Tallinn University

Institute of Informatics

Zahhar Kirillov

Towards a Standardization of On-line Business Cards

Master Thesis

supervised by

Vladimir Tomberg, MSc

Author: .... “........” 2009

Supervisor: .... “........” 2009

Head of the Institute…... “........” 2009

Tallinn 2009

Declaration

Hereby I, Zahhar Kirillov, declare, that this Master Thesis is my personal achievement and a result of an original investigation. It has not been submitted for any academic degree before.

……………………	……………………
(date)	*(signature)*

Acknowledgements

I thank my supervisor Vladimir Tomberg for many helpful advices he shared on the details of my thesis. I am also thankful to the Institute of Informatics, for my last three years were spent among active and intelligent people, and for an opportunity to benefit from their broad knowledge.

Thanks to my dear friend and colleague Andrei Filimonov, who is true *Ruby on Rails* expert, for having time on my issues. Further thanks goes to Anton Shilov for proofreading this paper and to Andi Meister for helping me with composing summary in Estonian.

I sincerely thank my parents for nudging me towards a finish line and, finally, heartfelt gratitude to my wife Nataly for her's faithful support and encouragement.

Abstract

Every year organizations are losing money due to outdated contact information about their customers and partners. The purpose of this research is to offer organizations a distributed approach to the contact information management, that would keeps contact details up to date and may become a solution of the data quality problem.

The cornerstone of the proposed approach is an on-line business card, encoded in the machine-readable form. The research investigates, how far existing web-standards related to contact information management (vCard, FOAF, OASIS CIQ) are applicable for the on-line business card exchange system, analyzes and compares specifications to select a leader. During the research author developes a web-based prototype application, that visually demonstrates benefits of distributed approach to the contact management

Current work, originally written in English, consists of 5 chapters with more then 18 000 words, includes 6 figures, 10 tables, 9 listings, 2 appendixes and has 59 content pages in total.

Keywords: *Web2.0, Semantic Web, customer relationship management, contact management, prototype, contact details, software development, Ruby, Ruby on Rails, RoR, CRM, XML, vCard, OASIS, CIQ, FOAF, ontology.*

Contents

Chapter 1
Introduction

The first chapter gives a brief introduction to the research subject. Backgrounds of the data quality and electronic data interchange are described here, what uncovers a main problem this work is aimed to solve. Study goals, as well as derived tasks, are set and expected results are proposed. Finally, this section discusses used research design method and specifies thesis structure.

1.1 Background

Without doubts, the Internet has changed the world. In particular, one of the main aspects of great changes induced by wide spread of World Wide Web (WWW) are communication technologies, daily used by both business and private Internet users. Personal computers once connected to the Network transform into powerful messaging centers, ready to provide access to millions of other people all over the world using text, voice, video or any combinations of those.

As a result, the amount of contacts of an average Internet user has increased dramatically. Almost everyone can find hundreds of contacts belonging to different people and organizations, he used to communicate with, now saved “just in case” for future use in his phone contact register, instant messenger's contact list or mailbox address book. As this data is usually distributed across numerous devices or applications, it needs synchronization and periodic updates in order to keep records up to date and to enable quick search of actual contact details.

Similar situation, with some corrections, can be observed in business segment as well. With help of the Internet, companies get access to both national and global markets with thousands of potential partners and millions of customers. In order to efficiently communicate with them, sell products or provide services, organizations have to apply certain processes, often known under common term called *Customer Relationship Management* (CRM). CRM can be described as interactions with the data bank of various information about customers in order to enable the most efficient communication with them and support of other business processes (sales, accounting, etc). The interest in CRM applications becomes more and more strong every year. According to *Gartner* (2008a, 2008b), CRM software market revenue increased about 23% in 2007, approximately 14% in 2008 and is set to continue growing at least till 2012, when the market is projected to expand by 50% compared to today.

There is hardly need to explain how customer information stored in CRM can be used to provide a variety benefits to the organization. But the cornerstone of CRM advantage is the quality of stored data. Experts say (Betts, 2002), that a company can spend millions on creation of CRM solution and supporting it up and running, but would not get return on investment just because the quality of data is low.

1.2 Problem Statement

According to Juran and Godfrey (1999), data is of high quality if it is fit for intended usage during operations, decision making and planning. Data is fit for use if it is free of defects and possess desired features.

Poor data quality is not only unsuitable for intended uses, but also it can cause impressing financial damage to organization handling low quality data. Ed Peelen's conducted research in 2006 showed, that due to lack of quality in customer data Dutch businesses have to deal with €400 million extra costs every year (Peelen, 2006).

These expenses consist of:

- postal charges on wrongly addressed invoices;
- extra product deliveries do not arrive at the right address from the first try;
- staff overhead on placing calls to outdated phone numbers or sending emails to not-ever-existent mailboxes;
- duplicate processing if one customer is registered in CRM twice or more times.

The same numbers of negative profit in United States were $611 milliards per year, committed by the research of The Data Warehouse Institute, but as source says, "*the real cost [...] is much higher*" (Eckerson, 2001).

Even in hypothetical 100%-accurate CRM data quality will degenerate in time, as contact data is not constant. Experts say (Betts, 2002), that 2% of records about a customers become obsolete in one month because of migration, marital status change or death. Contact details become even more volatile in unstable social and economic environment, like the global financial crisis. More companies move out of offices to cut down expenses; people look for less expensive service providers, hence, their bank account numbers and phone numbers may suddenly change; merges and acquisitions of businesses influence organizations' names and identities.

Summarizing the problem: *organizations are losing money due to obsolete customer information they have and use, particularly, due to outdated contact information*.

On the other hand, customers are also impacted if their valid contact details aren't accessible:

- after change of postal address, incoming mail can be read by a new owner and paid subscriptions may be wasted for some time;
- incorrect bank requisites make impossible money transfers;
- wrong phone numbers can cause missing an important call.

Usually, customers try to inform their partners about changes in contact details, but it causes customer to know exactly: whom, how and when to inform. In case of substantial changes and a large number of associates, informing them becomes a rather complex task in terms of resources (first of all, time and money), while the delay may also impact reputation.

