkonen, 2010) sets multiple directions for future studies on the subject, including the inspection of their development process.

*Problem identification & motivation.* We can identify the targeted problem as seeking answers to how BCSSs should be implemented. The outcomes of the development process will help understanding and identifying the individual features of BCSSs and make way for their future research and development.

*Objectives of solution.* The instrument to studying our set problem is the development process of a full-fledge BCSS in the domain area of weight loss and maintenance interventions. The prior research on BCSSs development, most fundamentally the PSD model is used. The *PrevMetSyn* research project provides an environment to develop and demonstrate the features of our IT artifact.

*Design & development.* The design and development of the produced artifact follows the standard software development process along with best practices in persuasive systems development. The constructs of persuasion postulates, persuasion context and persuasive software feature categories are used. Prior knowledge on the requirement specifications and software architectures of persuasive systems is utilized, as well as modern web technologies and design patterns. The design and development process is rigorously documented.

*Demonstration.* The resulting system is at first demonstrated in the development environment used by the selected test users. Standard user testing practices are used to capture the user’s experiences on the system. The final version will be put into production use later along with the intended weight loss intervention in the *PrevMetSyn* project.

*Evaluation.* The evaluation of the artifact is done by gathering the results of user testing along with heuristic evaluation conducted by experts with extensive knowledge on the PSD model and BCSSs.

*Communication.* The results of this study will be communicated in the form of this thesis as well as future research papers.
4. The Software Artifact

As the software artifact of our study, weight loss and maintenance intervention web service Onnikka was built, intended to support the prevention of metabolic syndrome through a yearlong lifestyle intervention. The name refers to a local dialectal word meaning “a bus,” creating a metaphor of the behavior change process as a journey. The main features in Onnikka are a channel to receive weekly content provided by health professionals, and the possibility to set goals and follow the individual’s progress within the system by submitting personal entries related to health behavior. The development followed the persuasive systems design process for BCSSs (Oinas-Kukkonen, 2012). This chapter describes the design and development process and the system qualities.

4.1 Persuasion Postulates

As a basis for our design process, we adopted the persuasion postulates stated in the PSD model. We acknowledge IT is not neutral, so the functionalities of the system we design will have an impacts on the behavior of its users at least for the duration of the intervention. Following the postulates 2 and 4, we should design the presentation of the intervention contents building incrementally on the users’ current worldviews. Postulates 3 and 5 are minded in using direct routes for channeling information and being open about the nature and the purpose of the system. Good software development, usability and user experience design practices are applied to comply postulates 6 and 7.

4.2 Persuasion Context

We analyzed the persuasion content, including the persuasion intent, event and strategy in order to achieve a general view of the desired system qualities.

4.2.1 The Intent

The persuaders behind the system are essentially the initiators of the research project, consisting of The Northern Ostrobothnia Hospital District, the Ministry of Social Affairs and Health and the University of Oulu. The intention of creating the system is to help its users lose weight to reduce the risk for diseases related to obesity such as the metabolic syndrome. Since a sustainable weight loss requires permanent changes in the lifestyle, we are ultimately reaching for an attitude change in the users concerning their health behaviors. As the users may come from different backgrounds, they may be required to form, alter or reinforce their attitudes towards a healthy lifestyle.

4.2.2 The Event

User context

The potential users of Onnikka are those struggling with overweight, which exposes them to diseases such as metabolic syndrome. Although pursuing a similar behavior change, the user group is not homogeneous; rather, it includes individuals with different backgrounds, goals, and motivations. The differences between individuals, primarily the starting time of the intervention and the possible need for additional eating behavior counseling, were to be considered. One of the most important aspects of the user context
was minding each user’s phase of intervention, so that they could be provided the proper information at the right time. The users sharing the same starting date would belong to the same group of users during the whole intervention, being able to interact with each other. For setting conscious performance goals, the users are encouraged to set a target weight.

It has been claimed that a successful health behavior change at the individual level requires receiving targeted and tailored health information (Enwald & Huotari, 2010). In addition, enhancing an individual’s eating behavior is one of the major factors in maintaining weight loss (Keränen, 2011). Since not all persons exposed to the metabolic syndrome have the specific need for counseling in eating behavior, the informational content related to enhancing eating behavior were determined to be only accessible to a certain group of tailoring-enabled users based on their behavioral profiles. This group would receive additional health information and exercises on enhancing their eating habits.

**Use context**

Using Onnikka is meant to support the weight loss and maintenance process of the users by providing them knowledge to make the right health behavior choices and tools for monitoring their progress. 52 weeks long intervention program is provided in order to intensively support individual’s weight loss process and persevering adoption of healthy lifestyle. Discussion among peers makes it possible to receive social support. To track their change progress, users can engage in the act of self-monitoring by adding entries about their weights, their current motivation, and exercise and eating habits. The users are also provided an opportunity to ask questions from the health professionals.

**Technology context**

The technological implementation of Onnikka was to be carried out using modern web technologies in order to produce a stable and highly responsive system that users could access on a vast set of devices: desktop PCs, laptops, tablets or smart phones. Instead of building separate web and mobile clients, we built a HTML (Hypertext Markup Language) and JavaScript-based web application using responsive design techniques so the site’s layout would adapt to screen size of the user’s device. The final implementation was carried out using Ruby on Rails web framework, MySQL database, Twitter Bootstrap front-end framework, and jQuery JavaScript library.

**4.2.3 The Strategy**

The persuasive messages conveyed are essentially the weekly health information contents provided to the users via the web interface and e-mails. For the period of 52 weeks, users are introduced new health counseling materials weekly: informational articles, simple exercises to practice the desired behaviors, and tips for achieving the change. Healthcare professionals provided the articles, exercises, and tips used. E-mail is also used as a communication form to distribute information and as an initiation to actively use the system. Direct or indirect routes can be used in the information contents, based on the desires of the content authors.

**4.3 Persuasive Software Features**

To initially discuss the PSD principles to be used to achieve the desired persuasive outcome, a workshop was held in which a group of six people with knowledge of the
PSD principles and the domain area gave vote on which principles they found the most important. This was a novel approach in designing persuasive systems. The people involved were nutritionists, informaticists, and information systems researchers within the research consortium. The same individuals were involved during the whole design process. Each person had ten points to distribute between the principles they thought should be most focused on. Of the 60 points total, the principles in the Primary Task category were given 24 points, while the Dialogue Support gained 16 points, the System Credibility gained 8 points, and 12 points were given to Social support. The principles with the most given points were tailoring (6 points) and self-monitoring (7 points), both in the Primary Task category. Reduction, reminders, and social learning each gained 4 points. Tunneling, rehearsal, suggestion, liking, and expertise gained 3 points each. 2 points were given to praise, rewards, social role, trustworthiness, social comparison, social facilitation, cooperation, and recognition, while personalization, surface credibility, real-world feel, and authority got a point each. No points were given to simulation, similarity, third-party endorsements, or verifiability.

According to the results of the workshop, the design principles in the Primary Task support category should clearly be emphasized. As the primary task of the users is to achieve efficient weight-loss, the design of the system should focus on supporting their process for permanent lifestyle change. Providing tailored information to user groups has its advantages, as for example providing additional nutritional information for those with issues in eating behavior might result in increased persuasion. In this kind of system the possibility for self-monitoring is also important as it allows users to keep track of their performance during the intervention. From the Dialogue support category, reminders especially are an efficient way to keep the users engaged. The System credibility support principles were paid less attention to, since it was stated that the parties involved were already familiar and trustworthy to the potential users. However, the provided information should indicate its scientific credibility by using reliable references and revealing the expertise of the providers of the information. The possibility for the Social support should also be considered, but rather by social learning from discussions than inducing competition or peer pressure within the system.

