



jamk.fi

Modern Business Analysis

Case Study for Koodain Solutions Oy

Tommila Aleks

Bachelor's thesis

April 2018

Technology, Communication and Transport
Degree Programme in Software Engineering

Author(s) Tommila Aleksii	Type of publication Bachelor's thesis	Date 29.04.2018 Language of publication: English
	Number of pages 59	Permission for web publication: x
Title of publication Modern Business Analysis Case Study for Koodain Solutions Oy		
Degree programme Software Engineering		
Supervisor(s) Lappalainen-Kajan Tarja, Huotari Jouni		
Assigned by Koodain Solutions Oy		
Abstract <p>Koodain Solutions assigned a technical business analysis for a software project that would perform as a service for communal skills and knowledge tracking. The objectives were to research the best practices of technical business analysis, elicit the requirements of the project and produce technical documentation that would allow developers to start production of the product solution to meet those requirements.</p> <p>In order to establish that the produced technical documentation was up to par with the developer requirements and that the right tools, methods and standards of business analysis were applied, a professional business analyst and a professional software developer were interviewed. The produced documents were presented to them, and their feedback was used to reach conclusions on the validity of the technical business analysis.</p> <p>The results are the product of an inductive, qualitative process based on a case study. The data gathering methods were source study of authors in the IT and business analysis fields, and professional interviews.</p> <p>The real requirements were elicited using client interviews, a SWOT analysis and the Problem Pyramid model. After deriving the requirements the technical documentation based on them was produced using the theory and best practices of technical business analysis and software development standards. The produced technical documents are attached.</p> <p>The conclusions drawn from the professional interviews were that the chosen tools, documents and standards portray the best practices of modern technical business analysis and meet the requirements from the perspective of software development.</p>		
Keywords/tags (subjects) Business Analysis, Koodain, good practices, IT, software project		
Miscellaneous (Confidential information)		

Tekijä(t) Tommila Aleksi	Julkaisun laji Opinnäytetyö, AMK	Päivämäärä 29.04.2018
	Sivumäärä 59	Julkaisun kieli Englanti
		Verkkojulkaisulupa myönnetty: x
Työn nimi Modern Business Analysis A Case Study For Koodain Solutions Oy		
Tutkinto-ohjelma Ohjelmistotekniikka		
Työn ohjaaja(t) Lappalainen-Kajan Tarja, Huotari Jouni		
Toimeksiantaja(t) Koodain Solutions Oy		
<p>Tiivistelmä</p> <p>Koodain Solutions toimeksiantoi teknisen business-analyysiin tekemiseen ohjelmistoprojektille, joka toimisi palveluna yhteisöön kuuluvien taitojen ja osaamisen etsimiseen ja kirjauttamiseen. Tavoitteena oli tutkia teknisen business-analyysin hyviä toimintatapoja, johtaa projektin vaatimukset and toteuttaa tekninen dokumentaatio, joka sallisi kehittäjien aloittaa palvelun tuottaminen vaatimusten täyttämiseksi.</p> <p>Jotta voitiin varmistua, että toteutettu tekninen dokumentaatio oli ohjelmistokehittäjien vaatimusten tasolla ja että käytetyksi tulivat oikeat business-analyysin työkalut, metodit ja standardit, ammattilaista business analyttikkaa ja ammattilaista ohjelmistokehittäjää haastateltiin. Heille esiteltiin tuotetut dokumentit ja heidän palautettaan käytettiin johtopäätösten vetämiseen siitä, oliko tehty business-analyysi validi ja täyttikö se vaatimukset.</p> <p>Tulokset syntyivät induktiivisen ja kvalitatiivisen prosessin tuloksena, joka suoritettiin tapausstudiona. Tiedonkeruu perustui alan ammattilaisten kirjoittamien artikkeleiden, blogien, kirjojen ja kirjoitelmien tutkimiseen ja referointiin ja ammattilaisten haastatteluihin.</p> <p>Projektin oikeat vaatimukset johdettiin haastatteleamalla asiakasta, tekemällä projektille SWOT-analyysi ja toteutus ja ongelma johon toteutus yrittää vastata ajettiin "Problem Pyramid"-menetelmän lävitse. Kun oikeat vaatimukset oli johdettu, toteutettiin niihin perustuva tekninen dokumentaatio käyttämällä teorian pohjalta johdettuja business-analyysin ja ohjelmistokehityksen keinoja, työkaluja ja standardeja. Tuotetut dokumentaatiot ovat liitteinä.</p> <p>Johtopäätökset perustuivat ammattilaisten lausuntoihin valituista metodeista ja toteutetusta dokumentaatiosta ja johtopäätös oli, että tuotettu dokumentaatio ja käytetyt metodit vastaavat modernin ohjelmistokehityksen tarpeita ja mahdollistavat vähähävikkisen ohjelmistokehityksen.</p>		
Avainsanat (asiasanat) Business-analyysi, hyvät käytänteet, it, ohjelmistoprojekti		
Muut tiedot		

Contents

Acronyms and terminology	4
1 Introduction	5
2 Koodain Solutions Oy.....	6
3 Scope	6
4 Research Methods and Model	7
5 Research Questions	7
6 Frame of Reference	8
7 Data Collection Methods	9
8 Business Analysis as a Field.....	9
9 Business Analysis in the Field of Information Technology	11
10 Knowhow required of BA.....	12
10.1 Position of a BA and skills that are often expected	12
10.2 Six areas of knowledge as defined by Johri.....	14
11 Challenges of a BA	15
11.1 Client time is precious and irregular	15
11.2 Information is not forthcoming.....	16
11.3 Avoiding accountability	17
11.4 Perceived needs over real needs	17
11.5 Changing needs	17
12 Tools of the trade	18
12.1 Some background to BA tools	18
12.2 The Problem Pyramid	18

12.3	Non-functional Requirements Analysis.....	20
12.4	The Requirements Workshop.....	21
12.5	The Activity Diagram	21
12.6	Flowchart.....	22
12.7	Database Diagram	23
12.8	Data Flow Diagram	23
12.9	Sequence Diagram.....	24
12.10	Business Domain Model	25
12.11	Use Case Diagram.....	26
12.12	User Stories	28
12.13	SWOT Analysis	28
12.14	System Architecture Diagram	30
13	Case: Koodain Solutions Oy.....	30
13.1	Pre-knowledge of the project	31
13.2	Interviewing the Client	31
14	Selection of Analysis Methods	32
15	Conducting the analysis	33
15.1	Applying the Problem Pyramid.....	34
15.2	SWOTting the project	35
15.3	Translating Needs into Features	36
15.4	Mind Mapping a Solution around the Features	37
15.5	Determining Entities and Their Relations	37
15.6	How Data is Stored and Referenced	38
15.7	Drafting the Dataflow within the System.....	39
15.8	The high-level components of the system	40

16	Interviewing a Professional Business Analyst	40
17	Presenting Analysis to professional SW Designer	42
18	Summary and Conclusions	43
19	Self-Evaluation and Critique.....	45
	References	46
	Appendices	49
	Appendix 1: Questions for the client interview	49
	Appendix 2: User Stories	50
	Appendix 3: Django Framework Sequence Diagram	51
	Appendix 4: Entity Relationship Diagram	52
	Appendix 5: Mind Map.....	53
	Appendix 6: Architecture-diagram.....	54
	Appendix 7: Database Diagram.....	55
	Appendix 8: Class Diagram.....	56
	Appendix 9: Use case diagram	57
	Appendix 10: Edit Entities Dataflow diagram	58
	Appendix 11: New Community Dataflow diagram	59

Figures

Figure 1: The Problem Pyramid	19
Figure 2 An example of a dataflow diagram with Gane & Sarson notation.	24
Figure 3: A business domain model in the style of a class diagram.....	25
Figure 4: An example use-case diagram depicted by Visual	27
Figure 5: SWOT-example	29

Acronyms and terminology

To fully understand this study, it is important to first understand the relevant terms used throughout the paper. Definitions are mostly related to different documents involved in business analysis and software development as well as general concepts and abbreviations of the information technology field.

IT – Information technology as a general field.

Business Analysis – The process of turning what the client wants or is willing to pay for into practical terms for developers and designers through various kind of documentation.

BA – Business Analyst, the person carrying out the business analysis. Interchangeable with System Analyst and Business System Analyst within the context of this paper.

SWOT – A chart and tool of business analysis. Abbreviation for “Strengths, Weaknesses, Opportunities, Threats”. The SWOT analysis was developed by Albert Humphrey for strategy planning in any project.

Flowchart – A diagram that is used to represent the workflow of a process in simple steps pictured as boxes connected to each other by arrows with one box leading to another.

DB – Short for database, a storage of data where it can be queried, edited, input and deleted from.

UML – Short for Unified Modeling Language, which is a general-purpose modeling language for the purpose of describing a system’s structure and flow. Under UML there are several different kinds of diagrams that describe different logical parts of the system and from different perspectives of operation.

1 Introduction

The study was conducted as a case study tied to a software development project issued by Koodain Solutions Oy. The existing theory and methodology of business analysis was first studied, and the requirements of the project were elicited using practices common in the field. With the requirements at hand, a professional software developer was consulted revealing what forms of specification would be needed were he/she to start developing the project. This is one of the most relevant issues that the study bored into: what is needed today for a software developer to understand and develop what the business case demands?

The specification documents were then drafted and presented to a professional business analyst to get their perspective on the relevance and validity of the tools and methods used. Having both the perspective of the software developer and the business analyst and their views on how the requirements should be translated for development specification, it can be guaranteed that the analysis that was carried out meets the needs for modern software development and as such, the needs for the project assigned by Koodain Solutions Oy.

Oftentimes the software development business is under tight deadlines every step of the way. It is common for projects to get bogged down on changes and additions no matter how agile the production cycle aims to be. Therefore, it is vital to understand the client's needs properly and make sure that he gets what he needs instead of what he wants. This way the need for changes due to misunderstandings can be minimized. Being able to do that as fast as possible also adds value. Changes and additions are still likely to appear and when they do, it is valuable to ensure that they can get into development as fast and as accurately as possible. The process of turning requirements into actionable user stories and specifications should be as streamlined as possible with everything excessive cut out, however, with everything required for developers included. Therefore, it is important to follow the evolution of the tools, methods and utilities available and used by business analysts everywhere, so the most efficient and the most relevant methods are in use. Using the generalizations produced by this study, the reader can get an idea of what the duties are and

what kind of tools and methods a business analyst needs when working in software development today.

2 Koodain Solutions Oy

Commissioner of this project was a software company based in Jyväskylä called Koodain Solutions Oy which was bought at the start of 2018 and is now owned by Etteplan Oyj. At the time of writing Koodain Solutions employs 13 people of who most are programmers most all of who are situated in various companies around Jyväskylä resourced in to software projects. Koodain was founded 2015 has seen significant growth ever since. Currently Koodain Solution's employed professionals employ their skills in android development and design as well as front – and backend development of web applications. Koodain Solutions is always looking for new exciting opportunities for business and improving the Jyväskylä and the thriving IT-community present there, which was what motivated the project that the business analysis done here uses as a basis.

3 Scope

As this study was done as a case study of a software project and aimed to generalize what is modern and relevant and what is not within the pool of methods and tools of a business analyst working on a software project, the scope of this project focused on technical analysis and how it has evolved into business analysis. The theory this study refers to were digital and literary sources relating to technical analysis, business analysis and software development between 1980s and 2018 as well as previous studies and analytical data available relating to the same.

Left outside the scope is most all theory of business analysis specific to other fields of business and development that does not directly relate to the field of information technology or software development. Within the field of IT itself, the study will not consider specifics of business analysis for integrated systems or hardware in general. The project assigned by Koodain Solutions Oy was software only, and no considerations for hardware were required.

4 Research Methods and Model

As Koodain Solutions Oy assigned this study for a specific software project, the chosen model of study is a case study. As a case study, the chosen methods based on the comparison of modern and dated theory on business analysis for IT could be applied to an actual project, and their relevance could be verified by a professional business analyst and a professional software developer within the context of that project.

All applied methods of analysis were specifically chosen to best serve the specification process of this single project and no broader applications of these methods were studied. No empirical or quantitative data was produced. This approach of study followed a qualitative approach where the perception of usefulness of the produced comparisons between different methods of business analysis and their appliance to the project are subjective to the researcher, the interviewed software developer and the interviewed business analyst.

The final summary of the research drew generalizations of modern business analysis based on the specifics of the process of analysis applied to the relating software project that can be seen as the result of an inductive process. As opposed, deductive process was not applicable as the study was not based on any initial theory.

5 Research Questions

The core research question this case study answers is “what business analytical tools and methods are needed in modern business analysis in order to produce a clear and concise technical specification to form the basis for development and to assure client satisfaction?” The answer to this question gives a perspective to the best practices of modern business analysis in the context of a modern software development project.

Before the actual study question could be answered, there were two prerequisite questions. The initial prerequisite question was asked of the client assigning the project: “What features, and functionality should be included? What are the requirements of the project from the client’s perspective?”

Having answered these questions, the company established the set of features and requirements from their perspective. This formed the first half of the specification

requirements that the business analysis aimed to answer. The second prerequisite question was asked from a professional software developer: “What kind of specification does a professional software developer require in order to be able to produce a concise feature and specifically, what questions does the specification need to answer?”

The answers to these two questions formed the full base of requirements for the project specific business analysis portion of this study. For the analysis to have been viable in the context of this project, it needed to cater for both the needs of the client and the needs of the developer. The specification needed to have all the features and requirements as requested by the client but specified in such detail that the developer has everything he needs to develop a concise solution.

Knowing the needs from both the ordering and the developing perspectives, the comparison of various methods, standards and tools of business analysis could be applied against the established requirement set, and the research question could be answered.

6 Frame of Reference

The “bible” of business analysis is the Business Analysis Body of Knowledge or BABOK for short, published by the International Institute of Business Analysis. BABOK does not set a standard nor does it describe a process, however, instead of these, it describes the shared knowledge pool of methods, tools and processes used by the BA community around the world. While this study refers to various online and literary sources, BABOK remains the core source of knowledge, usually referred to by the sources that are in turn referred to by this study. As stated, other sources beyond BABOK were also used. These are various publications on the software development and business analysis in general, spanning all the way from 1980s to 2018, also including earlier papers, statistics, analytics, studies and news on both.

Further information is leveraged through interviews of professionals in both fields. A professional software developer gave a perspective on what a developer requires from a BA in terms of specification. A professional business analyst, on the other

hand, validated the requirements elicitation process as well as the relevance of the specification.