For example, one of the far-famed case, affected tens thousands organizations across the Baltic, was renaming one of the biggest financial institution in this region – *Hansapank* (Estonian company name; brand was known in Latvia as *Hansabanka* and in Lithuania as *Hansabankas*) into *Swedbank* on March 17, 2009 (Hansapank, 2009). As a result, all business customers and partners of this bank have to update their document templates and homepages, to replace old name with the new one. Nobody knows, how long will the old name circulate on invoices and contacts created after the notification about changes. And what is more, even the official website of *Swedbank* still[1] displays to its visitors Secure Socket Layer (SSL) certificate issued by *VerySign*[2] to the old name, *Hansabank*. Possible impact is not yet analyzed, but there is direct evidence of slow reaction to changes and possibility to mislead customers.

How parties can inform each other about changes in contact details? A straight-forward solution is to send to everybody party "knows" a notification about changes in contact details (it is just like sending every associate new business card once it is printed with updated contact details on it). But

1 True on May 1, 2009, updates check at https://www.swedbank.ee/
2 http://www.verisign.com/

such approach has some weak points:

- it takes time to notify all the associates about changes;
- notification messages being sent to foreign partners should be translated to language they understand;
- it is complicated or impossible to make sure recipient has received notification and it was not ignored;
- errors and mistakes can occur if notifications are being processed by human – quoting one of the Murphy Laws, "*computers are unreliable, but humans are even more unreliable*";
- database of all associates has to be created and maintained to prevent sending "spam" – the same notification to the same recipient over numerous of communication channels (e-mail, *Skype*, Short Message Service – SMS, etc).

The other method sometimes used by organizations is to periodically send to customer information request, in order to find out if his contact details have changed since last check. Figurative, this is like asking one's business card every time meeting him. This approach also has some serious drawbacks:

- if primary contact details have been changed, it may be impossible to update them as customer will not get a request sent to outdated contacts;
- resources are wasted in case overwhelming majority of data remain unchanged since last check;
- frequent information requests can annoy customers who's contact details have not changed;
- if database to be checked is large, complicated and expensive tools are needed to automatically request information by traditional ways (plain mail, phone, e-mail) or impressive human resources should be allocated;
- a significant number of errors still may occur if information is not processed by machines.

Both methods have common negative features: they are acquisitive for resources, redundant and unreliable, as humans are involved on the both sides of communication. Moreover, described contact management methods still utilize old-time approaches: they are off-line or only partially on-line and poorly automatized, that opens the path for significant improvements can be done by means of modern on-line techniques.

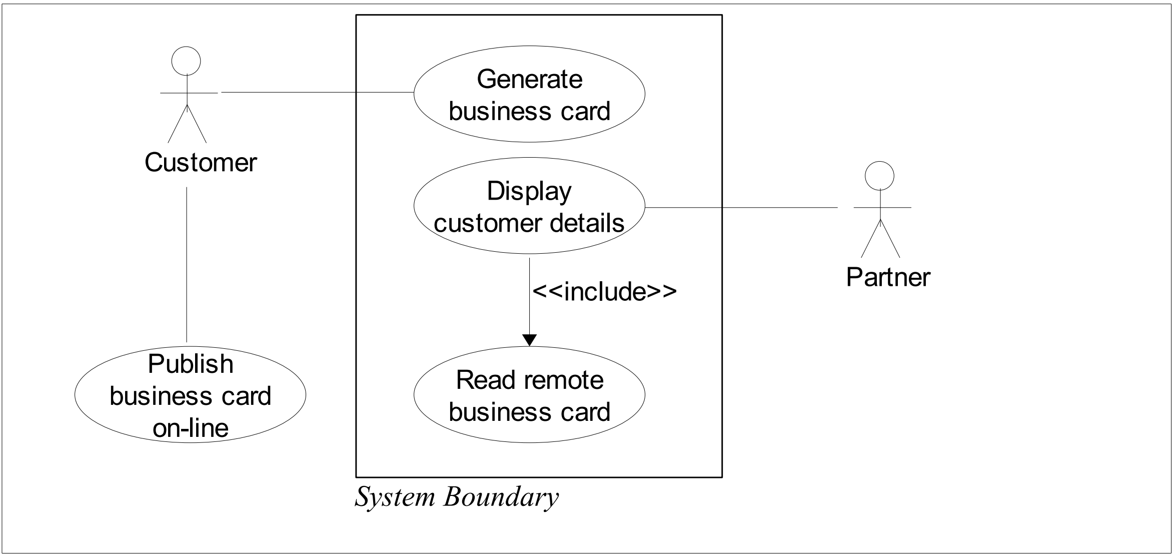
1.3 Proposed Solution

Nowadays countless millions of users have some kind of "self-controlled" personal Internet space, like a blog or a homepage. Organizations are also well-presented in the Internet: only in Estonia there are about 70'000 (EENet, 2009) registered .ee top-level domains (currently registration is available only to organizations and individual entrepreneurs), compared to about 150'000 organizations registered in Estonia in total (RIK, 2009).

According to the author's vision, *while the Internet is being actively used by both sides – information provider and information consumer, there could be developed a new contact management approach, that would help both parties to synchronize changes in contact details and update identity credentials in no time, with low setup costs and maintenance expenses.*

Briefly, author's idea of the next generation on-line business card exchange system (Figure 1) can be defined as follows: *a customer publishes his "digital business card" (contact details and identity credentials) on own personal Internet space in an agreed machine-readable format invisible for humans. To share his business card with associates, a customer provides an URL of the homepage. This URL serves as a unique identifier of a participant. If other participant has a CRM application, capable to read this format and then periodically to check given link against changes in business card, party will be keeping data about this customer up to date.*

Figure 1. Use case diagram of proposed system.



To make the concept clear, key features of the proposed system should be specified:

- it is not "yet another social network" or "yet another standard" – it is more like a framework, demonstrating how today's best practices, standards and technologies can be integrated to solve specific applied problem;
- it is based on distributed approach, so there is no any central data warehouse, but personal data of every customer is located on his own homepage totally under his control, so only

customer himself decides, what personal information he will make public;

- the role of the customer, who just wants to be accessible by any of his associates, but without notifying each of them about changes, is reduced to keeping his digital business card filled with actual details;

1.4 Research Objectives

Since the thesis' author is practicing software developer and an entrepreneur at the same time, it would be incorrect to limit research with theoretical exploration of proposed solution only.

1.4.1 Goals

The goals of current research are:

- to investigate, how far existing web-standards related to contact information management are applicable for on-line business card exchange system;
- to build-up a prototype of the proposed system, that later can be re-used by anyone who likes the idea.

1.4.2 Scope

Still, certain assumptions must be made to determine the exact scope of research.