In the next design phase, the desired persuasive system qualities were distributed as different functionalities of the system (see Table 2 for correspondence):

- The home page, containing a brief description of the system, links to the most important functionalities, the topic and the tip of the week, and social cues of the behaviors of others.
- The week page, containing a weekly article, tip, exercise and discussions, along with links to other week pages concerning the same theme.
- The self-monitoring view, which includes all the software functionalities for following one’s change process: weight statistics, motivation diary, exercise diary and food diary.
- The info bank, collecting the previous intervention weeks in a list form, being able to be browsed ordered by the time or the theme.
- The toolbox, collecting the printable forms and other “tools” presented in the week pages, along with useful links to third-party web pages.
- The directions page, containing a simple instruction of the system use for the users to get started.
- The questions page, containing a form to ask experts questions, also showing the previous questions answered.
- The reminders received by e-mail
Table 2. System features and the corresponding design principles used

<table>
<thead>
<tr>
<th>Feature</th>
<th>PSD principles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Home page</td>
<td>Tunneling, reminders, suggestion, social facilitation</td>
</tr>
<tr>
<td>Week page and Info bank</td>
<td>Tailoring, tunneling, reduction, system credibility principles, social learning, social facilitation, normative influence</td>
</tr>
<tr>
<td>Self-monitoring</td>
<td>Self-monitoring, praise</td>
</tr>
<tr>
<td>Toolbox</td>
<td>Trustworthiness, expertise, tunneling</td>
</tr>
<tr>
<td>Directions</td>
<td>Trustworthiness, expertise, real-world feel</td>
</tr>
<tr>
<td>Questions</td>
<td>Real world feel, expertise, social role</td>
</tr>
<tr>
<td>E-mail reminders</td>
<td>Reminders, suggestion</td>
</tr>
</tbody>
</table>

*Home page*

The home page provides a starting point for the user to access the different functionalities of the system. The user is reminded and suggested to submit her self-monitoring entries, as well as to read the weekly contents. The user is notified about new comments in the discussions, and user statistics such as the users logged in during the current week. The home page also shows a progress bar visualizing the remaining time in the intervention with supportive waypoints.

*Week pages and Info bank*

In week pages, the user is presented with health information and exercises according to her phase of intervention and tailoring preferences. Each week contains a weekly article, exercise and a short tip. The weeks also contain a comment space for discussion among users. The user can browse back over the past weeks to re-read earlier contents. There is also an index page called Info bank for the available weeks, sorting them either by chronologically or by their subjects. The distribution of the information content into separate weekly portions implements the primary task support principles of reduction and tunneling. The principles of system credibility support were also widely utilized—the informational content provided references different health authorities to support perceived trustworthiness, expertise, authority, and verifiability. The discussion was designed primarily to support the social support guidelines: discussion among peers allows social learning, normative influence, and social facilitation within the system.

*Self-monitoring*

The self-monitoring section contains the data entries submitted by the user. For tracking her change progress, the user can submit consistent entries about her weight loss progress, memo notes about her feelings during the intervention, and her eating and
exercise habits. It is suggested that the user weighs herself once a week on the same weekday and submit the result. Supportive messages are conveyed during the submission. The submitted entries can be browsed in table form or as a visualized graph. It is also suggested that the user write “memos,” short messages related to her current thoughts and feelings about the intervention, and that she monitors her weekly exercise amounts. She may then submit entries on her exercise efforts, describing the type of exercise, the level of strain, and the amount of exercise done. The overall exercise performed during the ongoing week is presented for comparison with the recommendations from health professionals. The user is asked to keep a food diary, a tool often used in weight counseling. The submitted food entries contain information about the meal type, eating time, description, and the eating-place of a meal. After adding a meal, the user can tag an individual meal as “good” or “unnecessary,” resulting in reflection on her eating habits.

Toolbox

The toolbox section contains the individual tools such as surveys and forms introduced in the week pages, available for downloading and printing. Links to useful external resources are also collected here. The page is mostly static, but the contents can be made to appear to users once they have been mentioned in the weekly articles or exercises. Therefore, the applied guidelines were trustworthiness, expertise and tunneling.

Directions

In the directions section, instructions on using system are given. It is also explicitly stated what is the intention of the system and who are the responsible parties involved. The meaning of the different functionalities of the system is described, and it is openly told where the third-party contents used originate from.

Questions

The question section is used to provide a possibility for the users to ask questions about healthcare professionals about the themes discussed in the weekly contents. The questions and answers are then answered publicly, keeping the anonymity of the questioner. Through questions and answers the system sort of adopts a social role of a virtual health counselor.

E-mail reminders

E-mail reminders are used to reach the users outside the actual system use. Messages can be sent the users for example proposing them to go read the week article, as suggested in the Dialogue Support guidelines of suggestion and reminders.

4.4 The Development Process

4.4.1 Requirements specification

To gather all the requirements needed to produce a satisfying artifact for the problem domain, requirements engineering was conducted during the early part of the system development process. Requirements engineering includes the process of discovering the purpose the software systems is intended for by “identifying stakeholders and their needs, and documenting these in a form that is amenable to analysis, communication,
and subsequent implementation” (Nuseibeh & Easterbrock, 2000). Räisänen, et al. (2010) state that in the specific case of BCSSs, domain modeling and behavior modeling should be used at this stage to analyze the persuasion context as suggested by the PSD model. The requirements can then be communicated upon between the stakeholders. When a consensus is met, the final requirements can be agreed on.

![Use Case Diagram](image)

**Figure 3.** An early Use Case diagram

The most fundamental requirements for the system were decided on the meetings of the research consortium. Tools for modeling the requirement included Use Case diagrams (see Fig. 3) and gathering the agreed requirements in a table form (see Table 3). The utilized PSD guidelines were considered when gathering the requirements.

**Table 3.** An excerpt from the requirements specification document

<table>
<thead>
<tr>
<th>Requirement</th>
<th>PSD guideline</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 THE WEEK PAGE VIEW</td>
<td></td>
</tr>
<tr>
<td>2.1.1 The user can read the article of the week.</td>
<td>Tunneling, reduction, expertise, trustworthiness, verifiability</td>
</tr>
<tr>
<td>2.1.2 The user can read the tip of the week.</td>
<td>Tunneling, reduction</td>
</tr>
<tr>
<td>2.1.3 The user can read the exercise of the week.</td>
<td>Tunneling, reduction,</td>
</tr>
<tr>
<td>2.1.4 The tailored user can read the tailored</td>
<td>Tunneling, reduction, tailoring</td>
</tr>
<tr>
<td>exercise of the week.</td>
<td></td>
</tr>
<tr>
<td>2.1.5 The user can add an answer to the week’s</td>
<td>Self-monitoring</td>
</tr>
<tr>
<td>exercise.</td>
<td></td>
</tr>
</tbody>
</table>
4.4.2 Software Architecture Design

Software architecture is an abstraction of runtime elements of a software system during some phase of its operation (Fielding, 2002). Complex systems often contain many different levels of abstraction and different phases of operation, each with their own architectures. Fundamentally, software architecture describes a configuration of architectural elements and their interactions with each other. There have not been many prior studies on persuasive system architectures, but those that have been done include an architecture design for in-home monitoring technology to send persuasive messages to elderly adult diabetics (Chatterjee et al., 2002) and a framework for designing intelligent healthcare self-management systems (Mukhtar et al., 2012).