7 Data Collection Methods

No statistical or empiric data was collected. As the study is qualitative in nature, the relevant input required to determine the methods of modern business analysis was achieved through interviews of professionals in the fields of business analysis and software development. Notes of the interviews are not attached and all references to the feedback gained from the interviews are paraphrased. Beyond the interviews, the bulk of the theory and applied methods were found out researching BABOK and other such available online-sources.

8 Business Analysis as a Field

As a field, business analysis is relatively new. It came into existence during the Information Technology boom in 1980s – 1990s when the industry saw tremendous growth. This growth introduced new challenges and requirements for the development processes which needed increasingly more analysis and specification. This requirement was brought on the field of business analysis with the role of business analyst who largely took over all the responsibilities of a system analyst and expanded on them. (Johri 2010, 32-33)

As defined by the International Institute of Business Analysis IIBA, business analysis is “the practice of enabling change in an organizational context, by defining needs and recommending solutions that deliver value to stakeholders” (IIBA 2017). IIBA's definition of business analysis is generally too narrow to reflect the reality of the profession in modern corporate world, which shows especially in the field of IT, mainly because the service and product models of companies are so varied, which reflects very different needs for business analysis.

Brandenburg (2009) goes into responsibilities and related titles in her article about the differences between a systems analyst and a business analyst. She states: “The line between business analyst role and systems analyst role is not a clear one and I’ve

found that the business systems analyst role sits right in between. As a profession, we might choose to distinguish the roles, giving the business analyst domain over the business and functional requirements and the systems analyst domain over the details of the system implementation.” The distinction, however, is often not made and the titles and job descriptions are used interchangeably within various companies. (Brandenburg, 2009.)

If one distinguishes between the two analyst titles, business analyst is more business-related and operates more on the organizational level thinking along the lines of “what” to make into business and the system analyst is much more focused on the technical plans of the actual solution design or the “how” to produce the “what”. They do, however, share the same status of having the role in between the stakeholder or the client and the solution developers, according to Brandenburg (2017).

IIBA’s definition of business analysis is, however, very open for interpretation, and studying business analysts and their responsibilities in various work environments quickly shows how much difference can be found in their daily work. Just by searching for a job description or a definition for business analysts quickly shows a great discord between various sources. A business analyst can be portrayed wholly as a business professional with no technical knowledge working with organizational changes in accordance with the organizations overall strategy; however, a business analyst can be just as quickly defined to be a highly technical person working closely with system and solution architects, sometimes with testers, and sometimes they can even be found in team leadership positions. On the other hand, a systems analyst is sometimes referred to as a business analyst with technical knowledge and then just as easily he/she is an IT support type of person with intimate knowledge of the internal information systems of the company and his/her job is to identify and improve upon the issues found in those systems. In practice, a business analyst is an ambiguous profession.

Some generalisations of the profession can, however, be made with relative confidence regardless of the variance in duties. A business analyst’s job from a high-level perspective is to identify and meet the business needs and opportunities that could produce value for the company. The business need or opportunity is translated into actionable requirements that when filled, actualize the value via a solution. A BA

stands between that business requirement or opportunity and those who can actualize that value through implementation.

9 Business Analysis in the Field of Information Technology

In the field of Information Technology, the core concept of business analysis remains largely the same. The business case is often represented by a client presenting needs or requirements. The solution is developed by programmers and other such technical professionals. There is, however, a gap between the two. Clients cannot express their needs technically enough for the developers to produce them concisely, nor can the technical developers elicit the requirements from the client clearly enough. The job of the business analyst is to bridge the gap and provide a translation that turns the wants and needs of the client into user stories with standardized data models, graphs and diagrams that are a language shared by developers across the world, enabling the development process to take place even in international contexts.

As stated in the previous chapter, a distinction can be made between a business analyst and a systems analyst, and they can even be combined and called a systems business analyst. In the IT field, they are largely the same and simply called business analysts, fully or partly incorporating the job description of a systems analyst.

There are systems analysts today, however, the title was more prevalent in the 1980s and 1990s before business analysis emerged. Their job description also suffered heavily from ambiguousness; however, according to a study performed by Graf and Misic, the system analysts of 1990s worked most heavily in communicative developer, analyst and technician roles. It is evident their duties were very similar to those of business analysts working in IT today. (Graf & Misic 1994.)

For simplicity, this paper will from here on out not distinguish between a business analyst, systems analyst and business systems analyst. They will simply all be referred to as business analyst. Incorporated in the definition will be both business and technical know-how.

10 Knowhow required of BA

10.1 Position of a BA and skills that are often expected

A business analyst works with stakeholders in various positions with various backgrounds and very different knowledge areas. This requires a business analyst to be able to approach different stakeholders with the language and mind set of that specific stakeholder. This relates to communication skills, which is overall the most important core skill required of a BA. Brandenburg (2009) in her article “What Business Analyst Skills are Important for a New BA?” opens the requirement for communication skills: “Business analysts must be good communicators. This means they can facilitate working meetings, ask good questions, listen to the answers (really listen), and absorb what’s being said. In today’s world, communication does not always happen face-to-face. The ability to be a strong communicator in a virtual setting (via conference calls or web meetings) is equally important.”

A BA must not forgo problem-solving nor critical thinking either. A project is prone to encountering problems and in a sense, the whole project can be seen as one large problem requiring a solution. Problem solving skills are not critical to a BA in the sense that he or she would be expected to solve the problem but because it is vital to understand the problem in order to communicate it to all stakeholders, and a shared understanding of the problem can be established. Critical thinking, on the other hand, is essential for eliciting the real problems instead the symptoms of the real problem and the real needs of the client. (Brandenburg, 2017.)

Facilitation and elicitation skills are an integral part of the BA profession. A BA needs to be in contact with different stakeholders continually throughout the development process, eliciting the initial requirements as well as managing any changing needs that occur down the line. The changing requirements, all concerns and emerging issues affecting the planned output of the development need to be mutually understood and acknowledged by the stakeholders. To accomplish this, a business analyst facilitates various meetings known as requirements workshops, validation sessions, requirements reviews, elicitation sessions etc.

To translate the needs and requirements, the problem and the solution being developed need to be analyzed carefully. This relates to analysis skills that for a BA relate to three key levels, i.e. the business-level, the software-level and the information-level. The first is concerned with business processes and work-flows on organizational level, meaning how value can be produced for the business. An analysis on this level produces business-requirements. The software-level analysis is about how the software system supports the business-level requirements. An analysis on this level produces use-cases and user-stories. Finally, the information-level analysis produces data-models and diagrams on data-flow and storage, which in turn supports the software-level.

The translation of the requirements into developer-friendly formats after elicitation and analysis requires visual modeling skills. Visual modeling is carried out using various tools and software that can be used to create diagrams, charts, tables and flows. There are various standards, one of the most famous currently is UML or the unified modeling language. However, the style and type of the solution and the requirements of its development determine what tools and modeling standards should be used.

Technical skills are essential for a BA working in the IT field. A BA is not expected to be a programmer, however, understanding of components, APIs, libraries, frameworks and such aid in understanding the concerns of the developers as well as help with resource planning and modeling the solution.

Knowledge of methodologies is also highly advisable, however, this goes for everyone working in the IT field. The most used methodologies are Scrum, Kanban, Lean, RUP and FDD. Chances are that as a BA an employee will be expected to be part of the development methodology cycle. Even if that is not true, it is imperative to understand what the development cycle model is and how it works.

Domain expertise is valuable to a BA no matter the field. Understanding of the workings of the related domains the project underway involves is key to understanding how the solution can result in value for the company. Working in the IT field, a BA

needs to understand the technology domain and the related business domain. Having knowledge of both, a BA can analyze how the IT solution should be made to meet the business domain's need in order to produce value.

10.2 Six areas of knowledge as defined by Johri

Because business analysis suffers from so much ambiguity, in order to further establish the knowledge areas of a BA beyond those outlined by Brandenburg, Amit Johri in his book *Business Analysis* condenses business analysis into six areas of knowledge: enterprise analysis, requirements planning and management, requirement elicitation, requirement analysis and documentation, requirements communication and finally solution assessment and validation. (Johri 2010, 106-107)

Enterprise analysis focuses on understanding the needs of the organization from the business perspective. For a business, there is a business need that requires solutions that meet that need. A business analyst carrying out enterprise analysis has the responsibility to identify those needs and the solutions. A successful resolution of a business need translates to income or an organizational change that puts the company in a better position to make turnover. A misjudged need or a badly executed or planned solution to a need damages the company in terms of turnover or worsens the organization's market status. (Masters, 2012.)

Requirements planning and management determine how requirements will be elicited, analyzed and documented. In practice, a business analyst conducting requirements planning and management ensures that the team roles are filled with appropriate personnel, efficient methods of information gathering are utilized; the team has a unified understanding of the scope, the goals and the action points of the project; the team effectively monitors problems and challenges and reacts to them accordingly. Finally, he/she ensures that the team has all necessary tools and resources and the requirements activities are coordinated with the team. (Universal Class)

Requirements elicitation is the process of finding out the requirements of a project, process or a system. Gabry (2016) outlines on his blog on requirements elicitation and analysis that requirements elicitation is part of a circular structure of four main

steps of requirements elicitation and analysis. The first step is requirements discovery where the initial requirements are roughly outlined. The second step is Requirements classification and organization where related requirements are put together, and the system being designed is decomposed into subcomponents with logical relations to one another. Finally, the relations between these subcomponents are defined. The third step comprises requirements prioritization and negotiation. During this step, the priority order of the requirements is negotiated with the stakeholders. These three steps are relevant to requirement elicitation. The fourth and final step is for requirements specification and in Gabry's model it is not part of elicitation but requirement analysis. (Gabry, 2016.)

After eliciting, categorizing, prioritizing and mapping their relations, the requirements need to be properly analyzed and documented, forming the specification that the designers and developers use for their specific processes. This includes everything and anything from user stories to various diagrams ranging from database and system flow to service and tool stack diagrams. Requirement analysis and documentation defines and describes a solution that answers the identified requirements. (Johri 2010, 192-194) Requirements communication is the step where the analyzed requirements and the documented solution are presented to the client or stakeholders and the implementation team of the project. Further value is achieved if they can be presented to users and all fields of the project implementation, i.e. design, development, information security and usability professionals. (Ibid.)

The last area of knowledge in Johri's six-area model is assessment and validation of the designed solution. The goal is to ensure that the proposed solution meets the objectives of the requirements and that it is thoroughly tested and can transition into development smoothly. (Ibid.)

11 Challenges of a BA

11.1 Client time is precious and irregular

Having defined that a BA stands in between the project owner or the ordering party and the solution developer, one has to ask if the job of a BA simply is not to ask the

commissioning party what they want to pay for and tell the developing party the same. Simply put, the answer is yes. The challenge is to eliminate the layers of ambiguity from between a worded “want” in a verbal language that is highly open for interpretation and the very specific development layer where features are defined by very specific parameters and use cases. This one challenge can be littered with various typical hindrances or sub-challenges. Listed will be the most typical challenges for IT projects from the perspective of a BA.

It is very common for the client representatives to have a busy schedule and too much trust in their money buying them a clairvoyant development team. This means a BA has little precious time with the clients to discover the requirements and conduct the proper analysis. Not only the scarceness of the client’s time but also its irregularity can cause issues. BA work is often takes place in workshops and meetings attended by several key people, product owners, developers, possibly designers and specialists from both the producing and ordering sides. However, the key personnel might attend irregularly, forcing the BA to proceed with incomplete information or even worse, unsure or false information. When the key personnel later attend again, they might revise the process and revoke the previous work, wasting time and resources due to their lack of attendance. (Parameswaran, 2011.)

11.2 Information is not forthcoming

There might be reasons why a client might be unwilling or unable to share information such as overly enthusiastic information security or simple ignorance. Their assigned client representative might simply be the wrong person for the job, pulled from a role or context of information inefficient for requirements elicitation. An overflow of information is also counterproductive for determining the requirements. Defining simple features or requirements from a stack of 2,500 pages of various levels of technical and business documentation is often very detrimental for the requirements elicitation process. (Ibid.)

11.3 Avoiding accountability

There are times when the process or project or business needs changes and those changes need executive decisions. All parties might agree upon the changes, however, no one is willing or is unable to take responsibility for them, and the changes cannot be executed. It might also be that the personnel present are only representatives that do not speak from a position of authority and might have to return with an executive decision on a perceived later date. (Ibid.)

11.4 Perceived needs over real needs

As described by Goldsmith (2004, 31) in his book *Discovering Real Business Requirements for Software Project Success*, requirements are discovered and not simply picked up from the client. This comes with various issues of which perceived needs versus real needs is the most common. When the client approaches a developer firm with a project, they have an idea of what they need. This described need is a perceived solution to a perceived problem. The issue here often is that the client has misjudged the actual problem and thus the perceived solution does not meet the real problem but often instead a symptom of the real problem. It is also possible that the client might have analyzed his problem correctly; however, instead he/she has not perceived the proper solution. It might be lacking, faulty, dated, misaligned or simply misinformed. If possible, the BA should study and recognize the real problem and then suggest a real solution that answers that problem. This way the client gets what he needs, instead of what he thinks he needs. (Parameswaran, 2011.)

11.5 Changing needs

Changing needs were mentioned previously; however, it deserves to be defined a challenge of its own. Often, a project undergoes changes. Something might not perform the way it was expected to, or the underlying technology or framework changes. This leads to the need to change the specification or feature set of the project. This is very frustrating for a BA because it might render a very large portion of previous work useless depending on how drastic changes are required and how far

into the development cycle the project is already. Some changes might nullify months of design and development. (Parameswaran, 2011.)

12 Tools of the trade

12.1 Some background to BA tools

Challenges met by BAs are shared by all variations of the profession regardless of the project types they work with. The requirement to solve these challenges has given rise to various methodologies and tools that BAs can use to better identify the real needs and specify concise solutions that meet said needs. These tools and methodologies are not specific to the field of business analysis. The tools and specification standards are commonly shared by designers, developers, testers, BAs and business people alike, since they are often standards and models common to the whole IT field. A business analyst needs to be able to identify, chart, describe and visually model the business benefit of a project and then be able to translate the requirements into actionable development tasks. Therefore, a BA often requires both business and IT specification knowledge and needs to be familiar with the tools for both fields.

The tools and methodologies that a BA needs do still vary greatly, dependent on the actual job description of the BA. Hence, it would be impractical and fruitless to try and explain all of them. As this paper focuses on the business analysis of the IT field, the unrelated tools and methods have been left out. Listed are some of the most common ones that have been used in relation to software projects starting from year 1990.