- Declared problem of the contact details exchange is observed strictly in the context of constantly growing Internet society. Problems of contact exchange among off-line customer group (who don't use Internet or do not have yet a personal Internet space) would not be discussed in this work.
- The research is focused first of all on the organizations of any type (commercial companies, public authorities and non-profit organizations), generally speaking, on the all legal persons. This amendment is done as lots of private persons may not be interested in sharing their private contact details globally via the Internet. Juridical persons, *vice versa,* have evident motives to give publicity to their full identity details. Author assumes, that his theory will get more followers (in case if it is strong enough to get any) if will be introduced at first to the organizations and, if adopted by them, later may be expanded to involve private persons as well;
- The evaluation of the proposed method and developed prototype is out of scope of this paper and will be discussed later in the separate articles during author's future study. This is done to keep research compact and focused on the thorough design (both theoretical and

empirical) of projected system.

1.4.3 Questions

As the goals of the research are quite complex, in order to achieve them with desired quality, a series of research questions is listed below.

1. What Internet environment features and information management trends should be taken into account before development of a long-life approach for contact management?
2. What web-standards are designed to exchange contact details?
3. Which standard is the most suitable as the basement for proposed contact information management system?
4. How to make a simple, but functional prototype, to help other developers integrate an idea of on-line business cards into their applications?
5. What are possible “bottlenecks” of the proposed approach and what are the most important areas for future study? (Note: to be done firstly without complex evaluation with real end-users).

1.5 Expected Results

The author estimates, that achievement of the goal will provoke a public discussion (perhaps, even on the international scene) about proposed on-line business card exchange system, rather than puts the final point in the research topic. That's a reason why some words are needed to be said about expected results.

1.5.1 Direct Results

The direct results will be reflected in:

- consolidation of the exhaustive information about contact information management standards in one document, their comparison and reasons for selecting one of them as the dominating standard;
- launch of a prototype web-application (consists of parts described below), to visually demonstrate the benefits of the proposed method, enabling organizations to keep their contact details up to date;

The prototype's functionality will be divided between the following two parts:

- a generator, that allows customers to create their digital business card, encoded with selected specification;

- a software agent, that by user request checks web-sites, supposed to have business card installed, and if business card is found there, extracts published contact details and displays them to the user (similar to “white pages” on-line catalog).

To share research results, author provides outcomes (including source code of developed prototype) on the project homepage[3] under *Creative Commons Attribution* license[4]. This license lets others distribute, remix, tweak, and build upon this work, as long as they credit author for the original creation.

1.5.2 Long-term Results

The expected long-term impact should be a wide spread of the approach, described here and demonstrated via prototype. It can be achieved by energies of the enthusiasts' community and with support of the business community.

After publishing current thesis, when the functional prototype becomes available to public, author plans to invite different associations and non-profit organizations (first of all those representing Estonian Internet community), to evaluate and discuss the improvements of the developed method. Organizations, which presupposed to be invited are: E*stonian Association of Information Technology and Telecommunications*[5], *Estonian Association of E-Commerce*[6], *Estonian CIO Club*[7], state universities and others. Also, a workshop about proposed contact information exchange method is planned for everybody, who will show interest to the offered solution.

The similar way went many present de-facto Internet standards, for example, *Dublin Core Metadata Initiative*[8] – a standard for cross-domain information resource description. *Dublin Core* standard was originated in Dublin, Ohio, United States by local library consortium who hosted a metadata workshop in 1995. Today *Dublin Core* implementation is defined in ISO and NISO (*National Information Standard Organization*, USA) standards and widely used for description of digital documents of any kind.

Author admits, that the feedback from the community will help to more effective to develop a proper solution around proposed method and offered prototype.

3 http://ciq.bits.ee
4 http://creativecommons.org/
5 Original name is “Eesti Infotehnoloogia ja Telekommunikatsiooni Liit”, http://www.itl.ee
6 Original name is “Eesti E-kaubanduse Liit”, http://www.e-kaubanduseliit.ee
7 Original name is “Eesti IT-juhtide klubi”, http://www.itklubi.ee
8 http://www.dublincore.org

1.6 Research Design

Since the author does not intend to use any statistical controls or randomly chosen control-groups, the design of the research cannot be considered as experimental or quasi-experimental and any of quantitative approaches are not applicable for the current work. For this reason, some qualitative non-experimental approach should be selected in order to guide the author through the process of collection and systematization of domain data. Because the final goal of the thesis is to synthesize new knowledge useful for society and implement it in practice, (research is dealing with proving hypothesis by means of development new application prototype), the design-based research (DBR) method will be used.

The author chooses DBR approach among others non-experimental designs subjectively, because he considers that this way, defined by Wang and Hannafin (2005) as "*a systematic but flexible methodology aimed to improve [...] practices through iterative analysis, design, development, and implementation, based on collaboration among researchers and practitioners in real-world settings, and leading to contextually-sensitive design principles and theories*" is the best description of sought paradigm.

The main three values of DBR from the author's viewpoint are the following:

- DBR is continuous – the phases of design, implementation and evaluation can recur as many times as needed, until the satisfactory result will be obtained (Collins et al., 2004);
- DBR is pragmatic – despite it is theory-driven, the approach promotes fundamental understandings, and, being strongly connected to real-life contexts, solves practical problems (Design-Based Research Collective, 2003);
- one of the main principles of DBR should be shareable results for practitioners and other designers (Learning Theories, n.d.).

The present research consists of two parts: theoretical and the empirical one. The main research instrument for the first part of the work will be a document-based research, when a new knowledge will be constructed through the analysis of the existing theories and documents. In the empirical part, the "newborn" knowledge will be extended with the experience gained from it implementation. Despite the fact, that, as already mentioned, this research lacks of proper prototype evaluation by real end-users, author will do his best to provide an adequate feedback and appraisal about the work done – so-called „self-evaluation". A proper evaluation of the proposed approach involving the real users is planned as a separate article during author's further study.

All of the above gives the author a confidence that research results will "*consider the role of social context and have better potential for influencing educational practice, tangible products, and programs that can be adopted elsewhere*" (Barab & Squire, 2004) what completely corresponds to the expected results mentioned in chapter 1.5.

1.7 Outline

The thesis consists of five chapters. A sort description of what can be found in the next four chapters can be found below.

Chapter 2 introduces relevant background theories applicable to modern Internet environment where the solution will be implemented, as well as takes a look into the nearest future to get a basic idea about how Internet space may change.

Chapter 3 focuses on theoretical exploration of domain knowledge. Chapter gives overview of approaches and studies a sample of Estonian organization, builds its contact profile to be used later for developing and testing the prototype. Second part of the chapter studies all major contact information exchange standards, compares them and selects proper standard for implementing prototype.