As a basis for outlining the architecture and the distribution of the system's architectural components, we adopt the persuasion context presented in the PSD model (Oinas-Kukkonen & Harjumaa, 2009). When first the desired intent, then the event, and finally the strategies are carefully identified and analyzed, the implementation of the system can take place through use context, user context, technology context, message, and route in order to fulfill the persuasion intent. These concepts can be actualized in the architectural components of use context, user context, user-system interaction, social interaction, and system-mediated messages (see Fig. 4). The described architectural layer separates between the conceptualization and the technical backend and supports the implementation of the desired persuasive features in different programming environments (Alahäivälä et al., 2013).

![Diagram of Architectural Design of Onnikka](image)

**Figure 4.** Architectural Design of Onnikka

Since the matters related to technological context are dependent on each implementation, but on the other hand, are provided by the surrounding system itself, the software features related to the persuasion event can be implemented in the corresponding software components of user context and use context. To accomplish a more explicit separation of different software functionalities, the specific features in the
use context related to social interaction and user-system interaction can be actualized as their own distinct architectural components. Persuasion strategies, actualized by the persuasive messages and their routes, can be implemented as the system-mediated messages component, to be presented to the appropriate users in their appropriate contexts of use. More precisely:

- The software features of the *user context* component should contain all the relevant information of the user, based on which the appropriate persuasion strategies can be put into action. A wide variety of differences between individuals' interests, needs, goals, motivations, abilities, attitudes, commitment, consistency, compromises, lifestyles, persistence of change, cultural factors, deep-seated attitudes, social anchors, and personality can be addressed in the modeling of the user context software element, giving the ability to present personalized and/or tailored content for most efficient persuasion (Oinas-Kukkonen & Harjumaa, 2009).

- The features of the *use context* component enable the environment, including the graphical user interface and the back-end data storage, to convey the appropriate persuasive strategies to the users. The implementation of this component should adapt to the problem domain in question.

- Setting goals and finding means to accomplish them in a systematic and effective way should be actualized in the *user-system interaction* components, allowing the system to record data about its users and the users to submit data into the system as well as to monitor their progress towards the desired change based on this.

- An important aspect in modern web services, interaction between users for giving and receiving peer support, should be implemented in the software component for *social interaction*. Its software features should allow discussion to be raised among the desired user group.

- The software component for the *system-mediated messages* define the content to be provided for the users in the selected form that is intended to result in behavior change. Depending on the situation, either direct or indirect routes may be used. The provided messages are then presented in the appropriate use context.

This proposed architecture can be used to implement the conceptualization of a BCSS as lower-level software components, independently of the problem domain or programming environment. Utilizing the architecture ensures that the use, user, and technology contexts, message, and route are actualized in the resulting system, provided that the desired persuasion intent has been first recognized. For most efficient persuasion, the appropriate design principles should be utilized in the implementation of the individual software components.

### 4.4.3 User Interface Design

The User Interface (UI) of the system was designed as an iterative process during which the designs were constantly evaluated in different phases.

*Layout mock-ups*

The interface was first roughly outlined with hand-drawn sketches. The early mock-ups then were made using Balsamiq Mockup application to demonstrate the implementation of the functionalities (see Fig. 5–6).
The navigation between the different functionalities related to the system contents was implemented as a vertical menu at the left side of the content container. The right side would then show each page’s main content. The header bar at the top would include the service logo as well as links to functionalities related to the system use, including changing the user setting, sending feedback or signing out.

Figure 5. An early mock-up of the Week page
Look and feel

The PSD guidelines of liking, surface credibility, similarity and real-world feel were considered when designing the final UI (see Fig. 7). To achieve a visually attractive appearance, the page layout and typography was based on a modified Twitter Bootstrap template, a common visual framework for websites.

The color palette was mainly based on shades of green, gray and yellow to achieve a fresh, but not too distracting visual atmosphere. Visual motifs following the current trends in web design, such as subtle background textures, gradients and noise, were used to attain an up-to-date feel. The top navigation bar is set to keep its fixed position so its always seen while scrolling the page contents. Metaphors to real life subjects were used in the design of the individual UI elements: the week tip’s container resembles a post-it note, while the box informing about the week’s subject hints to the color of a bus stop. Pictures were placed inside photo-like CSS (Cascading Style Sheets) frames.

In addition to the system-wide, de-facto sans-serif web typeface Helvetica, a Google web font called Courgette by Karolina Lach was used in the logo in the upper-left navigation bar and in the main heading of the front page (see Fig. 7) to imply a playful and friendly touch. An icon set called Glyphicons, included in Bootstrap, was used to throughout the system to familiarize the users to the different functionalities by referring them with appropriate symbols: for example, the week page links were preceded with a calendar symbol, and food diary links with a drinking glass.
To attain similarity to the users, pictures of environments and people from the region of Oulu were used throughout the system. The front-page features a rotating JavaScript slide show of randomized pictures from a collection of Oulu-based imagery (see Fig. 7). Similar pictures were also used in the middle of the weekly articles, aiming to make the users feel at home. The pictures were processed with vintage camera simulating color filters, often seen in popular photo sharing applications. Real-world feel was attained by including the logos of the organizations involved in the page footer of every page.

To remove a need for a separate mobile client and to make sure the system would work as well in high-resolution as in low-resolution devices, responsive design techniques were used to make the layout adapt to different screen sizes. This was made possible by utilizing CSS media queries for handling custom styling rules based on the target device screen size. When accessing the page from a mobile browser, the client would be served a layout with elements stacked instead of floating, and the bandwidth-demanding image carousel element in the front page would be hidden (see Fig. 8).

![Image](image.png)

**Figure 8.** The front page accessed from an iOS emulator

*Login page and password recovery*

The landing page (see Fig. 9) was designed as simple as possible to reduce cognitive load of the new users. It only features a sign-in form with text field for e-mail and password, a ‘remember me’ checkbox to keep the user logged in between sessions, a submit button, a link to password recovery and the logos of the parties involved. The password recovery (see Fig. 10) similarly only has a brief instruction text, an input field for e-mail address and a submit button.
Figure 9. The login page

Figure 10. The password recovery page

Home page

The home page (see Fig. 7) was designed to catch the user’s attention immediately, explaining the different functionalities of the system. The content in the home page is
presented in two columns on green background. On the left there is the carousel with scrolling pictures with description of the features of the system and a box showing the topic of the current week and a link to the weekly page. On the right there are three box one on the other: on with the ‘odometer’ with a progress bar describing the intervention progress and statistics about visited users and given comments, one with links to the main functionalities, and one showing the current week’s tip.

**Week pages and Info bank**

The week page content is presented in two columns. In the left column there is an introductory text to the week’s subject, and the weekly article, exercise and discussion separate boxes. The article and exercise boxes have backgrounds with different shades of green, and the discussion box is dark gray (see Fig. 11–13). On the right column there are the navigation buttons for the previous and next week, the week’s tip shown in a post-it note, and a list of links to the other weeks with a related theme. The info bank (see Fig. 14) shows a list of the past weeks, with tabs to choose the list ordered chronologically or based on the theme.