12.2 The Problem Pyramid

The Problem Pyramid is a tool for finding out the real requirements for any project. People often focus too much on presumed processes and presumed solutions. They misinterpret the problem. Producing a solution to a problem that is not confirmed to be the real problem often leads to the situation where the final solution is delivered

but the underlying real problem remains unresolved (Goldsmith 2004, 80). Figure 3 illustrates the Problem Pyramid.

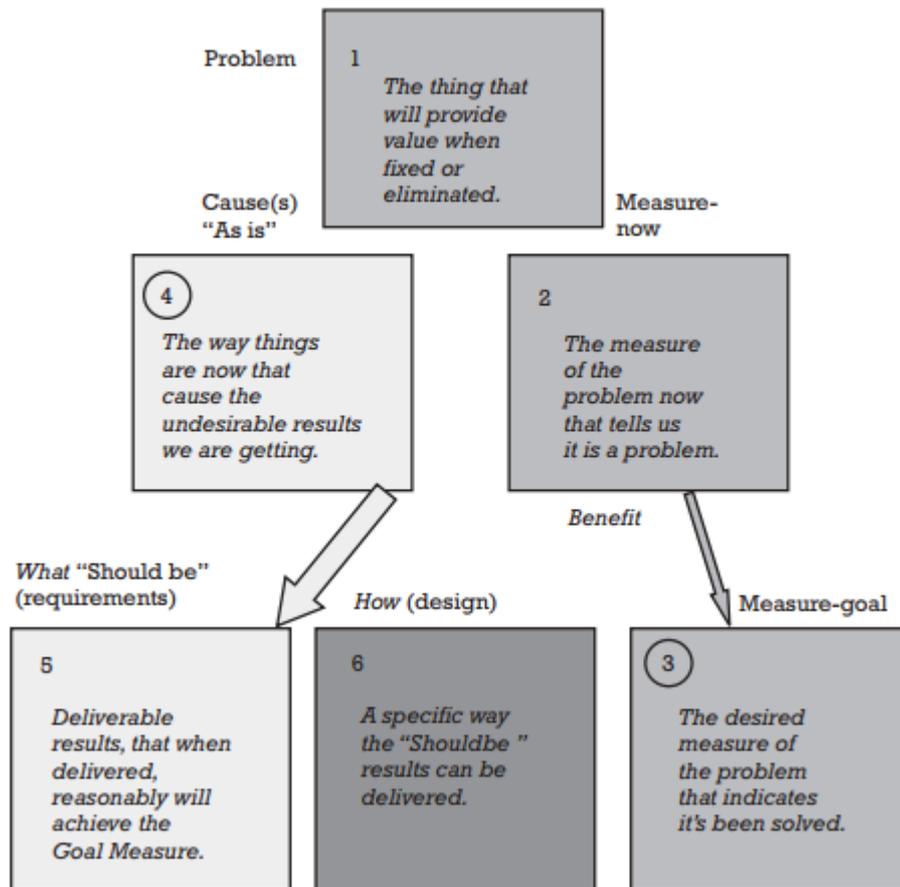


Figure 1: The Problem Pyramid (Goldsmith 2004, 81)

Following the Problem Pyramid shown in Figure 3, the initial set of requirements need to be formatted in such a way that it becomes apparent whether the requirements in their initial state are an actual fix to an actual problem or not.

Step one describes the problem as seen by the client. This is the starting point, the premise that the pyramid will take apart to see if it is the actual problem or not.

Step two of the Problem Pyramid forces the analyst to put the problem into measurable terms. If the problem cannot be measured, it is likely not defined well enough. Such measures are often e.g. time, money, quantity, amount of personnel.

Step three determines when the problem is fixed by defining a "victory condition"-measure. This is an increase or decrease in the same measure as defined in step two. For example, if the problem has too long a transition time from assembly to delivery,

the measure would be time and the estimation describing the current situation could be e.g. 12 hours. The goal measure described in step three could then be six hours, and when that is achieved the problem is successfully solved.

Step four forces the analyst to consider the cause of the problem beyond the problem itself by defining the current process. An example of this could be defining the problem as “slow delivery times” and then describing the current process partly as “vehicles breaking down and being down for maintenance during delivery”. By describing the current process, it turns out that the slow delivery times are actually a result of the problem and not the problem itself and the actual problem is outdated hardware.

Step five describes this process as it should be in order for the problem to be fixed. Obviously, in order to fix the process in step four, the ideal process describes the delivery vehicles being down for maintenance less and running deliveries uninterrupted.

Step 6 describes the solution to achieve the process described in step five and through that process the goal measure of step 3. In terms of a software project, this is often a high-level system description.

12.3 Non-functional Requirements Analysis

Esta Lessing (2014) describes non-functional requirements as the black sheep of requirements elicitation. This is due to them being of the type of requirements that need to be met, yet, are not necessarily ever seen or directly experienced as an end user. She also adds that non-functional requirements are the requirements describing the underlying qualities of a system rather than what specific functions one expects the system needs to be able to perform. (Lessing 2014.)

Non-functional requirements are dividable into five main categories, namely performance, reliability, security, support-ability and usability. Performance is the measure of speed and scalability of the system as well as how well any given function is executed. Usually there are thresholds set how fast certain operations should load through, how many users can a system service at one time or whether the result of an operation is what is expected. (Ibid.)

Reliability is a measure of operating efficiency under various conditions examples of which could be e.g. time, stress, throughput, quality level, and availability. Security is the measure of safety from unintended access to the internal configurations and data of stored in the system or subsystems. Security measures also include any and all communications between external services and third parties. Support-ability means the measure of difficulty for upgrades or changes to the system after the system is released for use. Usability can be measured in terms of how fast a new user can learn to use the system, how intuitive it is and how conveniently the user can perform various actions with the system. Usability takes into consideration color-blindness, deafness, blindness and other such disabilities. (Ibid.)

12.4 The Requirements Workshop

The requirements workshop is a meeting of stakeholders that takes place prior to any design or development. It is the meeting where the requirements of the project are elicited, and all parties of the production chain can and should present their input and find common ground on what should be done and how. There can be any number of these workshops and there should be at least as many as it takes to elicit the full set of initial requirements and then more as changes are required during the development process.

The requirements workshop method originates from Joint Application Design (JAD) method developed by IBM. It is a workshop where different stakeholders sit down and share their perspectives and find common ground on how to proceed. JAD itself is not involved with requirements alone; however, they are one of the discussion areas of JAD. (Yatco, 1999.)

12.5 The Activity Diagram

The activity diagram is a simple way to visually describe the paths of any workflow process with its sequential and parallel activities and decisions. It is suitable for both business application modelling and software system modelling. It is a type of a flowchart and there are various activity diagram models; however, the most common are the UML 1.x and 2x standards. (Richardson, 2006.)

While a flowchart displays a basic workflow with its activities and decisions, an activity diagram adds the possibility to describe parallel activities with forks and joins. Use of swim lanes is also common. When using UML 2.x standards, the activity diagram incorporates many more complex functionalities, such as node and token-based flows, pin-notation, structured nodes and activity partitions. (Ericsson, 2004.)

12.6 Flowchart

Flowchart is a genre for types of graphical or symbolic diagrams that describe the flow a system, process, operation or business. Some of the most common types of flowcharts are swim lane, data flow, influence workflow and process flow diagrams. Flowcharts show the process in steps called actions and decisions. A process is a series of actions that lead to other actions or in case of loops return to an earlier action or even the same action through decisions. (Hebb.)

Decisions depict choice on a flowchart. A choice is between two or more options that all lead to different actions. An example action could be “Add water to the bucket” that leads to decision “Is the bucket full” with the choices “yes” and “no”. No loops back to the action “Add water to the bucket” while yes continues down to a different path to a different action. (Ibid.)

Relevant components of a usual flowchart are terminators indicating the start and end of a process. Action is a logical, independent step in the process chain. Decisions illustrate choice and guide the flow to various directions. Connectors are the lines between other components, representing which component leads to which. There are various types of flowcharts for various different purposes and depending on the field a flowchart is used, there are different kinds of additional components e.g. data components representing input or output of data in the process and various results produced by the process between steps such as documents or profiles, objects, and accounts. (Ibid.)

12.7 Database Diagram

Database diagram describes how data is stored within a database. It does not describe interactions to and from a database or database service providers, attachments to databases or application interfaces; instead, a database diagram describes data entities within a singular database in the form of tables, relations, data rows, data columns and data fields. In a relational database, relations describe how the data “relates” to other data. An example of this is would be the relation between a car salesman and a car. A car salesman has a record of every car he has sold. Similarly, the car has a record of its salesman during its history. A car can be sold many times and a salesman can have sold many cars. This is a relation of many to many. Between them there should be a database entity called a sales transaction. A sales transaction has a single seller for a single car. This is a relation of one to one. This entity is a record of every purchase of a car from a salesman. Querying sales transactions, one can find, for example, which salesman has sold which cars, or who has sold a specific car. (What is a Database Diagram, 2018.)

12.8 Data Flow Diagram

Dataflow diagrams were first described in *Structured Design* written by Larry Constantine and Ed Yourdon in 1979, and they have seen wide use since. However, it has never been adopted into the UML standard. (SmartDraw.)

BABOK defines the data flow diagram’s purpose as to show how information is input, processed, stored and output from a system. In general, it provides a visual representation of how information is moved through a system. It also depicts external entities that provide or receive data from the system, the processes transforming the data, the data stores where the data is collected and the flows by which data moves between entities, processes and data stores. (Brennan, 2009.)

The most commonly encountered notations are the Yourdon & Coad notation and the Gane & Sarson notation. Their differences are at the first glance mainly graphical, but the Yourdon & Coad – notation is more often seen used with system analysis and design while their competitor is more commonly used for visualizing information systems. Figure 4 (below) shows all the elements of a dataflow diagram interacting

with each other, lines representing data flow between external entities, data stores and processes. (SmartDraw.)

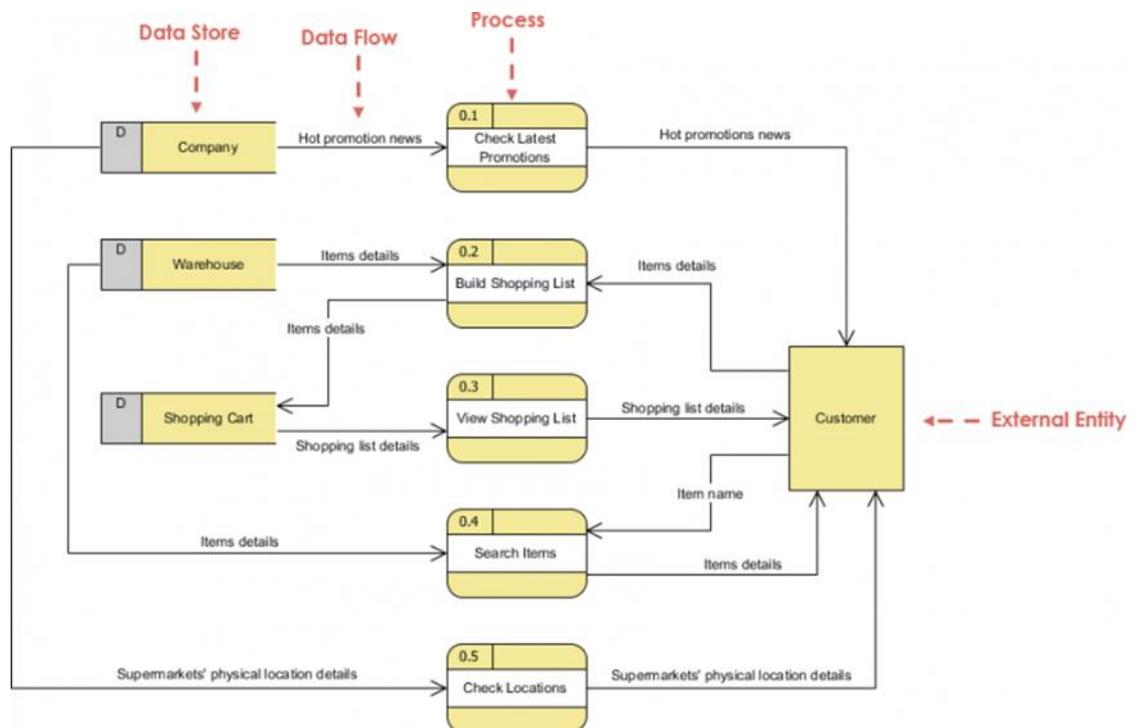


Figure 2 An example of a dataflow diagram with Gane & Sarson notation (Poole).

12.9 Sequence Diagram

Sequence diagram describes the sequence of interactions or “calls” between logical entities described as columns. An action is always done by a logical entity, and when that action requires interaction with another logical entity, it is represented by a horizontal line from that entity to the other entity with a descriptive name for that action. The interactions are sequential as a function of time. The entities as columns also represent the lifetime of that entity. When an entity no longer holds any data in memory nor is pending any actions, the life time of that entity ends. The purpose of this diagram is to show what actions and calls are done in what order and how long memory should be reserved for different entities. Sequence diagram portrays the flow of the system as actions through time. (Athuraliya 2018)

12.10 Business Domain Model

The business domain model describes a “domain” of related data and behavior centered on a solution or an issue. The purpose of a domain model is to visualize the patterns and interactions of a system on a high level. Key points are what kind of entities interact and how with what other entities. The symbology is very close to the class diagram apart from UML; however, the class diagram does not show behavior but dependencies between logical objects (Maxted 2008). Class diagrams are often used to create domain models with business vocabulary added to the relations as shown in Figure 5 (Pace University).