The 4th Chapter is aimed to create a prototype application itself. Both modules (generator and software agent) are implemented to make possible check the proposed solution in practice. Workflow diagram of the final application is published there. The innovate *Agile Modeling*[9] software modeling methodology and lightweight *Getting Real*[10] software development methodology are used to create the prototype.

Finally, Chapter 5 makes a conclusion of the work done in previous chapters and sums up the strength and weaknesses of developed solution. The directions to the future study will be outlined in the final chapter along with summarized research achievements.

Chapters end with a short-list of facts, that summarize key results of the exploration. This helps to keep track of the research and to focus on its objectives.

9 http://www.agilemodeling.com/
10 http://gettingreal.37signals.com/toc.php

Chapter 2
Related Work

The first principle task, that needs to be solved during the research is to learn a selection of available web-standards, designed for contact details exchange. This should be done in order not to “reinvent the wheel”. Instead, best available practices should be reused to get desired results faster and without excessive theoretical exploration. Besides, it helps to achieve the best possible interoperability with other similar systems if they are being developed simultaneously with this research or will be developed in the future. For this purpose, the current web environment, coupled with information management, trends should be described. This knowledge provides strong base for selection of proper standard for description of contact exchange system, what fits the Internet environment and lasts long time.

2.1 From Web 2.0 towards the Semantic Web

Although the inventor of WWW, Tim Berners-Lee says that “I *think Web 2.0 is of course a piece of jargon, nobody even knows what it means.*” (Anderson, 2006), a loads of people interpret this phrase as the key property of WWW, allowing them not only to read the content of the Web, but also to easily share their own content in any form with anyone in the Internet. As a result, Internet today is a huge amount of data in different formats: texts, videos, images (etc.). Tthe amount of information available on-line is growing daily. This data is provided in hundreds of different formats and is distributed across great number of information systems and web-pages.

For example, someone wishes to buy a computer. A series of decisions have to be taken before: where to buy it – in on-line store, in the nearest retail shop or via an Internet auction? What kind of a computer is needed? What are the technical requirements? What shop sells the desired PC for the most attractive price? Can the customer trust a shop? What is the quality of service provided by it?

Does anyone have positive experience buying there? And, in case if off-line shop is selected, customer possibly would like to check if chosen model is in stock and what if the working time of the store.

To complete all these checks and decisions, person needs to make several queries to the search engine by relevant keywords, then visit a dozen of web-sites, gathering disembodied data piece by piece. Although, price-comparisons available on price-hubs, customer feedback and experience about certain seller or computer's model can be found in social networks. Detailed technical specification can be found on the web-site of the manufacturer and stock's working hours on its homepage.

Instead of wasting time sitting in front of a computer and reading long texts, making calculations or waiting for one's opinion, the request to some sort of web-service could be performed to gather all relevant information from various sources on one single page. To make it possible, a web-based software capable to systematize, classify and compare user-generated content, provided in any human-understandable form, should be developed. As computes are lack of intelligence, that is inherent to humans, machines experience difficulties with understanding an actual meaning of the document and its relationships with other documents in the Web.

The approach, that allows to get rid of named obstacles, was defined as the Semantic Web: "*The Semantic Web is not a separate Web, but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation*" (Berners-Lee et al., 2006).

In brief, the Semantic Web is a set of initiatives to explain meanings of documents through the use of standards, markup languages and information processing tools. If the machine knows semantics of the content being processed, it can make one step forward to human intelligence and be able to relate, compare and rate different type of web-documents. Hence, it promotes WWW to the next level where data can be processed by automated tools almost the natural way like people do it.

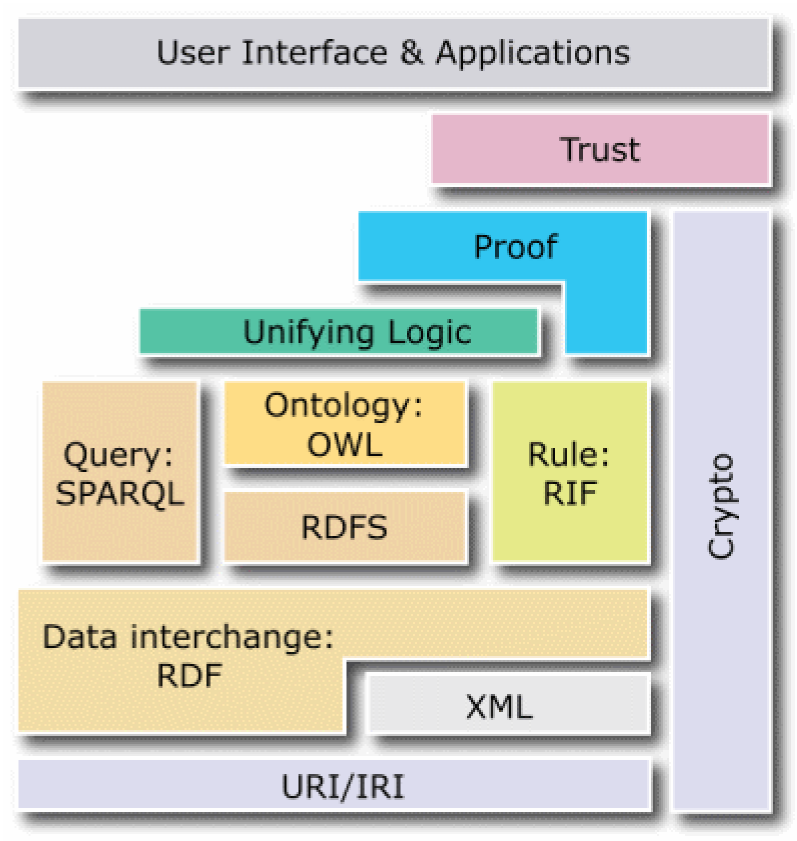
But what exactly should be done with data to make it ready for being processed by machines? In terms of Semantic Web, meanings are applied to the documents through the use of metadata and ontologies ("*a set of representational primitives with which to model a domain of knowledge*" (Encyclopedia of Database Systems (n.d.)), described in appropriate standards.

2.2 Standards and Technologies of the Semantic Web

Obviously, it is impossible to deliver a large-scaled effect like Semantic Web pretends to do, without elaborate design of its architecture. WWW Consortium[11] (W3C) represents architecture of Semantic Web as a hierarchical structure (see Figure 2), where one layers are built on top of the others and thus inherit their properties and features.