**Self-monitoring**

The different self-monitoring functionalities (see Fig. 15–18) are shown as tabs in the horizontal navigation. On each page the entry submitting form is on the left. There is also a blue info box, which shows additional instructions upon hover. The submitted entries are shown on the right, the weight entries as a graph and the rest in a table form.

![Image](image.png)

*Figure 11. The weekly article and tip*
Figure 12. The weekly exercise

Figure 13. The weekly discussion
Figure 14. The info bank

Figure 15. Weight monitoring page
Toolbox

The toolbox (see Fig. 19) presents the downloadable tools on the left, and useful links on the right.
Directions

The directions page (see Fig. 20) shows a brief description of system and instructions on using it.

Questions

The question page (see Fig. 21) shows the given questions and answers on green boxes on top of each other on the left column and a form to submit a question on the right.

Settings

The setting page (see Fig. 22) contains forms to set the user nickname, height and target weight and to change the password.
4.4.4 Implementation of the Software Components

In the implementation of the software components in Onnikka, model/view/controller (MVC) approach was utilized to separate the user interface and the business logic, and
the representational state transfer (REST) design approach was followed to provide an easily accessible interface for performing operations on application’s resources.

The MVC design pattern is a widely used methodology to distribute the software functionalities into separate interactive application components: models, views, and controllers (Krasner & Pope, 1988). Models are the domain-specific simulations or implementations of the application’s central structure; views display graphical representations of the data requested from the models; and controllers contain the interface between the associated models and views, as well as handling the user interaction from the input devices. Using the MVC design pattern makes developing and maintaining an application easier, since the application’s user interface can easily be modified regardless of the data structures and the business logic within. However, Jeff and Rayfield (2001) have stated that since the MVC pattern is expressed in terms of an interactive application running in a single address space, applying it to web applications can be challenging since they are often deployed in various client/server architectures. They present the concept of flexible web-application partitioning to enable partition-independent web applications to be built. A possible deployment architecture supporting such partitioning is the thin-client approach, in which the model and the controllers reside in a single address space on the server, and generate the views to be rendered on the client side.

REST is a “coordinated set of architectural constraints that attempts to minimize latency and network communication, while at the same time maximizing the independence and scalability of component implementations” (Fielding, 2002). It has become the de facto standard for service design in Web 2.0 applications, although REST is not specifically a defined W3C standard, but rather a widely used design technique with several interpretations. In ‘RESTful’ services, the application’s resources are identified and resolved by a particular uniform resource locator (URL), describing the target protocol, host address, application path, resource type, and resource identifier. As such, the application’s data content can be accessed by performing particular hypertext transfer protocol (HTTP) requests on the resource specified in the URL. REST also allows HTTP headers to be used to provide a request context around operations, so that the request can result in a desired media type specified in the header (Battle & Benson, 2008).

![Figure 23. The key software components in Onnikka](image-url)

In Onnikka, each user-facing component is modeled as a resource, and each resource is identified by and resolved through a URL. These RESTful components may then internally implement the MVC pattern. The distribution of Onnikka’s RESTful
resources and their relationships (see Fig. 23) follows our suggested architecture design for persuasion context. The user context component is implemented as the User resource. The use context component was implemented as an external content management system (CMS) software component called Refinery, which provided the environment to present the health information to the users and an interface for nontechnical personnel to update the information contents. The information itself was stored in the Week resource, corresponding to the system-mediated messages component. The Comment resource implements the social interaction component, and the user-system interaction component is realized as the Exercise entry, Food entry, Memo entry, and Weight entry resources. These are used as tools for users to monitor their progress during the intervention, and receive feedback from the system.

Following the MVC pattern, there is a separate model and controller file for each resource, and the RESTful controller actions have their corresponding view files, either HTML or JavaScript with embedded Ruby language. The instances of the models are mapped into the database upon creation. This way, there is no need to design a separate database schema, since the database entries are created based on the relations between the models. The main components are now described in more depth along with the code describing their model definitions. Since the controllers of the resources quite generically implement the REST principle, we don’t describe their code implementations explicitly. The Ruby-embedded HTML or JavaScript code for the views is also left out. Besides the components described here there were also software components for handling more general web service functionalities such as storing sessions and password recovery, but since they are not especially related to the studied BCSS aspect, they are not discussed more thoroughly in the scope of this thesis. The full source code for the system is available for viewing on the GitHub service.

**User**

The User resource contains the information about the users required to provide them the appropriate content and persuasive features of the system. The basic information to authenticate the user is included: the users use their e-mail addresses as usernames and the set password is stored in a secure format. The starting day of the intervention is stored to be able to present the user the appropriate use context according to their phase of intervention. The information about the tailoring preference for additional eating behavior content for the user is included. The target weight submitted by the user is stored to allow goal setting and personalized feedback on the weight loss progress. The given user height is used to pass on to calculate users’ body mass index (BMI) in their weight entries. Users can also set a nickname to be used in the discussions.

The user model also contains the methods for sending the e-mails to each user when needed. These methods include sending the confirmation mails, password reset links, weekly email, optional weekly reminder and optional weekly tip for tailored users. In its entirety, the user model is defined as follows:

```ruby
class User < ActiveRecord::Base
  has_secure_password
  has_many :weight_entries, dependent: :destroy
  has_many :date_memos, dependent: :destroy
  has_many :comments, dependent: :destroy
  has_many :food_entries, dependent: :destroy
```
has_many :exercise_entries, dependent: :destroy

before_save { |user| user.email = email.downcase }
before_save { create_remember_token(:remember_token) }

validates :name, presence: true, length: { maximum: 50 }
VALID_EMAIL_REGEX = /A[\w+-.]@[a-z\d\.-]+\.[a-z]+$/i
validates :email, presence: true, format: { with: VALID_EMAIL_REGEX }, uniqueness: { case_sensitive: false }
validates :password, presence: true, :on => :create
validates :password_confirmation, presence: true, :on => :create
validates :password, length: { minimum: 6, allow_nil: true }, :on => :update
validates :start_date, presence: true

def send_confirmation_mail
  create_remember_token(:password_reset_token)
  self.password_reset_sent_at = Time.zone.now
  save!
  UserMailer.signup_confirmation(self).deliver
end

def send_password_reset
  create_remember_token(:password_reset_token)
  self.password_reset_sent_at = Time.zone.now
  save!
  UserMailer.password_reset(self).deliver
end

def self.weekly_email
  @users = User.all
  @users.each do |u|
    if (((Time.now.to_i - u.start_date.to_time.to_i) / 604800) < 53)
      UserMailer.weekly_email(u).deliver
    end
  end
end

def self.weekly_reminder
  @users = User.where("last_seen < ?", Time.now.beginning_of_week)
  @users.each do |u|
    if (((Time.now.to_i - u.start_date.to_time.to_i) / 604800) < 53)
      UserMailer.weekly_reminder(u).deliver
    end
  end
end

def self.weekly_tailored_tip
  @users = User.where("tailored = ?", true)
  @users.each do |u|
    if (((Time.now.to_i - u.start_date.to_time.to_i) / 604800) < 53)
      UserMailer.weekly_tailored_tip(u).deliver
    end
  end
end

private

def create_remember_token(column)
  begin
    self[column] = SecureRandom.urlsafe_base64
  end while User.exists?(column => self[column])
end
Refinery CMS

In modern web services, it is usual to handle all the functionalities related to adding or editing content through a content management system, only accessed by the administrative users. This allows administrators to make changes to pages without the need for profound web development or programming skills. To provide an interface for the administrators to bring and edit content into Onnikka and the resulting web pages from the database contents, an external software component Refinery CMS was used (see Fig. 24).