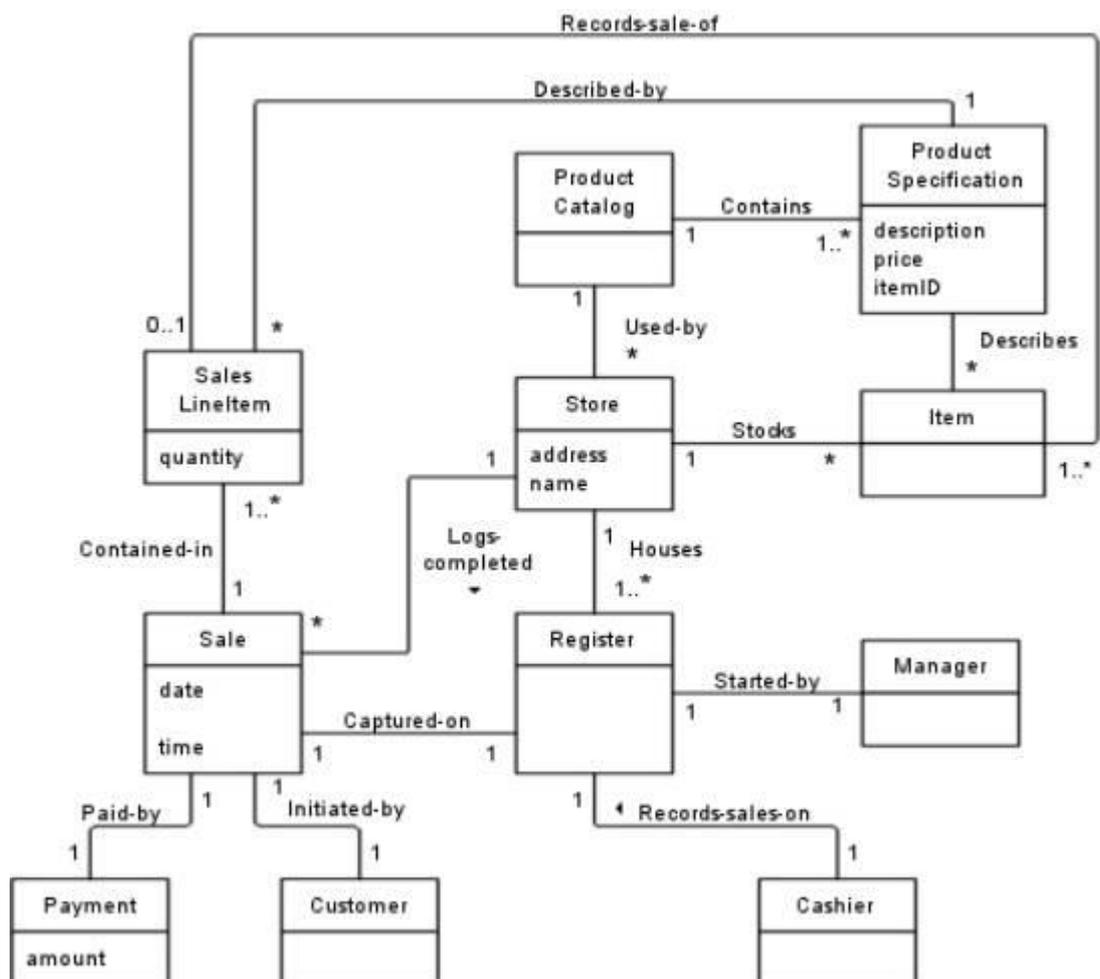


Figure 3: A business domain model in the style of a class diagram (Pace University).

The benefit of using a business domain model is to bring the discussion between the project members and stakeholders to focus on the same issues with better accuracy than what is generally achieved verbally. Concepts are simplified by reducing them to

simple graphical entities with linear relations to other entities and typing out their behaviors in a word or two. This way a possibly very complex system in real life can be simplified down into its base components. Business domain model is best used as a tool of to aid discussion and further planning. (Maxted 2008.)

12.11 Use Case Diagram

Originally formulated by Ivar Jacobson in 1986 and further popularized by his co-authored book *Object-Oriented Software Engineering – A Use Case Driven Approach* in 1992, use cases have since seen massive adoption and are an integral part of the UML set of design tools today.

The use case diagram is a simplistic diagram depicting the linkage between different users and the actions doable by said users. A single product or service can have different kind of users. Especially for web application projects this is commonplace. For a web store, there could be a basic user, a premium user, a content manager and an administrator user titles. These use the same product; however, have access to different features. For instance, a basic user may only see a product catalog and can browse it, place items into a shopping cart and check out. A premium user can do the same, however, he might have different content available to him at a discount. A content manager, on the other hand, is a staff-role and can edit, remove and add content on the product catalog. An administrator user typically can do all the things the other roles can do; however, with added high-level access to site settings. (What is Use Case Diagram, 2018.)

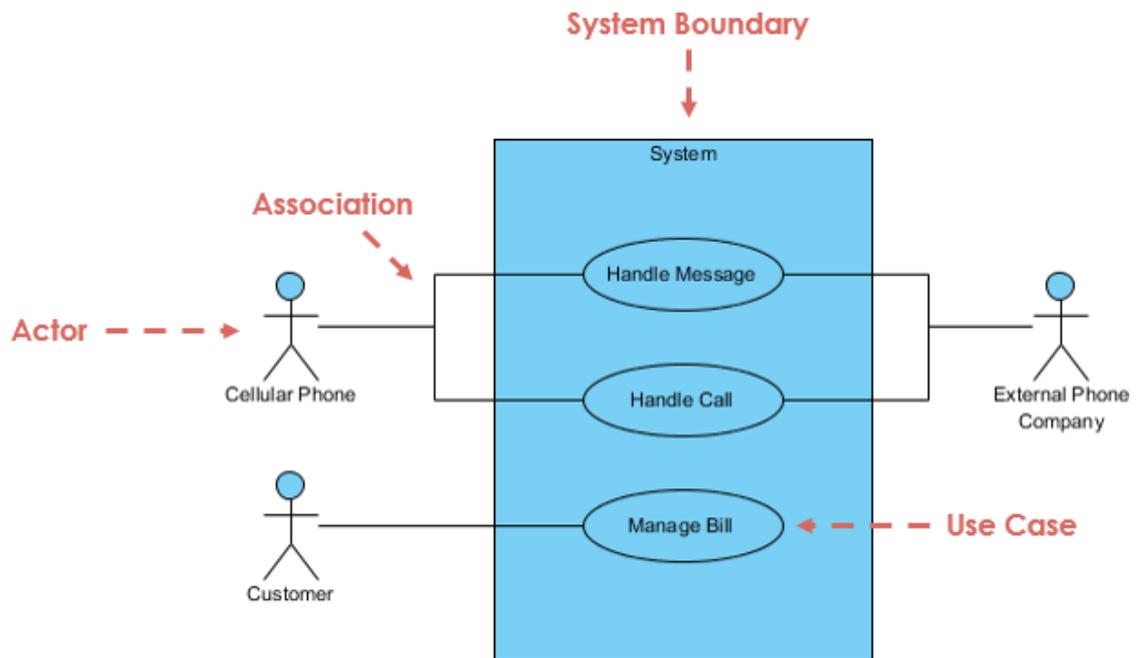


Figure 4: An example use-case diagram depicted by Visual Paradigm (What is use case diagram, 2018)

Different users of a system are called actors and they are depicted usually as a stick-men to represent a person, as shown in Figure 6. The actors are given names to represent their roles. The actual use cases represent actions the users can perform with the system and they are depicted as bubbles containing a few-word description of the action. Between the actor and the use case bubbles are association lines. These depict which actor can perform which use cases. Uses cases can also be associated with each other with one extending or including the other. (What is Use Case Diagram, 2018.)

Ambler (2004, 56) describes in his book *The Object Primer* the characteristics of include and extend associations. “An extend dependency ... is a generalization relationship where an extending use case continues the behavior of a base use case”. This means one use case can branch off another use case and can only ever happen going through the use case that is being extended.

An include dependency, on the other hand, Ambler (2004) describes to be “... a generalization relationship denoting the inclusion of the behavior described by another use-case.” Use cases that are logically different but share the same base functionality may all want to include the same use case. For instance, a student profile contains

plenty of information that is used by various use cases such as “manage contact information”, “manage grades”, “manage course enrollment”. These are all different use cases; however, they all partially incorporate a shared use case of “search and update information”.

12.12 User Stories

“User stories are short, simple descriptions of a feature told from the perspective of the person who desires the new capability, usually a user or customer of the system” (Cohn, 2018). These user stories are often related to software development projects or services. They are a way to simplify the requirements of a project into a simple who, what and why.

The common template for user story is: “As <type of user>, I want <some goal> so that <some reason>” (Cohn, 2018).

When this template is applied to an example software project, it could be: “As a customer, I want to be able to edit my account information so that it can be kept up to date”. This user story then describes a feature that allows a customer to have access to his/her account information and the possibility to edit that information, which can be justified with the value of being able keep one’s information up to date.

As the requirements of a project are elicited from stakeholders usually by a business analyst, it is likely that a business analyst translates these requirements into user stories that then are relatively simple to translate into functionality by the designers and developers. User stories are usually written into tasks that fill the backlog in Agile projects, especially this is the case with Scrum.

12.13 SWOT Analysis

S.W.O.T or SWOT stands for Strengths, Weaknesses, Opportunities and Threats. It is a simple and quick way to go through and list opportunities and threats to a business, a process or a project. (Berry.) In the following figure 7 Berry has laid out an example of a SWOT analysis table for a medium-sized computer store operating in the United States.

Strengths	Weaknesses
<ul style="list-style-type: none"> ● Knowledge: Our competitors are pushing boxes. But we know systems, networks, programming, and data management. ● Relationship selling: We get to know our customers, one by one. ● History: We've been in our town forever. We have the loyalty of customers and vendors. 	<ul style="list-style-type: none"> ● Price and volume: The major stores pushing boxes can afford to sell for less. ● Brand power: We can't match the competitor's full-page advertising in the Sunday paper. We don't have the national brand name.
Opportunities	Threats
<ul style="list-style-type: none"> ● Training: The major stores don't provide training, but as systems become more complex, training is in greater demand. ● Service: As our target market needs more service, our competitors are less likely than ever to provide it. 	<ul style="list-style-type: none"> ● The larger price-oriented store: When they advertise low prices in the newspaper, our customers think we are not giving them good value. ● The computer as appliance. Volume buying of computers as products in boxes. People think they need our services less.

Figure 5: SWOT-example (Berry)

The example above illustrates the simplicity of the analysis. While simple in terms of analysis tools, it forces one to consider not only the strength and opportunity that are usually the motivators of a project but also the weak points and threats. (Ibid.)

SWOT is best done in the brainstorming phase of any new project or a new major change in the existing plan. It is also best done within a group of invested people instead of doing it alone. Different perspectives provide diversity that in turn may provide value in recognizing the key points in terms of SWOT. An example of a diversely rich "swotting" team could consist of representatives from management, sales, customer service, design, planning, testing and development. (Ibid.)

Figure 7 shows the simple four-square layout of the typical SWOT table where the first square represents strengths. (Ibid.)

Strengths – the positives that support the success of the project; anything that can be considered as beneficial. It can often be something pre-existing such as hardware, locale, reputation, or brand.

Weaknesses – the vulnerabilities that can hinder, slow or bring down a project.

Weaknesses are unsurprisingly anything detrimental to the project. Examples of

such could be security vulnerabilities, old equipment or lack of training. These are areas that would need to be fixed or improved upon for the project or business to be successful.

Opportunities – the points that the project or business can and should capitalize. Examples of such could be expected growth of a certain market, new upcoming technology or any perceived reason why this project or business would prosper.

Threats – the external factors that are detrimental to the success of the project or business. These are separate from weaknesses, as typically these are something beyond businesses or project groups' control. Examples of these could be a weak market or a depression, a flaw in a used technology or a natural disaster.

12.14 System Architecture Diagram

The system architecture diagram describes the high level logical components of the system, often describing e.g. services, servers, technologies, software frameworks, packages, and APIs. It can also contain just hardware or software components. The purpose of the architecture diagram is to represent what components interact with each other. These components are not the same as the entities in an ERD (Entity Relationship Diagram), the entities in a DB diagram or the entities in a domain model. The server or service hosting the database that the database diagram describes, can be present in an architecture diagram. Similarly, the server hosting a web service that is in turn described by a class diagram, could be another component in the architecture diagram connected to the database component. (Balosin, 2017.)

13 Case: Koodain Solutions Oy

Koodain Solutions Oy is a software development company operating in Jyväskylä. As a programmer at Koodain Solutions Oy, the author of this paper was given the opportunity to analyze a project as a potential business case for Koodain Solutions Oy, and this case study presents the analysis and specification of that project. It needs to be

highlighted that the subject of this paper is not whether the project is or is not a viable business case for Koodain Solutions Oy; instead the company uses the project as a case study to determine the best modern practices in analyzing and conducting the specification for a software project.

13.1 Pre-knowledge of the project

Before interviewing the project commissioner for specific requirements, the following was brainstormed for the project. At this point of the study, the actual requirements were not clear, and an actual requirements elicitation interview had not taken place. The following describes the starting premise of the project.

The overall idea was to create a web application that users can browse to see the active companies or individual users in their community and their relevant know-how at a glance. It is beneficial for each member of a community to be able to effortlessly see what other know-how and fields of professionalism are present and available in said community. The program should also offer a streamlined solution for managing and booking the facilities available for the community.

The application should include an Android application (for mobile devices) that performs as a user ID for the community and any of its relative communities. With this application, a member of a community should be able to ID himself to other members of a related community.

A single-sign-on service was a requirement, i.e. a user can authenticate at any front of the application and be authenticated at all parts of the application. This would eliminate the need for separate authentications for different entrances to the service.

13.2 Interviewing the Client

The assignor for the project study was Mr. Jukka Laurinmaa. The interview with him took place on 21 December 2017 and was performed informally; however, prepared questions were used to give a structure for eliciting the requirements.

Laurinmaa presented a prepared outline of a solution. A brief description of the final solution is a central database for member data for the businesses present in a community. The required major functionalities would be provided by third parties utilizing the central member database. Such third-party services would include member authorization, member information and account management, charting and displaying community know-how and resources, access control, facility booking, calendar functionality, automated notifications to local third-party services (e.g. keys, credentials) when new members are added and contract uploading and delivery to billing.

The initially discussed set of features desired would include the following:

- Single-sign-on
- Authorization levels (Admin, Business Owner, employee)
- User management (view, create, modify, delete)
- Facility booking
- Automatic notifications sent for keys and credentials
- ID-badge mobile application

14 Selection of Analysis Methods

Judging from the writing of various authors in the field of business analysis there are various start points for requirements elicitation. For instance, it can start from analyzing the requirements of the suggested solution often initially presented by the commissioning party. This means finding out what is required for this specific solution to work. However, a more thorough approach would be to analyze the actual problem the initial solution suggestion (if provided) aims to correct and analyze the problem to verify if it indeed is the real problem and not a symptom of the real problem. This precise problem is solved by the Problem Pyramid.

After finding out the requirements, a draft of the solution should be made. It depends on the company how much of this draft is made by a BA. Sometimes it might not be necessary for a BA to make a draft at all if there are available architect resources available. Sometimes a BA is expected to make the draft down to the minute details of entity interaction and data flow. For completeness and for the lack of architects or otherwise better qualified personnel present for this project, technical level diagrams were produced, including a class diagram, database diagram, sequence diagrams, data flow diagram, an architecture diagram and a use case diagram. Non-

technical documents such as a mind-map, an activity diagram, user stories and a SWOT analysis were also produced.