Unified and agreed set of standards and methods (Unicode, URI/IRI, XML and RDF), make up a strong basement of the Semantic Web. As a lot of abbreviations are used in W3C diagram, the meaning and the purpose of the most significant technologies should be clarified.

Figure 2. "Layercake" diagram of the Semantic Web. (W3.org, 2009).



2.2.1 Unicode

A variety of character encodings (also known as code pages) was previously used in computer industry due to limitations in software architecture, didn't allowed to simultaneously operate with character sets of multiply languages. The solution to this problem was invented long ago in 1991 in

11 http://www.w3.org/

form of *Unicode*[12] standard, developed and maintained by *Unicode Consortium*, which is supported by most of information- and telecommunication technology (ICT) industry trendsetters – *Adobe, Apple, Google, HP, IBM, Microsoft, Sun* and others.

Massive use of *Unicode* started since mainstream applications (*MS Windows, Java, .NET, Mac OS, KDE*, etc) adopted *Unicode* as de-facto standard for their internal representation of data. (approx. around 2000). Simultaneously, web-developers choose *Unicode* as a default de-facto encoding for HTML-documents.

Today, in 2009, the latest version of Unicode 5.1 consists of more then 100'000 characters, including symbols of most known languages from ancient times to nowadays, dead writing systems and special character sets like Braille patterns, music, domino tiles, etc. All these symbols can be mixed in one document without any significant problems, allowing digitalization of almost any knowledge originated by people.

2.2.2 URI

Another fundamental component of the Semantic Web is *Uniform Resource Identifiers* (URI). As distinct from prevalent *Uniform Resource Locator* (URL, or web address), what is actually just one subset of URI used to identify a location of documents in WWW, the URI itself specifies general schemes for identifying virtually any resource through specific syntax and protocol. URI and its syntax is defined and described in RFC3305[13] and RFC3986[14] documents.

The idea behind an URI is that any existing resource (physical – like humans or paper books; digital – electronic document or web-service; abstract – like relationships) can be uniquely referred by a compact sequence of characters in agreed syntax, providing information about the type of the resource and methods for retrieving it.

RFC3986 defines the general pattern for URI:

```
<scheme name> : <hierarchical part> [ ? <query> ] [ # <fragment> ]
```

The scheme name is just a label, typically it named after the protocol used to retrieve a resource. The hierarchical part is schema-dependent and usually consists of authority information about terms of accessing resource and full path information. Query and fragment are option parts, aimed

12 http://unicode.org/
13 http://www.ietf.org/rfc/rfc3305.txt
14 http://www.ietf.org/rfc/rfc3986.txt

to navigate inside the resource to its specific part.

For example, URIs are:

- `ftp://username:password@example.com:21/path/node/filename.txt` – this URI describes a digital document `filename.txt`, located at `example.com` website that can be accessed by password-protected File Transfer Protocol (FTP).
- `mailto:somebody@example.com?subject=test&body=hello` – this URI describes an e-mail message sent to `somebody@example.com` through SMTP protocol with pre-filled subject and body fields.
- `urn:swift:iban:ee:909999123456789012` – in this example the *International Bank Account Number* (IBAN) is represented by *Uniform Resource Name* (URN) schema, serves as location- and protocol-independent URI.

In total, there are about 70 permanent URI schemes registered in *Internet Assigned Numbers Authotity* (IANA)[15] – organization, maintaining, among other responsibilities, a register of URI schemes. Beside that, competitive amount of schemes are in use without official registration. As the most schemes are describing references to some digital content can be addressed via the Internet, URI is the perfect choice for development applications for retrieving and processing of such content.

2.2.3 XML

On the next layer in the Semantic Web architecture raises Extensible Markup Language (XML)[16]. XML is the another open standard, recommended by W3C for implementing application-specific custom markup languages. Like widely known HTML, that is used to describe web-pages rendered by web-browsers, XML is also a subset of the Standard Generalized Markup Language (SGML) and utilizes similar syntax and rules. This simplifies people, who are already familiar with HTML, to begin using XML, when custom syntax is needed to prepare data for processing by the computer software.

Table 1. XML-encoded data compared to human-readable data.

Human-readable	Machine-readable

`John Smith,`
`president of “Example Company”`
	`<organization>`
	 `<name>Example Company</name>`
	 `<president>John Smith</president>`
	`</organization>`

15 http://www.iana.org/
16 http://www.w3.org/XML/

In the right column of the Table 1, an abstract company name and the name of its president are represented in machine-readable way.

As users are given almost unlimited freedom to name tags and attributes in XML, a problem can occur when two or more semantically different elements are referred with the same tag name within one document. To avoid this, the XML-namespaces were introduced.

Table 2. XML-namespaces and the same names of semantically different elements.

Without XML-namespaces	With XML-namespaces
`<organization>`	`<organization>`
` <name>Example Company</name>`	` <org:name>Example Company</name>`
`  <president>`	`  <president>`
`   <name>John</name>`	`   <person:name>John</name>`
`   <lastname>Smith</lastname>`	`   <lastname>Smith</lastname>`
`  </president>`	`  </president>`
`</organization>`	`</organization>`

The tag `<name>` in the left column of the Table 2 is used firstly to describe organization name and later to describe person's first name, what is incorrect, because these elements belong to different classes and have different meanings. To correct this, a prefix indicating the scope of tag is used at the right side (both `<org:>` and `<person:>` in the example), making possible for a computer program to distinguish meanings of the `<name>` tag.

Extensive use of XML-namespaces provide developers with a handy tool for resolving possible ambiguity issues between identically named elements, helping people to collaborate while working with complex XML-based documents.

When XML-document reaches a certain level of complexity, developers may want to extend it with features, helping them to make sure, that document content exactly corresponds with the data model (or designed XML structure). Developers may need to specify what data types may be stored in particular XML-elements as well as the vocabulary of allowed elements, attributes and rules (like, how many elements of every kind may be used in the document?). To make it possible, XML-schema[17] was introduced.

XML-document in the left column of the Table 3 is valid only if its structure corresponds to the scheme defined in the right column. The provided XML-scheme assumes, that document has a container element <organization> without any attributes, what may include two other elements, both of string data type. The <name> is mandatory and can occur only once, while the other

17 http://www.w3.org/XML/Schema

element – <president> – is optional and may be omitted.

Table 3. XML-document and corresponding XML-schema.