The way Refinery works, resources that are desired to make editable via its web interface are added as separate modules into the application. Refinery then generates interfaces for editing them. The editor for the text parts includes basic text processing functionalities. Refinery also then handles composing the resulting pages from the given contents. The resources implemented as Refinery modules were the week pages and the questions page.

![Refinery editing panel for a week page](image)

**Figure 24. Refinery editing panel for a week page**

**Week**

The week resource, editable in the CMS, includes the weekly introduction, main article, exercise, tip and e-mail summary provided by the health professionals. Each week may also contain an additional tailored exercise and e-mail reminder for the users with tailoring enabled. Each week contains a running number so that it is shown during the proper phase of intervention, and has a designated theme based on which it can be categorized in the info bank page. The themes are stored in a separate database table and are referenced with their designated id.
class Week < Refinery::Core::BaseModel
  self.table_name = 'refinery_weeks'


  belongs_to :theme, :class_name => '::Theme'
  has_many :comments, :class_name => '::Comment', :dependent => :destroy
  has_many :week_notes, :class_name => '::WeekNote', :dependent => :destroy

  validates :title, :presence => true, :uniqueness => true
  validates :number, :presence => true, :uniqueness => true
end

Week Note

The week note resource stores answers to the weekly exercises added by the users. Each week note entry is private, seen only by the author. The week note is also editable later on. The week note controller action responds to requests in JavaScript, so adding and editing the notes inside the week pages can be done without page reloads. In a week note, user id, week id and the content text is stored.

class WeekNote < ActiveRecord::Base
  attr_accessible :content, :user_id, :week_id
  belongs_to :user
  belongs_to :week, :class_name => 'Refinery::Weeks::Week'

  validates :content, presence: true
  validates :user_id, presence: true
  validates :week_id, presence: true
end

Comment

The social interaction component was actualized in the comment resource. The users can leave comments on the week pages, where discussion related to the weekly subject is encouraged. In the comments, user id, week id, content and a timestamp of the comment is stored. The maximum length of the comment was set to 1000 characters, making it possible to give moderately long statements.

class Comment < ActiveRecord::Base
  attr_accessible :content, :user_id, :week_id
  belongs_to :user
  belongs_to :week, :class_name => 'Refinery::Weeks::Week'

  validates :content, presence: true, length: { maximum: 1000 }
  validates :user_id, presence: true
  validates :week_id, presence: true

  default_scope order: 'comments.created_at DESC'
end
**Weight Entry**

The weight entry resource implements the weight entries added by the users. Each weight entry includes the user id, date and the weight figure:

```ruby
class WeightEntry < ActiveRecord::Base
  attr_accessible :date, :weight

  belongs_to :user
  before_save :calculate_bmi

  #Validations for Date Entry attributes
  validates :user_id, presence: true
  validates :date, presence: true
  validates :weight, presence: true, :numericality => {
    :greater_than_or_equal_to => 40,
    :less_than_or_equal_to => 300
  }

  default_scope order: 'weight_entries.date DESC'

  scope :previous, lambda { |d| {:conditions => ['date < ?', d.date], :limit => 1, :order => 'weight_entries.date DESC'} }
  scope :next, lambda { |d| {:conditions => ['date > ?', d.date], :limit => 1, :order => 'weight_entries.date DESC'} }

  def calculate_bmi
    unless self.weight.nil? || self.user.height.nil?
      bmi = self.weight / (self.user.height / 100.0)**2
      return bmi
    end
  end
end
```

The weight entry model also contains a function for calculating the user’s BMI at the time of the entry. This could then be utilized for example in showing the BMI next to each entry in listing the weight entries:

```html
<td class="numeric">
  <%= weight_entry.weight %>
</td>
<td class="numeric">
  <%= sprintf "%.1f", weight_entry.calculate_bmi %>
</td>
```

The current BMI of the user could also be easily shown by calling the function on the latest weight entry.

**Food Entry**

The food entries were used to contain the individual meals eaten. Upon time of insertion the user id, meal type, description, date, time and place are stored as entered. The review attribute is null by default, and can be set to ‘good’ or ‘bad’ afterwards from the interface. This ensures users’ reflection on their eating habits.

```ruby
class FoodEntry < ActiveRecord::Base
  attr_accessible :amount, :date, :description, :food_type_id, :time, :place, :review
```
belongs_to :user
belongs_to :food_type

validates :user_id, presence: true
validates :date, presence: true
validates :description, presence: true
validates :food_type_id, presence: true
validates :time, presence: true

default_scope order: 'food_entries.date DESC, food_entries.time DESC, food_entries.created_at DESC'
end

**Memo Entry**

The memo entries contain the user-submitted notes about their current motivation and feelings. A memo entry contains user id, timestamp, the content text and a motivation value from 1 to 5, which are shown in the interface as emoticons from a sad to a happy face.

class DateMemo < ActiveRecord::Base
  attr_accessible :content, :motivation
  belongs_to :user

  validates :content, presence: true, length: { maximum: 1000 }
  validates :user_id, presence: true

  default_scope order: 'date_memos.created_at DESC'
end

**Exercise Entry**

The exercise entries contain information about the exercising acts of the users. Each entry has user id, exercise type id, exercise name, date and amount attributes. Exercise type names are located in a separate table, although we only had two options: light or heavy exercise. The amount is given in minutes and stored as an integer to the database. To view the amount in hours and minutes in the interface if the amount is greater than one hour, there is a helper method available in a separate helpers file to return the proper time format.

class ExerciseEntry < ActiveRecord::Base
  attr_accessible :amount, :date, :exercise_type_id, :exercise_name

  belongs_to :user
  belongs_to :exercise_type

  validates :user_id, presence: true
  validates :exercise_type_id, presence: true
  validates :exercise_name, presence: true
  validates :date, presence: true
  validates :amount, presence: true, :numericality => { :only_integer => true, :message => "on annettava minuutteina" }

  default_scope order: 'exercise_entries.date DESC, exercise_entries.created_at DESC'
  scope :previous, lambda { |d| {:conditions => ["date < ?", d.date], :limit => 1, :order => "exercise_entries.date DESC"] } scope :next, lambda { |d| {:conditions => ["date > ?", d.date], :limit => 1, :order => "exercise_entries.date DESC"] } end
Questions and answers functionality was implemented as a Refinery module. The form in the questions page sends the question into the e-mail of the corresponding expert. The expert can then log into the CMS admin interface and add the question and answer to the system. Each question contains the question, the answer, a title and a timestamp, based on which the questions are sorted.

```ruby
module Refinery
  module Questions
    class Question < Refinery::Core::BaseModel
      self.table_name = 'refinery_questions'
      attr_accessible :title, :question, :answer, :position
      acts_as_indexed :fields => [:title, :question, :answer]
      validates :title, :presence => true, :uniqueness => true
    end
  end
end
```

**E-mail reminders**

E-mail messages were sent as reminders for users to actively use the service. A maximum of three messages to each user was sent during a week.