Mind map is for describing how components and features are related to each other on a high level. This is a non-standard, very high-level description of the solution. A class diagram is relevant for any software project as it describes the entities or objects of the system and their properties, methods and relations to other entities. A sequence diagram describes the interaction sequentially and the lifetime of these entities as a function of time. The database diagram describes the data structure of database entities and generally what kind of data is stored and how it relates to other data. The data flow diagram describes how the data moves through storages and processes within the system. Use case diagrams describe the actions different users can perform with the produced system and user stories write these open in the form of who, what and why. These were all relevant for any software project that processes and stores data and has a user interface as such they could be used to describe the Koodain Solutions Oy case.

15 Conducting the analysis

In the first interview, Laurinmaa presented a set of features for a software solution. Going by the methodology described by Goldsmith (2004) in his book *Discovering Real Business Requirements for Software Project Success*, in order to produce more reliable value, the software and the required features should not be taken at face value as the starting point for the project but instead should be analyzed to find out what the problem is that the software aims to respond to. There is one specific tool identified earlier, called the Problem Pyramid that is specifically suitable for this purpose and finding the real needs. In this exact case, finding the real requirements is a moot point mostly, because the requested features are locked in. Even if the real needs revealed by the Problem Pyramid turn out to be something different than what the features resolve when implemented, no changes to the features could be made. However, for the sake of completeness and curiosity, the Problem Pyramid was applied.

15.1 Applying the Problem Pyramid

Applying the theory presented by Goldsmith (2004), the real requirements can be elicited by applying the Problem Pyramid. In order to apply the pyramid, what needs to be assessed first is if the presented requirements can be placed into the pyramid. The first step of the pyramid needs a definition for the problem to be solved. The first issue is that Laurinmaa has presented a solution, not a problem. More in-depth questions presented to Laurinmaa reveal that that the issue has no centralized solution for seeing and managing the members or active parties of a community or a geographical area and the knowledge and skill they project. This translates into “too hard to find relevant skills and professionals close by or within my community”. (Laurinmaa, 2018.)

The second step of the pyramid determines the measures of the problem currently. If the problem cannot be measured, it is not defined clearly enough. Since the issue is the difficulty of finding relevant skill in an area or a community, it could be measured in either time or number of different web-domains that need to be visited to gather a sense of the availability of a certain field of knowledge. Time would be difficult to average for this purpose. On the other hand, it is quite straight forward to set out to find five local companies that utilize a certain technology. Finding this information currently would take visits to each company’s proprietary web-site. Ideally, this information would be available at one location.

The third step is the desired measure of the problem that when reached indicates the problem has been solved. Centralized location for finding relevant skills and know-how would mean a single location for this information. This means the measure to achieve instead of five different domains for the information, it could be found in a single domain.

Step four determines the current state of things that cause the current undesirable measurements. The current state results from a lack of available services that would display local and communal skills-sets available. Every firm has their own website and their own products and services on display, sometimes relevant technologies are described as well, but there is no a place to centrally look for professionals of a certain skill or technology or knowledge-area across companies.

Step five describes what the deliverables are from a business point of view that will change the process so that the measure-goals of step 3 are achieved. The deliverable is search efficiency when looking for relevant know-how in a community or area. This efficiency is measurable in time or the number of domains required to visit to gather the desired information. With a service enabling search efficiency through centralized data, the goal measures are achieved.

The sixth step describes an initial high-level technical solution that produces the deliverables described in the previous step. If step five is the “what”, then step six is the “how”. As such the deliverable result should be a web-application that centrally catalogs and displays communities, companies, organizations and all such groups, and the know-how they contain with a “clear at a glance” design. The web application would allow for entities to register with the service by themselves, allowing them to input their own information, categories of business and skill-areas. Then any user could perform searches on the service and find graphical and textually summarized results of skills available in different communities and companies.

15.2 SWOTting the project

Performing a SWOT analysis for an idea is informally called swotting. The purpose of swotting is to find out the strengths, weaknesses, opportunities and threats to the idea under analysis. Having used the Problem Pyramid previously to validate the problem and outline a rough solution, SWOT is a good early next step for enforcing the solution idea and potentially finding out weaknesses previously overlooked.

A major strength of the project is its uniqueness as a service. As this project was commissioned out of a need for a service catering for those looking for specific know-how, it is unsurprising that this should also be its greatest strength. There is no competition. Another strength is the simplicity and availability of the technology required to develop this kind of a service as it does not require development or implementation of any kind of new technology. Instead, widely used web technologies can be deployed in development.

Brainstorming for the next step, it was found out that the weaknesses of the project are primarily with the level of adoption of the system. A service intending to perform

as a search platform for communities and skills is vulnerable to not having enough registered communities and users, on which the quality of service is heavily dependent on. Also, if the displayed skills are admin input and not user input, this will cause a weakness where a user creating his/her profile, or a community might not be able to select the skills he/she is looking for because the admin might have overlooked or simply was not aware of such skills. On the contrary, if the service is designed to allow the users to input their own skills completely freely, this creates a weakness where there might be several different keywords for the same skill that differ only slightly. Searching for communities with a certain search word for a skill might not show an accurate list of results as a portion of eligible communities are omitted due to having the same skill written differently. Finally, one major weakness is the monetizing model for the service. It cannot be hidden behind a paywall; however, running the service costs money and increasingly more of it as the system needs to scale. Adverts on the page are a possibility; however, they are unreliable and low-key income that also diminish the user experience. Premium features might be an option, however, the income provided from that requires a large user base. Remaining weaknesses are usual considerations of scalability and information security.

Opportunities include monetization and lack of competition. Lack of competition is an opportunity to adopt plenty of traffic for the service. Monetization was listed as a weakness prior; however, given enough user traffic and a well enough thought of model, it can be a considerable source of revenue and as such can be considered an opportunity.

Threats include the appearance of competition, abuse of the service by creating phony communities and adding to them phony skill areas and phony users. Also, if users are able to create skills freely, they might be inappropriate or simply spam. As per usual, the greatest threat to the service is the intended user of the service.

15.3 Translating Needs into Features

Requirements are translated into features by first describing users and the actions they can perform in a use case diagram. Then they are verbally described in user stories, which then describe the role of the actor, the action and the value produced by

that action. Users stories were written for authenticating, creating communities, adding skills to communities and to users, adding a community under another community, searching communities, searching skills in an area, removing communities, removing users from communities, removing skills from communities, removing skills from users, removing users, editing community information and editing user information. Full user stories can be seen in Appendix 2 and the use case diagram in Appendix 9.

15.4 Mind Mapping a Solution around the Features

Mind mapping is often used during or soon after the first requirements workshop. The intention is to write down and link together concepts required for the solution. This includes features, technologies, hardware concerns, safety issues and general notes. It is a simple way to graphically steer discussion and visually orient people to what has been discussed. It should not be used to describe the status of a system or contain too much detailed data. Detailed information such as methods and variables should always be in documents holding more relevance to said minute data, i.e. variables and methods in e.g. a class diagram, database structures in a database diagram. The mind mapping for this solution in Appendix 5 shows roughly what the system should include and what it should be capable of doing and roughly in one order these actions should happen. It also shows ideas for authorization levels. A community information manager should be able to conduct different actions than a normal user.

15.5 Determining Entities and Their Relations

The entity relationship diagram describes physical or logical entities and their relations to one another. The entities for this solution were identified as users, communities and skills and their junction entities as can be seen in Appendix 4. Between a user and a skill was a professional, meaning when a user has a skill, he/she is a professional with that skill. The term “professional” was not of course fully accurate in all cases but served here just as an entity name. Also, between a user and a community was a “member” entity which was a logical junction of a user being a member of a community.

Junction entities were there to work around the many-to-many relationships of their connected entities. Many-to-many relations are not allowed in SQL; however, such a situation arises when a user can be a member of any number of communities, and a community can have from one to any number of users as members. Hence, it is fixed by a logical junction entity “member”. A member is a user who is the member of a single community. This way, the user can be attached to various communities as a member. Literally speaking, a member entity can only have one person and one community attached to it, allowing users to have as many memberships as they want and communities to have as many members as they want. The same applies to the user-entity and skill-entity. A user can have any number of skills and a skill can have any number of people who know that skill. Avoiding the many-to-many relationship between the two a junction entity “professional” was created.

A user entity was simply a user of the system. A person would hold such attributes as user id, first name, last name and email. Of these, the user id would be the unique attribute. A community entity would be various firms, organizations, clubs, companies and the like with e.g. a community id, name, address, description, admin authorization, operator authorization, parent community. Of these, the community id would be the unique attribute. The skill entity describes a skill with such attributes as skill id, name and description. Of these, the skill id was the unique attribute. The junction tables are unique by the combination of both keys from both connecting tables, i.e. a member is unique by his/her user id combined with the community id. As long as a user cannot become a member of a community twice, uniqueness is assured.

15.6 How Data is Stored and Referenced

Data was to be stored in a MySQL database. The database structure that was drafted was based on the entity relationship diagram shown in Appendix 4. The resulting database diagram in Appendix 7 shows with more in-depth detail the attributes of the database entities.

15.7 Drafting the Dataflow within the System

Dataflow diagram describes the flow of different sets of data through the different components of the system. The diagrams were produced per action performable by the user. When a user performs a search by typing in a community or a location and optionally a keyword for a skill or a technology, the search words are transmitted from the UI to the business logic layer behind that parses the search words into a database query. The query is then sent to the database engine. More specifically, the “skills” and “communities” tables are queried. The response from the database is handled and parsed into more suitable data structures and then transmitted to the UI layer that in turn displays the search results for the user on the interface.

While a search can be performed by any unauthenticated user, adding a new community requires authentication. The registration process is separate and the dataflow for community assumes already existing credentials. The process begins when the user clicks to create a new community. If the user is not logged in, data is sent to the login service that communicates with the user database to authenticate the credentials. The login service responds with success or failure. If the login was successful, the session storage is updated with approved authentication status. The authenticated user is then able to start the community creation process. The user fills in the community information for the new community and submits them. The new community information is moved to the process responsible for parsing a database query. The new information is sent to the database via the query. The database returns a response that is parsed into a more readable format and returned to the UI for the user. The diagram for this flow can be seen in appendix 10.

For removing a community or skills and users attached to community, the dataflow is the same as previously; however, with the additional step of checking whether the user is either an administrator, community owner or community manager for that community. This is obvious to disallow unrelated or mischief motivated people from doing damage to communities owned by others.

Editing these entities requires to have been logged in and the appropriate access to the entity under editing. It should not be possible for users to edit items that they are

not authorized to edit such as communities owned by other people or the profiles of other users. The dataflow diagram for this can be seen in appendix 11.

15.8 The high-level components of the system

As seen in appendix 6, the system was designed to leverage the functionalities of Django framework. It contains a restful-API, a database and a customer management system and a polling application. These are Django specific and their further specification can be accessed on the Django documentation page hosted on their domain. Outside of Django, an attached third-party service called Keycloak is leveraged to handle single-sign-on services, allowing for expansion of the system with new and different features and even portals if need be; however, to stay under one single login system.

16 Interviewing a Professional Business Analyst

In order to validate the chosen the methods and tools for the analysis of the case project, a professional of the field was interviewed. Wishing to stay anonymous, the name, company and the literal commentary were omitted from the paper. References to her commentary are paraphrased. Having worked over five years in the field of IT business analysis, it could be established that she was well versed and experienced in the ways of working of a BA and as such suitable to validate or invalidate and otherwise comment on the chosen methods of this analysis.

Questions presented to the BA were purposed to first find out what kind of methods she would have employed had she been responsible for the BA work of the Koodain Solutions Oy project. There were some differences mainly due to different resourcing within her company. There were solution architects and a product owner present. The architects did most of the systems planning, and the product owner was the first contact with the client. The BA work for her required being in contact with clients and the product owner both in order to establish the requirements. The requirements were then translated into documents describing features such as user epics and user stories. The solution architect would then meet with the product owner and the BAs to discuss the requirements from the perspective of the system stack.

This way of working is somewhat different from the way of working needed for this case project as there was no product owner and no solution architect available, meaning the BA work had to be much more involved in technological specification. Regardless of the differences in the way-of-working and project resourcing, the BA was sufficiently able to provide a perspective and validation for the chosen methods in the Koodain Solution Oy's project by putting herself in the role of a business analyst for this project.

Going into the documents and diagrams, the BA had concerns about the clarity of the chosen technologies. Mainly the class and sequence diagrams are valuable as documents for the project design; however, the use of external frameworks such as Django that was chosen for this project confuses the documents somewhat because the framework is large and provides a great deal of functionality used in the project. The border between what is introduced with the framework and what needs to be built in-house is difficult to put into design planning and would at the least require a Django knowledgeable person for consultation. Furthermore, the BA wanted to establish that the sequence and architecture diagrams should only be created if the BA him- or herself is familiar and up to date with related systems architecture but these diagrams are nevertheless a valuable part of design planning.

Going forwards, the BA stated that all requirements elicitation should always begin in a requirements elicitation workshop for the stakeholders; however, it is not wrong to have a one on one discussion either prior to bringing other stakeholders into it. This should always include all requirement areas, not only functional requirements but also non-functional ones.

SWOT analysis is a valuable tool in the brainstorming phase where solution ideas are thrown around and while it is not vital or strictly speaking necessary, it is a valuable tool to catch problems of a solution idea early. The Problem Pyramid was a new concept to her and as such, she had never used it herself; however, he imagined that it could prove useful in the same brainstorming phase as SWOT and in situations where the client is not "hell-bent" on an idea for a solution of their own.

User epics and user stories are a staple of the IT world and according to the interviewee, going without them would severely hinder the translation of requirements

into development tasks. Prior to writing these, a use case diagram is another staple but often for smaller systems. A large system can have hundreds of actions for the same user and presenting them all regardless of them being on a single document or multiple documents defeats the purpose of easily attainable information to graphical simplicity. For a project as small as the case here, it was deemed a usable option while it might not have been made exactly to standard.

Different flowcharts are good overall for all stakeholders of the project to visualize large and difficult systems and features. The business domain model, mind map, data flow, activity diagram and the like are there to describe the same system from different perspectives. Their effectivity and as such the need to have them in planning is hard to gauge, but at the least they are powerful tools to be used and they should be used if there is any confusion how a certain part of the solution works. At the bare minimum, having them might eliminate misunderstandings and prevent unnecessary resource overhead.