XML-document	Corresponding XML-schema
`<organization>`	`<xs:schema>`
`  <name>`	`  <xs:complexType name="organization">`
`    Example Company`	`    <xs:element name="name" type="xs:string" />`
`  </name>`	`    <xs:element name="president" type=”xs:string"`
`  <president>`	`        minOccurs="0" />`
`    John Smith`	`  </xs:complexType>`
`  </president>`	`</xs:schema>`
`</organization>`	

There are also a couple of other languages, more or less widespread for describing XML schemes: *Schematron*[18] and *Relax-NG*[19]. Any of those can be used in terms of Semantic Web, but developers should bear in mind possible interoperability issues and consider if selected language is well-supported in all other related tools in use.

2.2.4 RDF

The previous two layers of Semantic Web architecture were applicable to particular documents, describing virtually any structured data. But to proceed further with the idea of Semantic Web, relations and meanings of described objects (possibly distributed across several XML documents), should be explained. How can it be done? Thanks to *Resource Definition Framework*[20] (RDF), anything that has an URI can be put into correspondence with anything having URI. This is done through *triples* – expressions, that generally looks like "*subject-predicate-object*" statements. For example, Table 4 presents statement about relations between organizations and its president.

Table 4. Making statements with triplets.

	Subject	Predicate	Object
Statement	`President`	`Runs`	`Organization`
Example	`John Smith`	`Runs`	`“Example Organization”`

The same way other predicates (or in other words – relations between two things in the world) can be introduced, as well as new entities (objects or subjects) included in the hierarchy. As a result, a some kind of database or graph is formed, what can be compared to well-known Entity-Relationship (ER) Model traditionally used by many database management systems. When relations and roles of every element in the database is known (either it is subject, predicate or

18 http://www.schematron.com/
19 http://www.relaxng.org/
20 http://www.w3.org/RDF/

object in context of specific triple), a simple queries can be executed against triple database.

For example, with ease can be queried:

- what organizations are run by John Smith?
- is there anyone else except John Smith who run "Example Organization"?
- is John Smith somehow connected with "Example Organization"?

The advantage of RDF is that framework provides developers with software-independent tool, what is not tied with proprietary data storage or representation technology. But the most valuable aspect of RDF is that "*if someone refers to something as X in one place and X is used in another place, the two X's refer to the same entity*". All together, using RDF developers get powerful and flexible framework for knowledge representation. (Tauberer, n.d.)

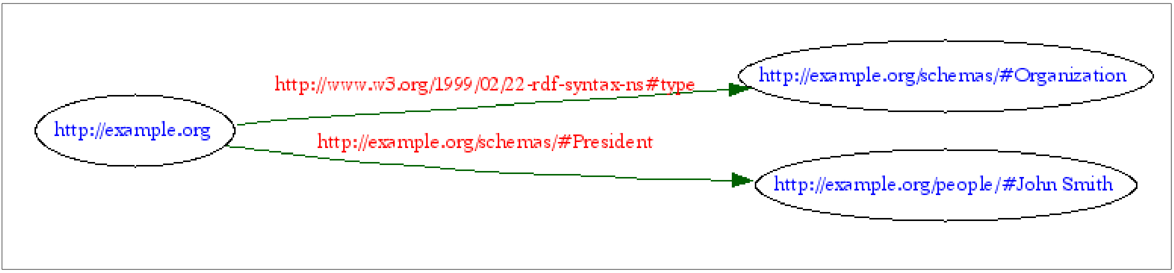
To make representations be understandable by computer software, ontologies described with the help of RDF should be stored in a machine-readable text format (Listing 1).

Listing 1. Example of RDF-scheme.

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:org="http://example.org/schemas/#">
 <org:Organization rdf:about="http://example.org">
   <org:President rdf:resource="http://example.org/people/#JohnSmith" />
 </org:Organization>
</rdf:RDF>
```

W3C offers a tool for validation and visualization of RDF-schemes. Figure 3 is the output of W3C RDF Validator[21] for the RDF-scheme from the Listing 1.

Figure 3. Visualization of RDF-Scheme from Listing 1.


http://www.w3.org/1999/02/22-rdf-syntax-ns#type
http://example.org/schemas/#Organization
http://example.org
http://example.org/schemas/#President
http://example.org/people/#John Smith

John Smith is identified by URI http://example.org/people/#JohnSmith; the "Example Organization" company is identified by its web-address (URL) – http://example.org and belongs to the Organization class; the relationship between them is described by predicate President defined somewhere in the vocabulary located at http://example.org/schemas/.

21 http://www.w3.org/RDF/Validator/

Example above shows, that RDF itself is not capable to define any of the referred resources or predicates in terms of human intelligence, but only describes relationships between classes. Still, there is no definition for President and Organization terms and nothing is known about their relationship, except that they are connected through “Example Organization” instance.

To clarify this issue, an ontology vocabulary – or RDF-scheme[22] – should be built up. RDF-schema is another W3C recommendation and provides an XML syntax for describing RDF data vocabularies through terms “`class`”, “`subclass`” and “`property`”. Like XML-schema provides rule system for XML-documents, RDF-schema provides type system for RDF ones. It describes, how resources can be organized into hierarchy, and which resources can be connected together through specific predicates. RDF-schema also may define even “orphan” predicates does not yet connected to any of the resources. Listing 2 gives an example how class `Organization` and property `President` may be defined with RDF-scheme.

It is pretty clear, that `Organization` is a generic RDF class, and `President` is generic RDF property, available for `Organization` domain. This approach makes possible for other developers, familiar with RDF methodology, to re-use classes and properties already defined via public accessible RDF-scheme instead of creating their own with the same meaning.

Listing 2. Example of RDF-scheme.