On Monday at 10 AM everyone in the intervention would receive a message containing the weekly topic of the ongoing week, a brief summary of the week’s subject and a suggestion to read the article in the system and update their self-monitoring entries. If the user had not signed in by Thursday 10 AM of the same week, a new email reminder would be sent, reminding the user to read the weekly contents. In the mail it would also be mentioned that if the user forgot their password they could recover it using the password recovery functionality at the sign-in page. Finally, at Friday 10 AM a tailored tip for enhanced eating behavior would be sent to the users with tailoring enabled. These were only included in a set total of 23 weeks. See Fig. 25 for a flow chart of the e-mail reminders functionality.
Figure 25. A flow chart of the e-mail reminders in Onnikka

For scheduling the reminders, we utilized the job scheduler program Cron in Linux. The timing of the reminders was set in the schedule.rb file in the configuration directory of the application, contents of which would upon deployment to the production environment be converted into proper format and updated into the server’s crontab file. The contents of the schedule.rb are as follows:

```ruby
every :monday, at: "10:00 AM" do
  runner "User.weekly_email", :environment => 'production'
end

every :thursday, at: "10:00 AM" do
  runner "User.weekly_reminder", :environment => 'production'
end

every :friday, at: "10:00 AM" do
  runner "User.weekly_tailored_tip", :environment => 'production'
end
```
5. Evaluation

To evaluate our resulting artifact, we now analyze the results from the conducted user testing and expert evaluations.

5.1 User testing

The functionalities of the system were demonstrated during three months before the actual deployment within a test group of 15 test users. The test users were mostly researchers of the information systems and internal medicine disciplines. The test user group was instructed to try out all the different functionalities of the system and report their perceptions and possible issues found by e-mail or by a feedback form within the system. The given feedback was then gathered and discussed, and next iterations of the design and development process were then designed based on it.

In addition to the free-form group testing, user testing was conducting using the observation method. A tester was instructed to follow the given scenario:

1. Read the article, exercise and tip of the on-going week.
2. Participate in the on-going week’s discussion.
3. Set yourself a target weight.
4. Add weight entries for three preceding weeks.
5. Write a memo entry.
6. Add an entry to the exercise diary.
7. Add an entry to the food diary.

Notes were gathered during the duration of the test session about possible countered issues in the system use, and the tester was briefly interviewed about his perceptions during the session. The opinions of the test group were also gathered during and after testing period. A part of the feedback given was related to the health information contents, which is not discussed in the scope of this thesis. The rest of the feedback was labeled being related to either the UI or the functionalities of system, or the software bugs found. Based on the feedback, we constantly updated the system according to the revised requirements during the test phase.

UI feedback

A majority of the feedback was related to the user interface related issues:

- “It took some time to notice the possibility for adding answers in the weekly exercise. At first I thought they were supposed be pondered without storing any answers. It could be mentioned that the answers can be stored under the question or the answer form could be shown by default.”
- “My thoughts about Onnikka are positive as a whole. I liked many of the details such as pop-up info boxes, page appearance and navigation, many details of the interface and the odometer. The is also a whole bunch of information available both directly and through links.”
“Should the review buttons in food diary be explained in someway, like this: ‘if you are content with your eating, press the green thumb up, otherwise press the red thumb down’?”

“Could the topic of the week appear same yellow color as seen in the front page?”

The odometer nicely mentions there are four comments on the weekly discussion. But how do I get to the discussion? Could it happen straight from the front page?”

The test users found the UI simple and appealing. Some of the functionalities were considered to be too easily missed, or needed more comprehensive explanation. Some additional shortcuts and small UI details were also suggested. As much as the people’s perceptions of a beautiful interface differ, we tried to take into account the comments when constantly trying to reach a better UI in the iterations.

Functionalities

Comments were also given how to enhance the functionalities of the system:

• “It would be nice to be able to edit your comments in the discussion afterwards.”
• “You should be able to edit the weekly exercises afterwards.”
• “The target weight should be shown in proximity to the weight graph.”

The given comments about the functionalities were compared with our original design intentions. It is apparent that some of the features making the use more convenient would conflict with our intentions. For example if the users could delete or alter their comments afterwards, it could encourage anti-social behavior in the discussions, or subtract from our eventual research data. The suggested features that would not conflict with our intentions, like editing the exercise answers and showing the target weight next to the graph were implemented, successfully improving the user experience.

Bugs

The test users also reported some bugs found in the system:

• “Nothing visible happens when clicking the send button in discussions, although the message is apparently stored. As a result of this you accidently send multiple messages.”
• “The pop-up info box does not appear completely on my screen, the bottom part was left under the browser window.”
• “If you use the print button to print the page contents it comes out wonky, does not appear confident.”

We inspected carefully the reported issues and found that some of the functionalities didn’t work as intended on older web browsers, such as Mozilla Firefox 4.0. It is unfortunate that as web standards evolve, compatibility issues with older browsers emerge. But considering the future development of the web, it was considered best to follow the modern standards and ask the users to update their browsers instead of spending resources on trying to accomplish all-inclusive compatibility.
Some issues were also related to the responsive layout: in some of the smaller displays the elements would not fit to the screen correctly. These issues were solved with more comprehensive testing on different screen sizes.

Finally, some of the issues were related to inadequate design or implementation of certain software features. In some cases, like the ‘print’ button functionality, it was considered best to remove the feature as a whole, rather than leave a partial implementation in the system.

5.2 Evaluation of the Persuasive Features

In addition to our internal discussions, two people outside the research consortium with extensive knowledge on the PSD model were asked to evaluate the system using the PSD guidelines as heuristics. The results were gathered and compared with our design intentions.

5.2.1 Primary Task Support

Reduction and tunneling

The guidelines of reduction and tunneling clearly corresponded with the planned intervention process:

- “Reduction is strongly used.”
- “The user learns something new every week and does exercises that awake ideas and hopefully help to make the change.”
- “By following tasks and keeping daily or weekly diary user can monitor easily the overall progress and change step by step the bad habits into more healthier.”
- “The odometer informs about the duration of the journey and also shows it in a graphical form”

Distributing the articles and exercises into segments of 52 weeks makes it easier for the users to incrementally digest the information and reflect it to their behaviors. The diary entries help people understand their behaviors as a collection of individual acts such as daily meals and exercising actions.

Tailoring and personalization

The tailoring in the system is not explicitly visible to the users, so the evaluators also did not recognize it. Tailoring was still an important feature of the system, since it allowed providing additional exercises for the users with issues in eating behavior. This will probably also be a major concern in the development of future BCSSs, so the results of channeling tailored information in Onnikka should be properly studied afterwards. Ways to implement more extensive tailoring and personalization according to the user profiles should also be thought of. For example, we discussed adding a feature where tailored users could tag their food diary entries as binge eating, uncontrolled eating or emotional eating, based on the studies by Keränen (2011).

Self-monitoring

Self-monitoring was clearly one of the major features in the system:
• “Self-monitoring in Onnikka is very well implemented.”
• “There is a weight-monitoring diary that shows the results and motivate to keep up the good pace.”
• “There is a notebook feature to keep record of daily feelings.”
• “There are also food and exercise diaries to record and compare these with experts.”

The users can keep track of their performance by browsing their weight entries, motivation entries, exercise diary entries, food diary entries and exercise answers submitted. This allows them to concretely see the change in their behavior during the intervention. Especially the graphic representation of their weight plot can be an evident demonstration of the change.