Summarizing the results of the interview, valuable perspective was gained. None of the chosen tools or methods were found unnecessary or worthless to the project. On the contrary, the chosen models provide a good basis for discussion for starting development. Much of the solution, its purpose, features, architecture and flow is documented and while it is not in every case perfectly up to standard or always clear enough to be self-explanatory, it should provide much of what a systems developer would need in order to start production and in the cases it would not suffice, further elaboration should not prove difficult to reach understanding.

17 Presenting Analysis to professional SW Designer

(asiantuntijahaastattelu, kehittäjän palaute toteutetuista kaavioista ja user storyistä, näkeekö kehittäjä puutteita tai ongelmia, onko analyysi kattava ja voidaanko sillä vastata asiakkaan tarpeisiin)

In order to gauge the value of the produced documents and diagrams, an interview with a professional software developer was needed. Atro Lähdemäki was interviewed, a professional from Koodain Solutions Oy with solid background in software development. Lähdemäki was a particularly prime choice for this interview as he was

working towards the actual technical implementation of this very same project. As such there would have been no one better to gauge the adequacy of the planning documents.

The interview was impromptu as no questions had been prepared. Without a set of questions to structure the commentary from Lähdemäki (LÄHDEVIITE?), the commentary itself was omitted from the paper and is instead paraphrased here within. The interview style was an open dialog.

Starting off Lähdemäki outlined that at the bare minimum the non-functional-requirements, the class diagram, the database diagram and the user stories would be needed. They contain the most relevant information for a software developer. From a class diagram, Lähdemäki stated, he would see the entities in the system, what kind of data they should contain and what kind of interactions they should have. The database diagram would be mandatory in order to understand how and what data the system needs to store. User stories detail the features that the system should be able to do, which usually pretty much describes the functional requirements. The non-functional requirements are usually secondary to the functional ones; however, it is important to know details such as scalability requirements and accessibility concerns from the get go. With these documents, Lähdemäki states it is possible to start development and have it go in the right direction from the start. Additional documents such as the architecture diagram reinforce the design and minimize misunderstandings and while they are not always strictly mandatory, they are always good to have according to Lähdemäki.

Summarizing the input from the interview, it can be said that for the most part the analysis has produced the core requirements of what a developer needs in order to start development. It is possible that strictly from a software developer perspective some unnecessary documents were produced: unnecessary in the sense that they are not strictly required while they provide additional information about the system.

18 Summary and Conclusions

The conducted analysis produced a class diagram, a database diagram, a dataflow diagram, an activity diagram, a use case diagram, an architecture diagram, sequence

diagrams, a SWOT analysis and user stories. These documents are based on research of the business analysis field and way of working as well as the common practices of modern software analysis. The purpose of these documents was to analyse a problem presented by Koodain Solutions Oy to find out the requirements to a solution that would address this problem and finally, to draft the solution to meet those requirements. The produced documents should be relevant and precise enough to allow for a software developer to begin the development of the solution based on them. The relevance and preciseness of these documents were tested with professionals, both a software developer and a business analyst. The conclusions drawn here are based on their input on the produced results.

The perspective and feedback offered by these professionals validated much of the produced documents. The BA found that while some of the documents were unfamiliar to her, they seemed to produce value at a glance and she would not discredit their relevance to the analysis. Others, more commonly used in the IT field she felt were almost standard to the way of working in IT projects and as such more of an expectation than a choice, validating their relevance here with the Koodain Solutions Oy project. The rest of the documents were not so common, however, allowed that they definitely brought further detail and clarity to the project. The only critique she offered was to do with the complexity of trying to entity model a third-party framework, confusing the documentation and that some of the documentation had parts that were improvised and not strictly to standard.

Lähdemäki, a professional software developer also confirmed much of the produced documents as relevant and from his perspective, functional in their intended purpose. His commentary was more involved in the technical documentation, disregarding such products of the analysis as the Problem Pyramid, mind mapping and SWOT analysis as those do not pertain to his field of work. Lähdemäki did not offer much in the way of critique but noted that some of the technical documentation is more optional than mandatory while also allowing that they provide value in a more case-by-case way.

Concluding from these feedback summaries, it can be said that the analysis successfully produced a set of specification documents that form a clear and concise basis for development. The chosen tools and methods were found to be relevant from

both the perspectives of a software developer and a professional business analyst. Also based on Lähdemäki's feedback, the produced set of documents among others contains the documents that can be considered mandatory for IT projects generally. This beyond everything else confirms that these documents are needed in modern IT projects, regardless whether they are the result of BA work or not. What cannot be properly established based on the feedback is whether the specification or analysis lacks some other diagrams or documents that could have produced significant value for the project.

19 Self-Evaluation and Critique

The research question was successfully answered although the result is somewhat subjective. It would be safe to say that the conclusions produced here are generalizations of technical business analysis that could be applied to any software project but not in the way that every single document and diagram and method produced and applied to the Koodain Solutions Oy project could or should be applied to any software project imaginable, hence why the results should only be considered generalizations, not as a rule.

The chosen methods of analysis were based on available sources on technical business analysis and software development which means the chosen methods were selected based only on what was available and what was read, leaving out unquantifiable amount of theory. Selection of the tools and methods might have different if the sources for theory had been different.

References

- Ambler, S. 2004. *The Object Primer*. E-Book. Accessed on 26 April 2018. Retrieved from <http://library.books24x7.com.ezproxy.jamk.fi:2048/toc.aspx?bookid=8942>.
- Athuraliya, A. 2018. *Sequence Diagram Tutorial: Complete Guide with Examples*. Blog article. Accessed on 10 January 2018. Retrieved from <https://creately.com/blog/diagrams/sequence-diagram-tutorial/>.
- Balosin, I. 2017. *The Art of Crafting Architectural Diagrams*. Article. Accessed on 10 April. Retrieved from <https://www.infoq.com/articles/crafting-architectural-diagrams>.
- Berry, T. *What Is A SWOT Analysis?* Article. Accessed on 29 April 2018. Retrieved from <https://articles.bplans.com/how-to-perform-swot-analysis/>.
- Brandenburg, L. 2009. *What is the Difference Between a Business Analyst and a Systems Analyst?* Article. Accessed on 17 December 2017. Retrieved from <http://www.bridging-the-gap.com/difference-between-a-business-analyst-and-systems-analyst/>.
- Brennan, K. 2009. *A Guide to the Business Analysis Body of Knowledge*. E-book. Chapter 9 – 9.6. Accessed on 6 February 2018. Retrieved from <http://library.books24x7.com.ezproxy.jamk.fi:2048/toc.aspx?bookid=42496>.
- Business Analysis: Planning and Management Requirements*. Page on Universal Class' website. Article. Accessed on 15 January 2018. Retrieved from <https://www.universalclass.com/articles/business/project-management/business-analysis-planning-and-management-requirements.htm>.
- Cohn, M. 2018. *User Stories*. Page on Mountain Goat Software's website. Accessed on 29 April 2018. Retrieved from <https://www.mountaingoatsoftware.com/agile/user-stories>.
- Data Flow Diagram*. Page on SmartDraw LLC's website. Accessed on 4 February 2018. Retrieved from <https://www.smartdraw.com/data-flow-diagram/>.
- Domain Model: Visualizing Concepts*. Page on Pace University's website. Accessed on 7 February 2018. Retrieved from http://csis.pace.edu/~marchese/CS616/Lec5/se_15a.htm.
- Ericsson, M. 2004. *Activity diagrams: What they are and how to use them*. Article. Accessed on 26 January 2018. Retrieved from <https://www.ibm.com/developerworks/rational/library/2802.html>.
- Gabry, O. 2016. *Requirements Engineering – Elicitation & Analysis, Part 2*. Blog article. Accessed on 17 January 2018. Retrieved from <https://medium.com/omarelgabrys-blog/requirements-engineering-elicitation-analysis-part-2-a02db801f135>.

- Goldsmith, R. 2004. *Discovering Real Business Requirements for Software Project Success*. E-book. Accessed on 26 April 2018. Retrieved from <https://ebookcentral-proquest-com.ezproxy.jamk.fi:2443/lib/jypoly-ebooks/reader.action?docID=227685&query=>.
- Graf, D. & Mistic, M. 1994. *The Changing Roles of a Systems Analyst*. Journal. Accessed on 18 December 2017. Retrieved from <http://www.irma-international.org/viewtitle/50992/>.
- Hebb, N. *What Is a Flow Chart?* Article. Accessed on 2 February 2018. Retrieved from <http://www.breezetreecom.com/articles/what-is-a-flow-chart.htm>.
- Jacobson, I. 1992. *Object-Oriented Software Engineering – A Use Case Driven Approach*. Book. 1. Ed. Addison-Wesley.
- Johri, A. 2010. *Business Analysis*. E-book. 32-33, 106-107, 192-194, 220-221, 338, 368-369. p. Accessed on 7 January 2018. Retrieved from <https://ebookcentral-proquest-com.ezproxy.jamk.fi:2443/lib/jypoly-ebooks/reader.action?docID=588078&query>.
- Lessing, E. 2014. *Non Functional Requirements – The Black Sheep in the Requirements Family!* Article. Accessed on 25 January 2018. Retrieved from <http://business-analysis-excellence.com/non-functional-requirements/>.
- Lewis D., Russman H., Stanton M., Weber J.H. 2014. *Chapter 2 Planning and Designing your Database*. E-book. 12-16. p. Accessed on 7 January 2018. Retrieved from https://wiki.documentfoundation.org/images/7/7b/BG4202-PlanningDesigningYourDatabase_DEL_JHW_20140116.odt.
- Masters, M. 2012. *An Overview of Enterprise Analysis*. Article. Accessed on 7 January 2018. <http://www.modernanalyst.com/Resources/Articles/tabid/115/ID/1567/An-Overview-of-Enterprise-Analysis.aspx>.
- Maxted, S. 2008. *The Importance Of Business Domain Modelling*. Article. Accessed on 9 February 2018. Retrieved from <https://www.batimes.com/articles/the-importance-of-business-domain-modelling.html>.
- Parameswaran, A. 2011. *Six Common Problems Faced By A Business Analyst*. Article. Accessed on 22 January 2018. Retrieved from <https://www.batimes.com/articles/six-common-problems-faced-by-a-business-analyst.html>.
- Poole, R. *Supermarket App*. Diagram. Accessed on 6 February 2018. Retrieved from <https://circle.visual-paradigm.com/supermarket-app/>.
- Richardson, M. 2006. *Differences Between UML 1.x and UML 2.0*. Accessed on 26 January 2018. Retrieved from http://www.michael-richardson.com/processes/rup_for_sqa/core.base_rup/guidances/supportingmaterials/differences_between_uml_1_x_and_uml_2_0_CA70F2E6.html#activitydiagram.
- What is Business Analysis?* Page on IIBA International Institute of Business Analysis' website. 2017. Accessed on 16 December 2017. Retrieved from <http://www.iiba.org/Careers/What-is-Business-Analysis.aspx>.

What is Entity Relationship Diagram. 2018. Page on Visual Paradigm website. Accessed on 27 April 2018. Retrieved from <https://www.visual-paradigm.com/guide/data-modeling/what-is-entity-relationship-diagram/>.

What is Use Case Diagram. 2018. Page on Visual Paradigm website. Accessed on 26 April 2018. Retrieved from <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/>.

Yatco, M. 1999. *Joint Application Design/Development*. Analysis. Accessed on 29 April 2018. Retrieved from <http://www.umsl.edu/~sauterv/analysis/JAD.html>.

Appendices

Appendix 1: Questions for the client interview

Question 1: From the perspective of a user of this app, what should the user be able to do?

Question 2: Should it be possible to do everything from a phone, a computer or both?

Question 3: If functionality should be spread between devices, which device should handle what functionality?

Question 4: Will people need to authenticate on a personal level, or company level or should both be possible?

Question 5: If both should be possible, should their functionality set differ?

Question 6: Is a 3rd party single-sign-on service a requirement even if the program doesn't have multiple points of authentication?

Question 7: Is skillhive a requirement or are there options? We'll most likely have to order the skillhive presentation package, will you as a client want to participate?

Question 8: The mobile application should work as an ID. Should it work as an ID on personal level, company level or CrazyTown level? What information should be shown? Are there any functional requirements?

Question 9: Depending on how skill-hive works, is there a requirement for an external website or web application beyond that of the mobile identification app? Skillhive might accept single-sign-on and show this data on their own domain. Do we need to export their data and present it on our own domain?

Question 10: Any other requirements, accessibility, responsiveness, scalability, exportability, security, budgeting?

Appendix 2: User Stories

As an individual user I want to browse through any of the companies in community so that I find a company with specific skills and knowledge areas.

As an individual user I want to search for a specific community with a specific search keyword so that I can look for relevant skills.

As an individual user I want to browse through communities' employees so that I can find individual professionals.

As a individual I want to create a new community for my company so that I can show my company's capabilities to possible clients.

As an employee I want to apply to my company's community so that my profile is shown on the community's page.