```
<!DOCTYPE rdf:RDF [<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">]>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xml:base="http://example.org/schemas/">
   <rdfs:Class rdf:ID="Organization" />
   <rdf:Property rdf:ID="President">
      <rdfs:domain rdf:resource="#Organization" />
   </rdf:Property>
</rdf:RDF>
```

Like XML-schemes, RDF ones can be produced by other tools and standards, like Ontology Web Language[23] (OWL).

2.3 Information Management Trends

To keep the estimated outcome of the thesis' research topical for a possible long time, author should be aware of trends in information management and future of Internet, as experts say,

22 http://www.w3.org/TR/rdf-schema/
23 http://www.w3.org/2004/OWL/

"*Today's Internet stands at a crossroads. Interactive and collaborative Internet usages are proliferating. [...] Current Internet technologies and architectures may not be capable of supporting such sweeping evolution, from an information service to a critical infrastructure underpinning our lives and economies.*" (Future of the Internet, 2009).

While developing contact information exchange standard with an eye toward the future, a few aspects should be kept in mind:

- *The amount of data that organizations are currently facing is staggering, and it is going to continue to increase only.* While data integration is an important part of the business strategy, it becoming more closely aligned with data quality and master data management environments. Data governance will emerge as a required discipline for organizations, giving rise to greater trust. (McKendrick, 2008).
- *Intelligent personal agents will supplement keyword-based search engines with answering user's queries.* Personal agent – a piece of software, that receives some tasks and preferences from the person, performs queries, communicates with other agents, compares information about user requirements, selects certain choices, and gives answers to the user. People would like to seek information from the WWW using natural language and wish to receive results over several documents in a user-friendly form. To make it possible, besides personal agents, additional layer of Intelligent infrastructure services should be delivered to the Semantic Web. (Antoniou & Harmelen, 2004)
- *Significant energies will be aimed to solve the ontology plurality problem.* The more people are involved into building the Semantic Web, the more ontologies they describe and more attempts to integrate pieces of knowledge they make. But it is one thing to develop own ontology and quite another is to make it compatible with other ontologies. As nobody can prohibit users from creating ontologies, the solution must be found to somehow derive meta-ontologies, or to map and extend existing ontologies in automatic- or semi-automatic way, allowing to get rid of multiple isolated ontologies about the same domain. If it would not be done, Semantic Web will "die" in a swamp of unrelated ontologies. Having a few interconnections, they make impossible for personal agents to function better then current keyword-based search engines. (Spivac, 2004)

Without any doubt this list is not final. There are also quite much new trends, like Cloud Computing, growth of truly mobile Internet users with pocket Internet-access devices, etc, but to the author's opining there is no need to mention all of them, as very likely these technologies would not dramatically affect the contact information management.

2.4 Conclusion from Chapter 2

Key points derived by author from the above exploration of Internet standards and trends are:

1. Data integration is undoubtedly important, as volumes of machine-to-machine communications continue to increase, that's why data should be organized by the way of making software agents possible to access it;

2. To promote current Web to the new level, where computers will assist people to solve complex information processing problems, the meaning of data should be explained to the computer programs;

3. Building blocks of Semantic Web are already available: through the use of Unicode, URI, XML and RDF everyone with a basic programming skills can define the data semantics and ontologies and present them in the machine-readable form;

4. While designing a Semantic Web application, any temptations to define own ontologies should be cut with “Occam's Razor”[24], as “*entities must not be multiplied beyond necessity*”; in other words, before defining new ontology it's better to check if there something similar defined already, and to extend or precise it.

24 http://math.ucr.edu/home/baez/physics/General/occam.html

Chapter 3
Contact Information Management (CIM)

There is no single widespread term for "*Contact Management*" or "*Contact Information Management*" (CIM) available, although some sources pretend to define it as "*recording and tracking the people with whom you network*" (eSight, n.d.). This chapter gives overview of approaches, what are applicable for digital contact information management, studies a sample Estonian organization and builds its contact profile to be used later for developing and testing the prototype. Second part of the chapter gives overview of all major contact information exchange standards, compares them and selects proper standard for implementing prototype.

3.1 Scope of the Contact Management

Under the term "detailed contact information" author here and further bears in mind a set of agreed communications services and identity credentials, that is used in communication process. For example, communication services are: plain ground mail, plain old telephone system, cellular phones, instant messengers (IM), voice over the Internet protocol (VoIP), web-sites and web-services (e-mail), etc.

Identity details may vary significantly, and can be: participant's full name, home or work address, telephone numbers of any kind (home, work, mobile), nicknames or account names in IM, URLs of web-sites and e-mail addresses, as well as time periods when associate can be contacted with the desired service, personal preferences against using one or other service (primary, secondary, urgent), personal photo or cryptographic public key file and so on.

There is no exact border what is contact details and what is not. The exact list of essential elements depends on the industry and purpose of using contact details, legal regulations for collecting and storing of such data, personal privacy preferences and other matters.

3.2 Evolution of the Contact Management Systems

With the evolution of users' visions and requirements for contact management, a number of approaches were introduced. The history of sharing contact information started with printed address-books (mostly known as “Yellow Pages”, what are still in use), but even since people started to utilize computers, a several generations of contact information management software was developed.

3.2.1 Personal approach

CIM software of the first generation looked like locally stored address books, intended for private use by their owners. Examples of this approach can be divided into three classes:

- address books incorporated into e-mail clients (*MS Outlook Express, Windows Mail, The Bat, Mozilla Thunderbird*, etc);
- personal organizers and address books *(Lotus Organizer, Kontact*, etc);
- tiny applications bundled with mobile phones.

Key features of this generation applications are:

- user needs to take care about data synchronization between devices;
- address books are stored off-line with no access to the data from outside of the device;
- data safety is the job of users, they should back-up data periodically.

Later, with spread of web-based e-mail applications, personal CIM also migrated to on-line environment. Personal address-books are stored at the service provides side while users access them via the Internet. Still, users can't share contact details with others. Manual data import-export or synchronization is needed, if customer utilizes some off-line communication device. Data back-ups became optional, as usually service provides take care about this. One example of personal on-line CIMs are *Google Mail*[25] and other web-based e-mail services. With the help of open-source software personal mailbox with basic on-line address book can be quite easily set-up on a typical web-server.

The key disadvantage of this kind of application was impossibility to quickly share data with family, partners and colleagues, what is especially important for business users.

25 http://gmail.com

3.2.2 Centralized approach

With spread of computer networks and Internet, CIM software designed for corporate usage quickly started to adopt benefits of technology to offset a disadvantages, inherent to personal address books.

Key feature of centralized approach is the shared database of all contact details about common parties (persons or organizations), maintained by a group of users.

Centralized CIM method (for both web-based and off-line type) solves most of the problems with contact details availability and improves the quality of data :

- if one user has more exact information about particular customer, it's enough to update his record in the central database and the changes will be available to the whole group in a moment;
- redundancy of contact details is eliminated, as the record about every particular customer is created only once;