5.2.2 Dialogue Support

Praise

Giving praise was used quite sparsely in the system. When adding a weight entry with a weight loss more than 0.5 kilograms since the last entry, motivating cheer was given. Achieving one’s target weight was also praised. Other than that, we paid close attention to the tone of the notifications messages and the instructions given. For example, if the user submitted wrong authentication credentials or tried to type an erroneous self-monitoring entry, a friendly request was given to revise the given information. In the future development, more explicit ways of giving praise could be thought of. For example giving virtual rewards for achieving their weight-loss goals could appear motivating to users.

Reminders and suggestion

Reminders and suggestion were widely used both within the web interface and in the e-mails sent to the users. In the weekly e-mails, the users are prompted to read the weekly contents and add to their self-monitoring entries. The additional e-mail is sent at the end of the week if the user is not logged in before that, serving as a reminder. There are also subtle suggestions what to do in the system at the front page, and the info boxes at the self-monitoring pages also give information in order to make suggestions how to act. It is notable that too consistent reminding may irritate some users. Therefore it should be further evaluated if for example the amount of the e-mails sent is appropriate.

Liking and similarity

The appearance and the overall feel of the system were considered pleasurable by the evaluators:

• “The system looks pleasant. It takes a moment to find everything, but that is natural when using something new. Many functionalities add to the pleasantness for me, like the tips showing up upon mouse hover.”
• “I like the metaphor about the journey to healthier life.”
• “Plenty of pictures from Oulu are used in the system, so there is something familiar in it right from the beginning.”

Appealing visual interface adds to the persuasiveness of the system, so paying attention to the layouts, typography and the color palette is necessary. The decision of utilizing a ready-set front-end framework proved successful, since it provided a useful set of tools
for building an attractive user interface. Utilizing the photos from the local region was also beneficial to attain a feel of familiarity.

**Social role**

It was noted the system adopts a social role of sorts in the form of an expert answering the questions asked:

- “It is possible to ask questions from experts. The answers are shown to everyone anonymously.”

We decided there would not be any individual counselor present in the Onnikka, but still wanted to provide a possibility for users to ask question from an expert. The presented answers conceal the identity of both the asker and the expert, but the questions functionality is still based on the expertise of the answerer.

### 5.2.3 System Credibility Support

**Trustworthiness, expertise and surface credibility**

The system was found reliable and trustworthy:

- “The information provided by the system seems honest and trustworthy.”
- “There is professional information about weight loss available, not just simple “eat less, exercise more” type of knowledge. In addition, there is diverse third-party information available through links. There also were some book recommendations.”
- “The system feels reliable, because there are no advertisements or anything redundant.”
- “Onnikka is free of advertisements banners and commercial spam.”

Much of the reliability of the system comes from the professionalism of the information contents provided. But a convenient appearance free of advertising and any surplus features also clearly had an effect.

**Real-world feel, authority and verifiability**

Revealing the purpose of the system and the organizations behind it openly were considered positive:

- “The affiliated parties are revealed both in the directions page and in the footer logos.”
- “There are links to the national institute for health and welfare, and information is drawn from many sources.”

The information coming from an ‘official’ source is considered trustworthy and persuasive. Openly revealing the information sources and the affiliated parties is therefore clearly beneficial.
5.2.4 Social Support

Normative influence, social facilitation and social learning

The evaluators felt the social support principles could have been utilized more widely in the system:

- “I’d like to see more means for gathering together and co-operational action.”
- “I’d like to see more of these features like forum, chat, share diaries, etc.”
- “Shared diaries and getting positive feedback from other users and experts might be a good add-on.”
- “What I’d like to see are success stories of other users who share the journey.”

The social interaction in our system was mostly based on the weekly discussions, although there were also indicators of the amount of users present on the front-page odometer as a normative influence cue. Modern social web services provide many ways for their users to interact socially, and it should be thoroughly inspected which as these could be utilized in this kind of system. We purposely wanted to keep the discussions related to the weekly themes instead of a general discussion forum, and all the diary entries were kept private for the sake of the users’ privacy. Allowing writing public blog-like entries or publishing the success stories of the users could be beneficial as long as the privacy concerns are kept in mind.

A common practice in modern web services is to add functionalities related to sharing content to third-party social networks such as Facebook, Twitter and Google+. This can result in delivering normative influence, social facilitation and social learning to the users’ social circles outside the originating service. Since weight loss and lifestyle changes are subjects the users do not necessarily want to openly share to their friends, we ended up not having a possibility to share any personal details or user data to social networks. In the end, we decided to only have a possibility to share the on-going week’s tip to Twitter with an appropriate hashtag via an URL-based request. This way, the users could induce social learning from Onnikka to their own Twitter followers without having to worry about compromising their privacy.

Social comparison and competition

The social comparison and competition on the other hand were not considered to be appropriate features in this specific environment:

- “I don’t see any beneficial aspects in scoreboards or top lists.”
- “Losing weight as a competition is not healthy. For that purpose there are other concepts.”

Social comparison and competition were one of the most contradictory features discussed when designing the system. Modern social web services often encourage for comparing your performance with the performance of others. So-called ‘gamification’ features such as hi-score lists are commonly considered valuable motivational tools for increased system use (Deterding et al, 2011), but since we operate under the very sensitive subject of lifestyle change and weigh loss, encouraging competition or comparison among users might have very negative results overall. To avoid causing any negative feelings, we chose to drop all software features that would have lead to social comparison or competition. But we acknowledge that in a domain area other than this they might prove beneficial.
6. Discussion

In this chapter, the qualities of the resulting artifact and its design process are discussed regarding to our research intentions and theoretical background.

Design considerations

The PSD model was affirmed to be a useful tool in both design and evaluation of the system. Understanding the persuasive postulates and analyzing the persuasion context created a solid starting point to start deciding on the system qualities. The role of the postulates was not really emphasized during the design process, though. They were treated more as laid down assumptions than as actual guidelines. The design guidelines for primary task support, computer-human dialogue support, perceived system credibility and social influence instead helped shaping the individual software functionalities for efficient persuasive influence. Using the PSD model also made it easier to communicate on the system qualities within the research consortium consisting of people from different disciplines.

This study was based around the PSD model, but we acknowledge that it might have been beneficial also to utilize some other theoretical constructs on par with it in the design process. Since weight loss and maintenance is such a sensitive subject, conducting an ethical analysis using Value Sensitive Design or Participatory Design (Davis, 2009) could have helped better account for human values in the designs. To design for optimal perceived user experience, constructs such as Webflow (Oinas-Kukkonen, 2011) could have been utilized. There may also have been multiple other models and heuristics, for example related to usability and user experience, that could have proven useful.

Architecture considerations

We suggested that when designing BCSSs, the software components would be distributed according to architectural components of use context, user context, user-system interaction, social interaction and system-mediated messages. This way all the different aspects of the persuasion context are realized in the system. The resulting system shows a full-fledge BCSS can be implemented by applying the architecture.

It is notable that our architecture is presented at a moderately high level, and more detailed description of the characteristics of each architectural component would be beneficial. It is also difficult to say how generalizable the architecture is to different BCSS domain areas and technical implementations without further research and applications of the suggested architecture.

UI considerations

As a basis for the interface design of Onnikka, the Twitter Bootstrap front-end framework was used. This turned out to be a timesaving practice to produce an appealing interface that follows web standards. Its responsive qualities helped making the service available to mobile users without a need for native mobile client. The layout
was applied a customized color palette and many UI practices that follow current trends in web design were put into use. The test users found the layout pleasant and appealing.