As my company's community manager, I want to invite my employees to my company's community so that

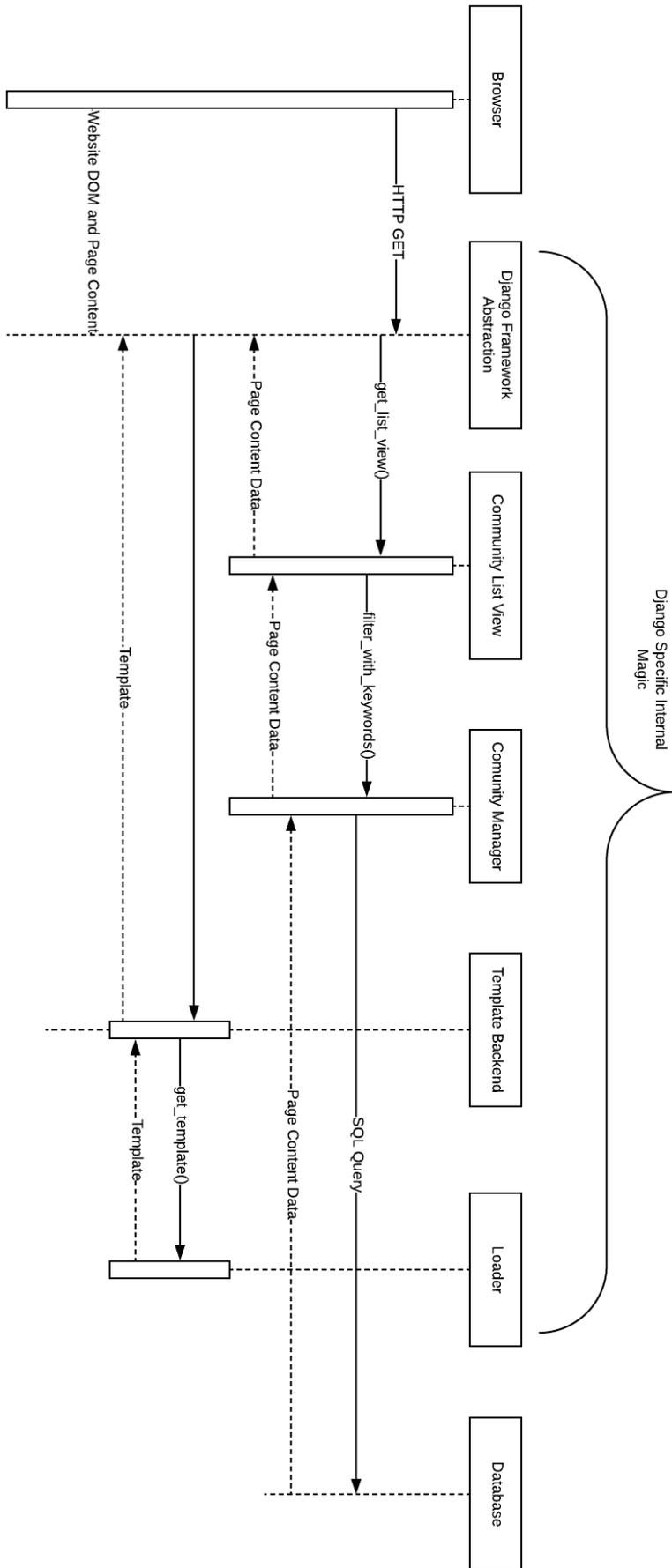
the employees' skills are browsable and represented in the community.

As a community admin I want to deactivate a specific community so that it isn't publicly visible.

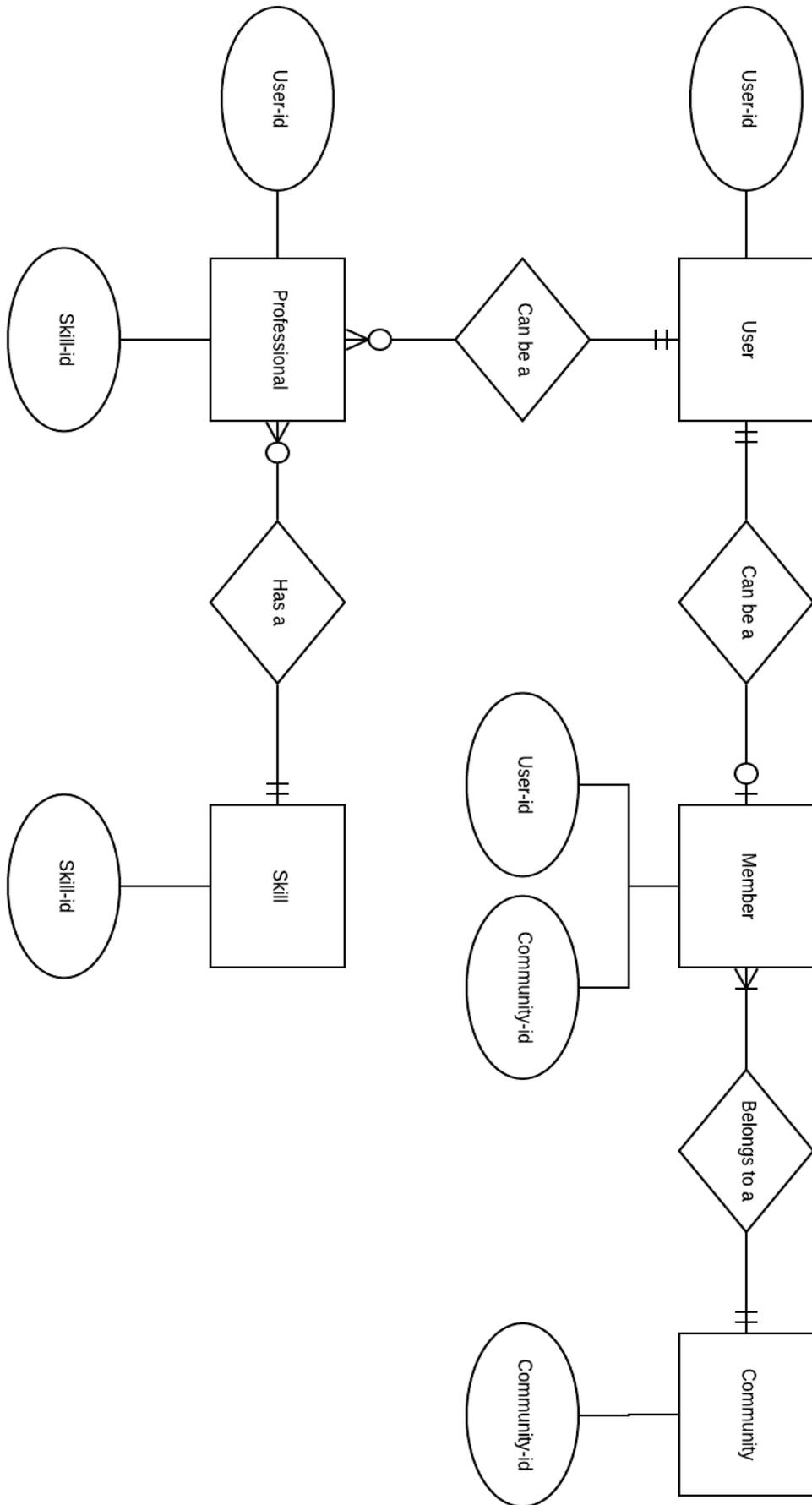
As an individual I want to edit my personal information which is shown on my public profile.

As an individual I want to login through an SSO service so that my login data is securely handled.

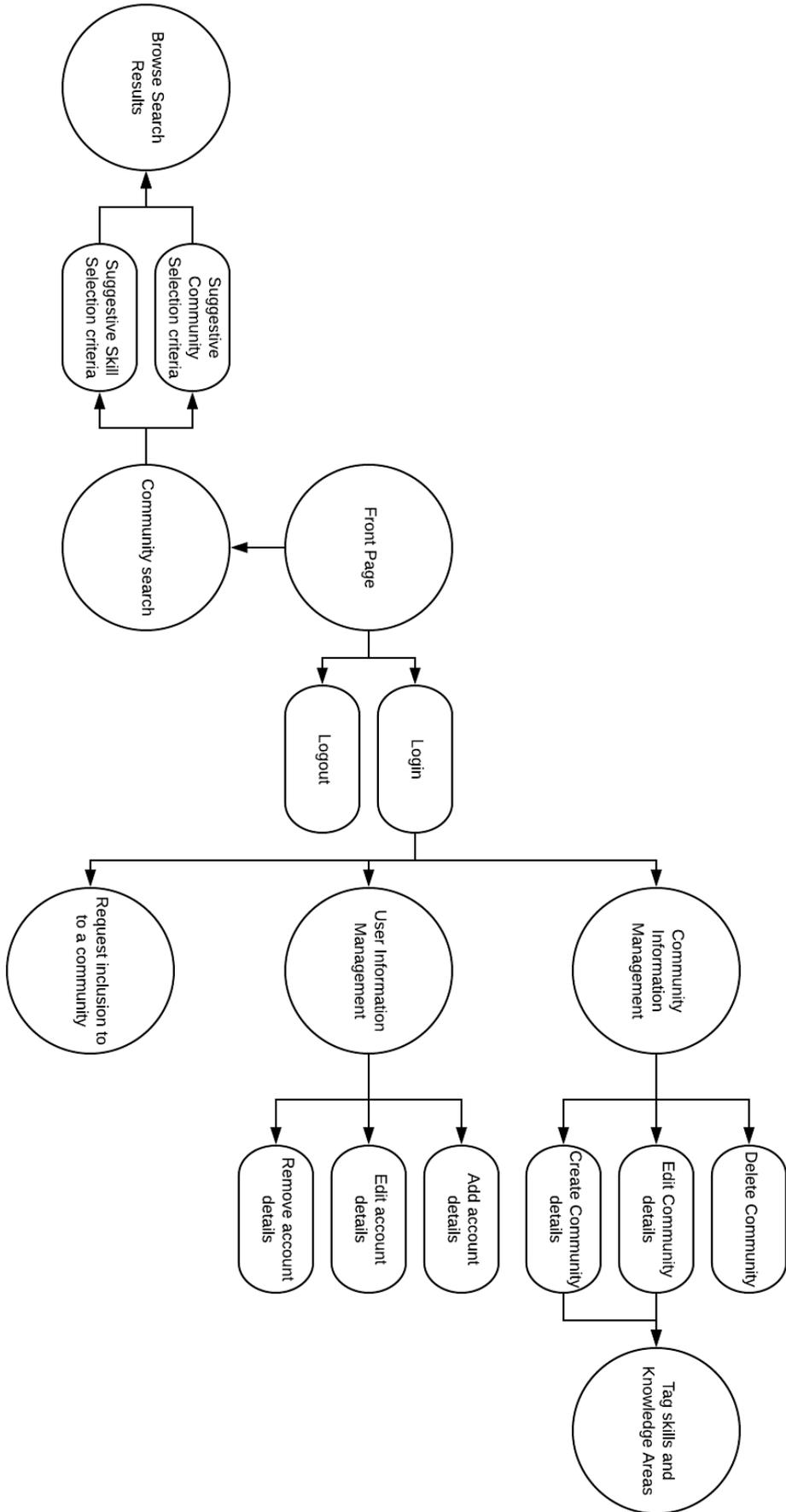
Appendix 3: Django Framework Sequence Diagram