Utilizing a ready-made front-end framework may well be beneficial in many cases of BCSS development. It saves time and resources and helps creating appealing layouts if there is no designated web designer available. But is should be noted that without customizing the layout out to each project’s needs, the resulting pages may end up looking generic and dull. The early adopters of web services appreciate following the current trends in UI design, but it may well be that less experienced users may not at first understand all the metaphors and common practices used.

*Technical considerations*

The details of the technical implementation of the system will hopefully prove useful for the future BCSS research and development. The system was developed using the Ruby programming language and the Ruby on Rails framework. It suited well for this sort of agile development, where the software features were consistently iterated. The framework has extensive documentation on the web and an active user community, which makes it easy to learn. It also encourages many of the best practices in web development like the MVC and REST patterns and the “Don’t Repeat Yourself” principle in order to produce high quality, reusable code. The GitHub service used for version management and the Red Hat Linux distribution used as the server’s operating system also suited the needs of the project well.

A downside to using Ruby on Rails is that since it automates many phases of the development, researchers and practitioners may found it difficult to afterwards understand how some functionalities of the system actually work. This problem also occurs when using third-party software components like the Refinery CMS. For more detailed technical inspection of BCSS development, it could be beneficial to create all the functionalities ‘from scratch’, without help from a ready-set code frameworks or software packages.

There probably is also room left for improvements in the implementation of the software components. Explicit system profiling should be conducted to find out bottlenecks in complex database queries to improve the loading times of the pages. It would be beneficial to implement some kind of constant polling for changes to notify users about new content appearing in real-time instead of after page loadings. Finally, it should be investigated if it would be beneficial to implement the whole front-end section using some modern JavaScript framework, which seems to the current trend in web services, making the system more platform-independent and easier to utilize for future applications.

Security is a major concern in this type of system where the users will submit sensitive data in to the system, the privacy of which cannot be compromised. To address this issue we forced the Secure Sockets Layer (SSL) protocol to be used upon every connection to the system. The user inputs are validated and sanitized to prevent entering malicious database queries. Many of the vulnerabilities are luckily already taken into account in the Rails framework. The possible vulnerabilities of each technology used should always be regarded when developing BCSSs.

In summary, the addition to the scientific knowledge base from our study is the detailed description of the actions needed to produce a full-fledge BCSS. This will hopefully make way for producing generalizable patterns and guidelines for future BCSSs.
development and research. Especially our suggested software architecture for BCSSs could prove a valuable starting point for implementing new systems in the future, similar to the PSD model for designing them, as verified by this study.
7. Conclusion

In this thesis, we have examined a process of designing and implementing a novel weight loss and maintenance system by applying design science research methodology and utilizing the prior research on persuasive systems.

As theoretical background, we adapted the concepts of behavior change support systems and persuasive systems design. Persuasive technology research is a prominent area today, studying how technology can shape our attitudes and behaviors. Many social psychological theories can be utilized to study the persuasion of information systems, such as the information processing theory, cognitive consistency theory, elaboration likelihood model, influence techniques approach, and coactive approach. Novel constructs such as the Outcome/Change design matrix and the Persuasive Systems Design model have been also created. The PSD model can be utilized in designing and evaluating persuasive systems. According to it a persuasive systems design process should contain the phases of acknowledging the persuasion principles, analyzing the persuasion context and the designing the persuasive software features, related to primary task support, computer-human dialogue support, perceived system credibility or social influence.

As a research method we utilized the design science approach in information systems, where a viable artifact is produced as a result of the research. Our entry point for the research was the need more detailed system descriptions of BCSS development process, stated in the research agenda for BCSSs. Our objective of solution was to produce a full-fledge BCSS in the weight loss and lifestyle change domain, operating within the PrevMetSyn research project. The design and development process utilized the theoretical methods for persuasive systems design and was comprehensively documented. We demonstrated the resulting software artifact by conducting user testing on the system and evaluated it by analyzing the results of the testing and the given expert evaluations based on the PSD model.

We implemented our software artifact Onnikka following the persuasive system design process. We began our design process by acknowledging the persuasion postulates and analyzing the persuasion intent, event and strategies. The users of Onnikka are weight losers with different goals and needs. Some for example require special attention to monitoring their eating habits. Our system use context should provide them a way to acquire appropriate health information and give tools for monitoring their change progress. The technological context enabling this should be independent of the user platform. The desired persuasive features were decided on to be applied to the different functionalities of the system. These were then specified into actual software requirements. We designed software architecture for our system, consisting of user context, use context, user-system interaction, social interaction and system-mediated messages components. We created the user interface, utilizing the Bootstrap framework and common web design practices. We implemented the individual software components of the system following MVC and REST approaches and our suggested software architecture, using Ruby on Rails web development framework.

To evaluate our resulting system, we analyzed the feedback from a three-month testing period of the system and two given expert evaluations. The results of the user testing
gave us valuable observations of the UI, system functionalities and bugs found. Following the DSRM process, we iteratively updated the software features to match the enhanced requirements. The PSD model was confirmed a useful evaluation tool regarding the persuasive features of the system. According to the evaluation, we made successful design choices to support the users in their carrying out their primary tasks by distributing the information contents into weekly sections, providing tailored information and encouraging self-monitoring behaviors. The appearance and planned reminders supported the dialogue support guidelines, but there might be room for enhancement in for example praising the user more frequently. The system was considered credible and trustworthy, much due to the professionalism in the contents and being open about the sources of the information and the parties involved. In providing social support, there could be benefits in developing more social features beyond the discussion section. This should be done keeping the users’ privacy and the sensitivity of the subject in mind though.

In the discussion section, we reviewed the different sections of our artifact development process. We confirmed that the persuasive system design process provides many useful tools for BCSSs development, including the PSD model. In addition, using other IS frameworks and models would be beneficial. We perceived that our suggested software architecture for BCSSs was applied successfully in our system development. Further studies on the applicability of the architecture should still be conducted. We observed that using a ready-set front-end framework such as Bootstrap is useful, as long as it is customized to fit the needs of the project. Finally, we inspected the technical implementation, identifying the most important technical concerns in BCSS development.

The major implication of the research presented in this paper is that it provides a deeper view of the technological aspects involved in BCSS development. The phases in the development and the technical solutions chosen can be inspected to acquire insight of the actions needed to produce this kind of a system. This will help in getting out of the black-box thinking approach to the research and development of persuasive systems, enabling designers to recognize characteristics of software components that may lead to the success or failure of applications. For practitioners, utilizing the findings will help to create conventions that will ease and accelerate the development of new BCSSs. For example the software architecture presented could be applied to create similar systems.

One limitation of this study is that, since the developed system was in its early development phase, no perceived or actual persuasiveness has yet been measured. The test users and expert evaluators were mostly people with academic background and experience on using web services. Thus, the resulting system should be more carefully studied to indicate its impact. To further demonstrate the generalizability of our findings, multiple systems should be developed utilizing them. These should also include systems outside the currently used application domain area. For example, how the system architecture would adapt to native mobile applications or web applications for sustainable energy behaviors could be studied.

In the future, the developed system will be carefully evaluated with regards to various facets of persuasiveness, and its software functionalities will be studied with real users. Future work will also include further definition and verification of our suggested architectural model. We hope to be able to apply the suggested software architecture to multiple different problem domains and settings.
References