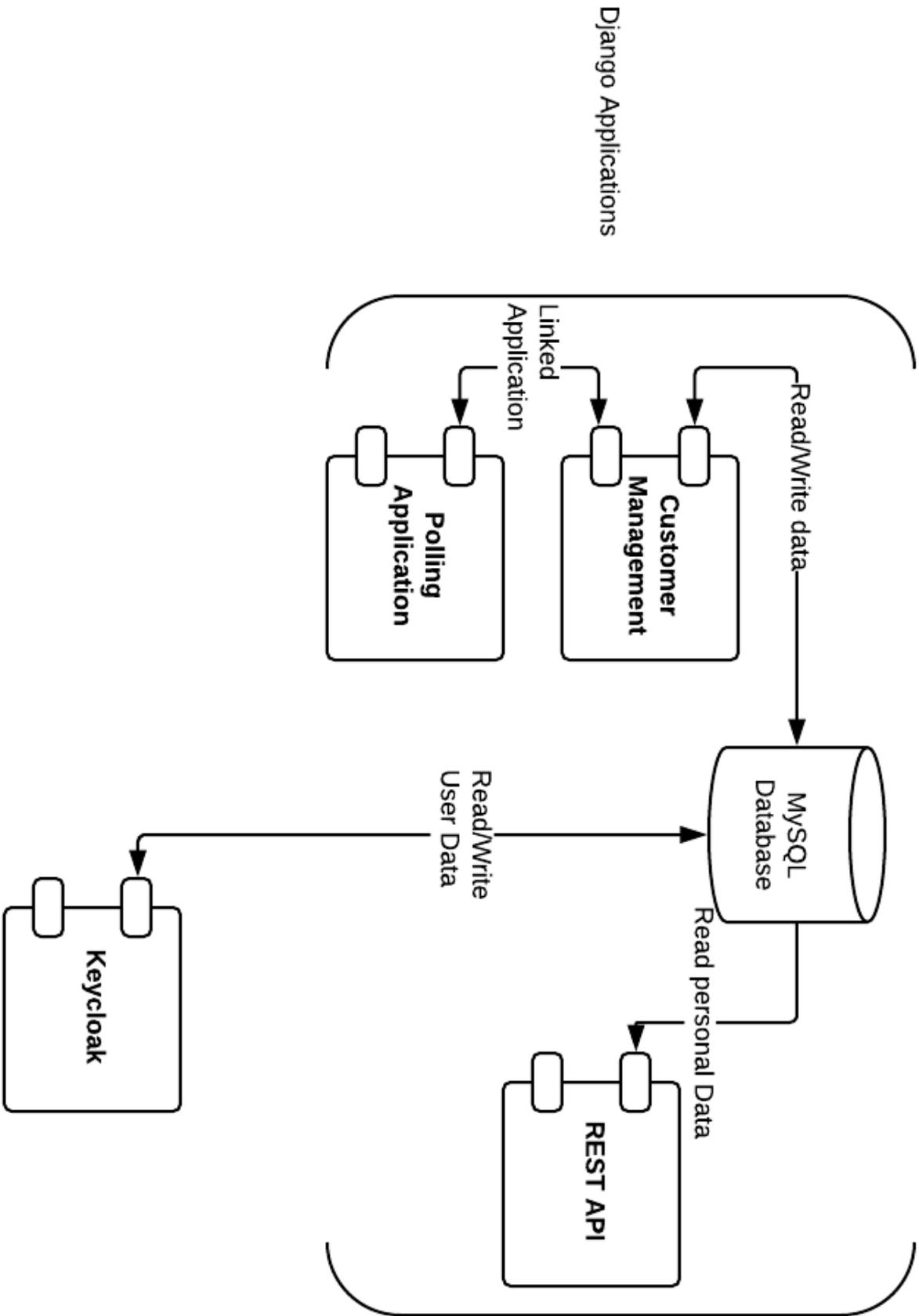
Appendix 4: Entity Relationship Diagram



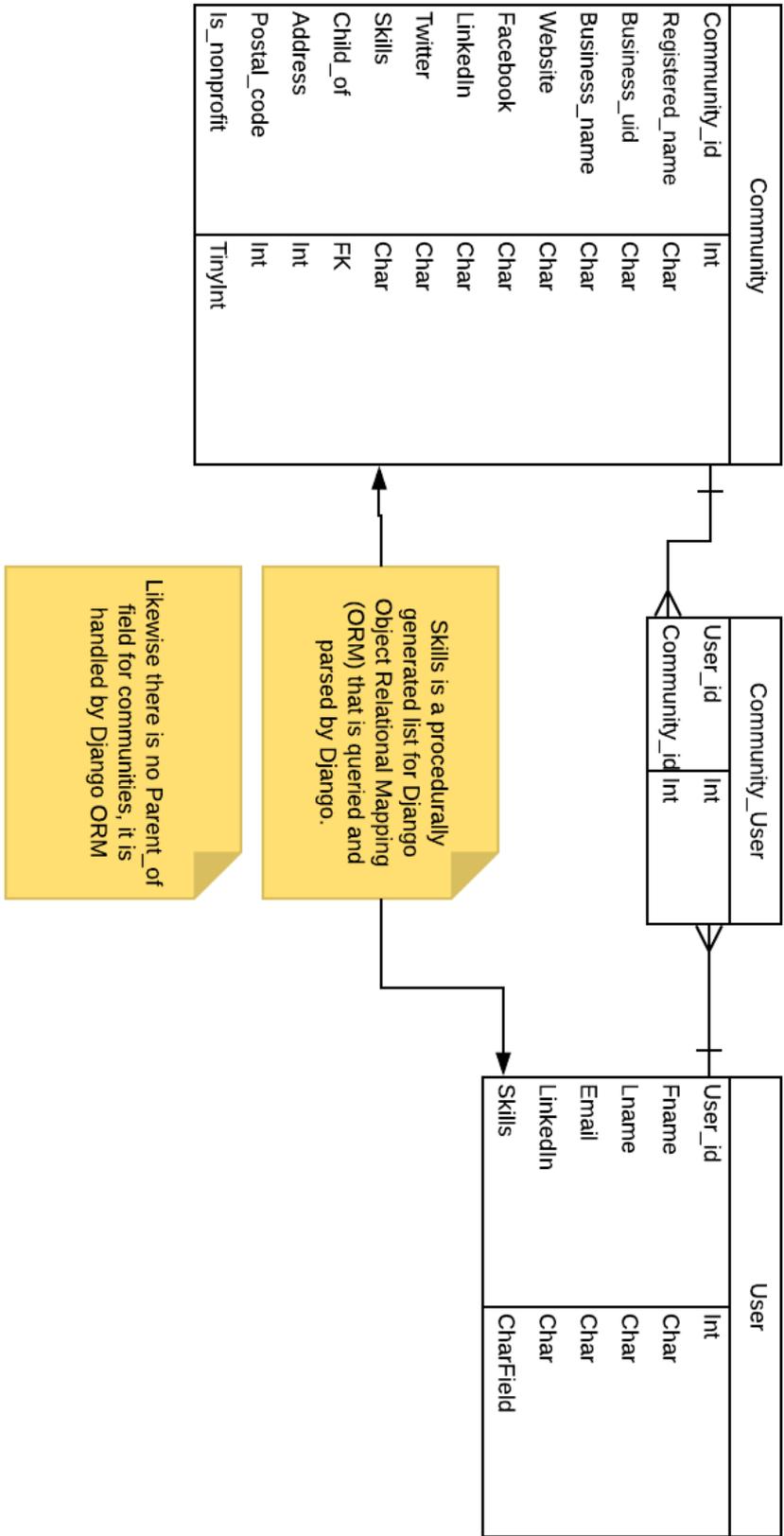
Appendix 5: Mind Map



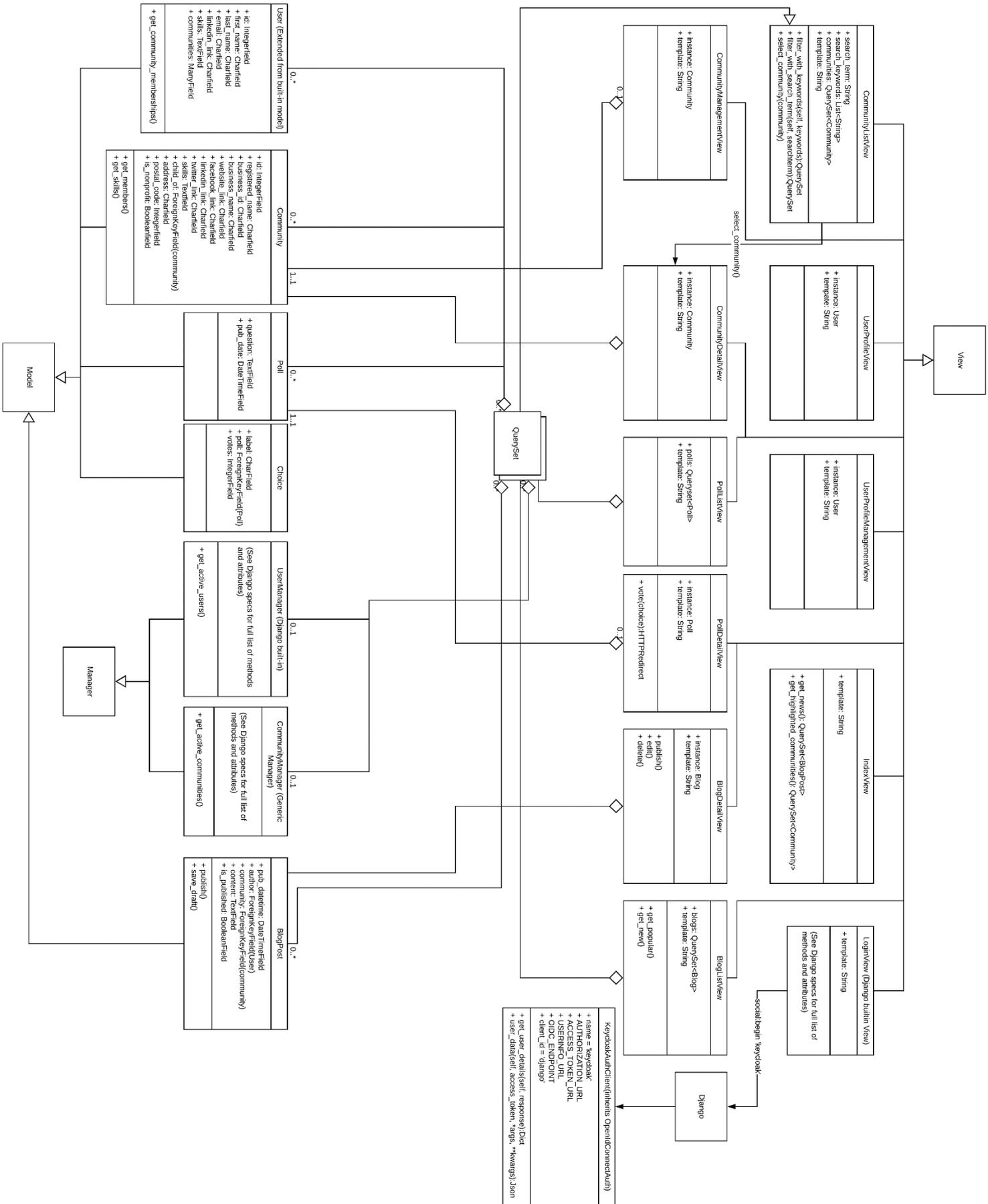
Appendix 6: Architecture-diagram



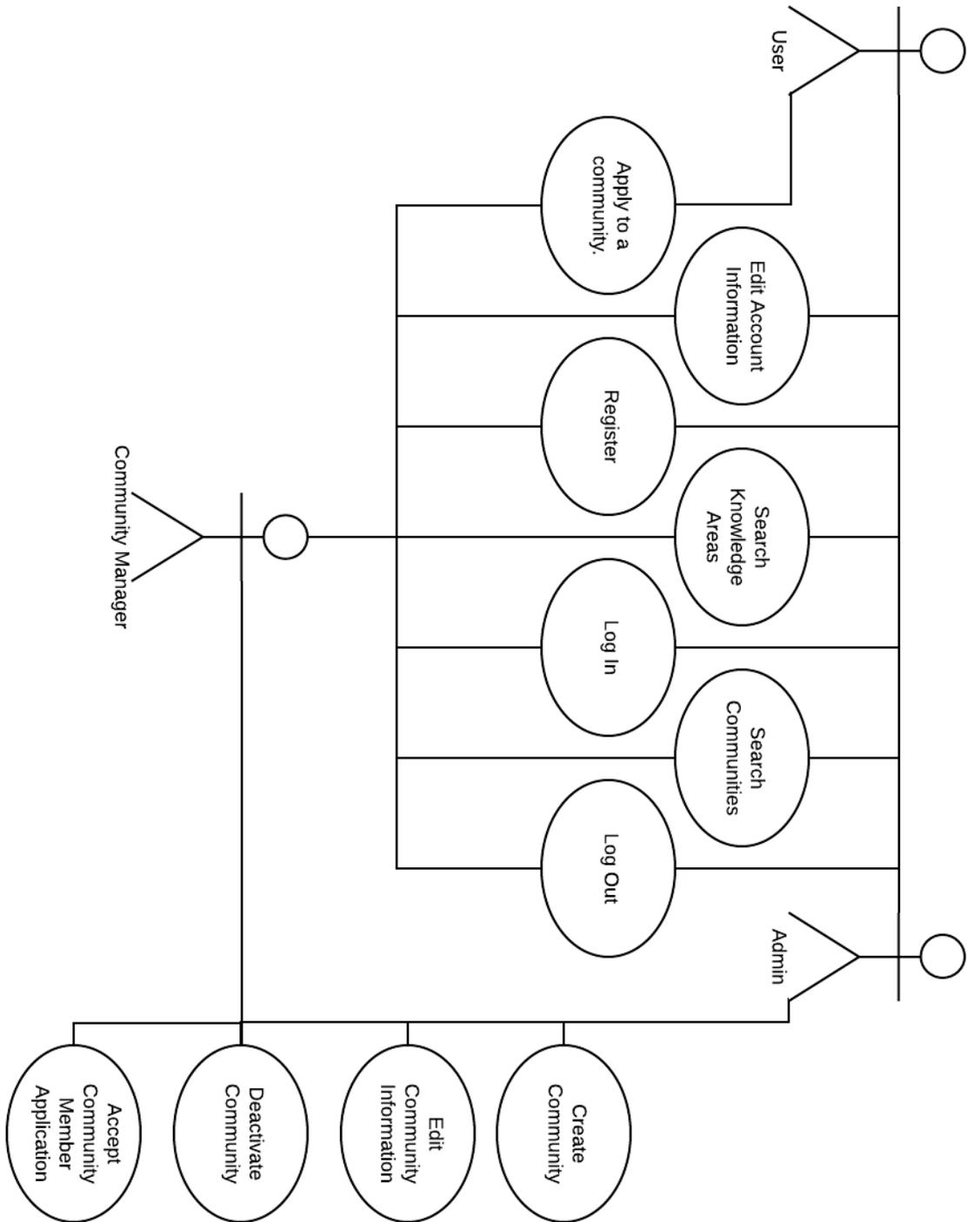
Appendix 7: Database Diagram



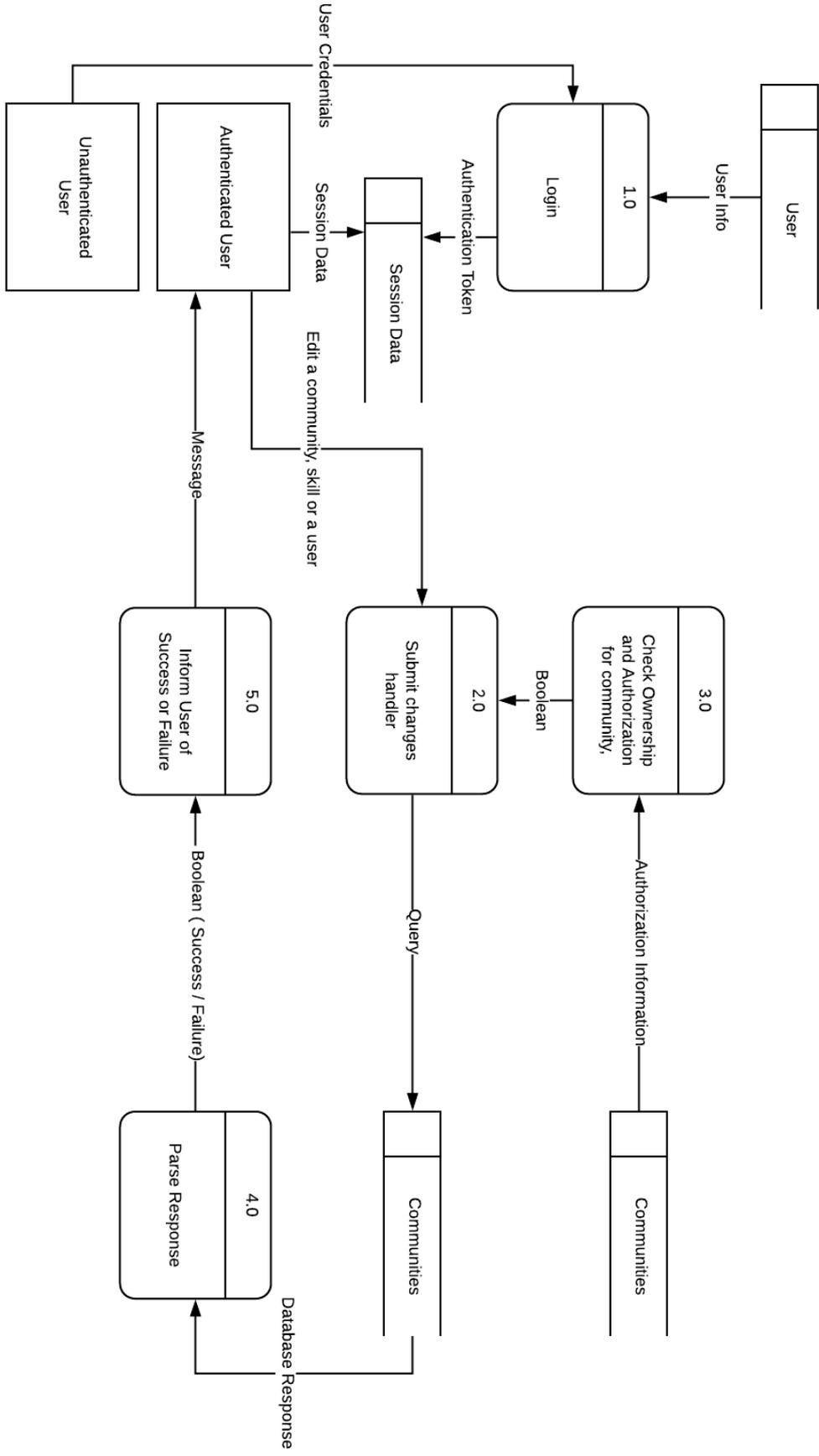
Appendix 8: Class Diagram



Appendix 9: Use case diagram



Appendix 10: Edit Entities Dataflow diagram



Appendix 11: New Community Dataflow diagram