- users should not worry about possible data loss or backups on local devices, as a fresh copy always available from the central contact database;

Depending on the scope of access to the stored data, the centralized CIM software can be implemented either as web-based systems or as stand-alone applications, working in local network only.

Centralized CIM software is usually integrated as a part of multifunction applications, widely known under common name *groupware* or *collaboration software.* Such type of software offers their users among other features contact management module in a form of shared address books, what may be accessed by any authorized user. There are plenty of *groupware* examples, like well-known communication software like *MS Exchange Server* (commercial software) or *Zimbra Collaboration Suit*[26] (free software). Various flavors of specialized applications, like CRM (out of the open-source products most memorable are v*Tiger*[27] and *SugarCRM*[28]) or advanced project management tools typically also integrate some kind of contact management subsystem and that's why it may be classified as an implementation of centralized CIM approach.

Still, there are two problems, both connected to the data quality, that centralized approach is unable to solve. First one appears if two users update the same attributes of the same record with different

26 http://www.zimbra.com/
27 http://www.vtiger.com/
28 http://www.sugarcrm.com/crm/

new values. Assume, that both users believe their data is valid. In this case a conflict of versions occurs, illustrated in the Table 5.

Table 5. Conflict while updating a centralized contact database.

1) Contact details, stored at central repository:	2) User „A“ updates them as follows:	3) User „B“ wishes to change the same record like this:	4) What data should be saved to the central repository?
Name: John Smith *Organization*: „Example Organization“ *Position*: president	*Name*: John Smith *Organization*: „Example Organization“ *Position*: **vice-president**	*Name*: John Smith *Organization*: „**Other Organization**“ *Position*: president	**?**

It is clear, that even the smartest software can't decide automatically, how to deal with this conflict, but should give to user a possibility to choose between two conflicting versions, to merge them (if possible) or to revert changes to the state before any changes will be made. In any case, data quality will be affected until both users who tried to update the record discuss the issue and choose a resolution. Usually to get the truth, a customer should be contacted and asked about his actual details. Needless to say, how complicated and expensive it could be, but sometimes it is just impossible. Even if collision between customer records were successfully resolved, there is no confidence about how accurate is stored data. Typically, record is updated only when customer lets to know about changes, or when he is queried about them.

The most efficient solution would be if customer could access his record and update it himself at any time. Will he do it soon after his contact details change? Is customer well-motivated to inform his partner about changes? To give affirmative answers to these questions, organization should invest into customer relationship, but without doubts may be considered, that obtaining actual contact details directly from customers it the right solution.

3.2.3 Distributed approach

As was shown above, despite of storing data in centralized database, in the most cases only the customer himself has his own contact details of the highest quality. Based on this consideration, the newest CIM approach can be defined: customer should maintain his machine-readable contact details record by himself. Data originated from the customer's public record should be considered as the highest quality data and therefore dominate over any other data of the same kind about this customer, that anyone else may have. If there is any other data about this customer, it may be used until customer provides this kind of data by himself (Table 6).

Table 6. Updating customer data in a distributed CIM.

1) Contact details, originated from customer himself	2) Customer's associates know from untrusted sources additional details about him	3) Customer's age considered to be valid until customer updates his data, providing the exact value of desired data type	4) Customer's associates have to update they data, calculating new age based on birthday provided by customer (data of the same type)
Name: John Smith *Organization*: „Example Organization“ *Position*: president	*Age*: **35**	*Name*: John Smith *Organization*: „Example Organization“ *Position*: president Birthday: 01.04.1970	*Age:* **39**

The distributed CIM approach actually expands the centralized one, assuming, that authoritative data about customer is stored outside of the centralized repository. Meanwhile, centralized repository is still necessary, but in the context of the distributed system, it stores the following:

- a working copy of data (for quick processing and backup purposes, in case original data becomes inaccessible);
- additional "unverified" data, what are not defined as original information obtained from customer's public record;
- all the data about customers, who don't have their business card available on-line, thus their contact details can't be automatically obtained.

To implement distributed approach to the contact management, a new data exchange infrastructure has to be established. As distributed CIM approach implies communication among a large number of participants, the main focus should be on the data exchange standard rather then application itself. In other words, – all business cards should be described in an agreed form, using common definitions and mark-up language. Then a method used to retrieve and process the published data should be specified. After that a development of the particular application could be started.

Currently there are no well-known either commercial or open-source software applications, based on distributed CIM approach. Some big Internet companies provide an Application Programming Interfaces (API), which equip developers with a set of tools for aggregating user details and thus suitable for building such kind of applications. The most notable efforts in distributed CIM are *Google Social Graph API*[29], *Google Contacts Data API*[30], *Yahoo! Address Book API*[31] and *Microsoft Live Contacts API*[32].

29 http://code.google.com/apis/socialgraph/
30 http://code.google.com/apis/contacts/
31 http://developer.yahoo.com/addressbook/
32 http://dev.live.com/contacts/

3.3 Case-Study: CIM Model for Organizations

Author admits, that it is impossible (at least in the context of current research) to define universal CIM model for abstract organization, that would be suitable for any organization over the World. Moreover, there is out of scope of current research. In no doubt, the proposed on-line business card exchange system should be global: it may be used in any country of the world, it should be able to deal with almost any kind of information, that represents organization and that's why may be put on its digital business card. Still, some vision about what data can be placed on virtual business card is needed to understand, which specification is the most suitable for representing them in format way.

3.3.1 On-line Business Card Structure of Typical Organization

As virtual business card compared to its real sister does not have very clear physical borders, on-line business cards may hold much more details about organization. Eventually, they can hold unlimited volumes of information and let the other party to decide, what information is important for him.

Currently, virtual business card in some form can be seen on “Contacts” page of the organization homepage (a single page, filled with contact details of organization and other useful information, that is controlled by organization authorities and is shared with everyone in the Internet. The other container for more structured “virtual business cards” is on-line catalogs of different kind. Most notable are catalogs of the following three types:

- “yellow pages” – a directory (published in the Web or on paper), where organizations are categorized according to industrial taxonomies (by supplied product groups or provided service types);
- “white pages” – the same as “yellow pages”, but organizations are listed alphabetically;
- national registers – the same as “white pages”, but directory is managed by government or local authority and holds compact legal information about organizations, officially registered in particular country or administrative area.

Author selected 20 catalogs (the list is available in Appendix A) from different regions starting from his homeland Estonia and up to the catalogs of Australia and New Zealand, analyzed them and selected a list of properties, most often used on virtual business cards of organizations. The results are presented in the Table 7.

Table 7. Mostly used organizational credentials and contact details.

Property	Cardinality	Occurrences	Data Type
Legal name	1	20	Text
Legal form	0 – 1	14	Text
Business name	0 — ∞ (*old name, brand name*)	16	Text
Registration code	0 – 1	16	Text
VAT number	0 – 1	11	Text
Logo	0 – 1	20	Bin/URI
Address	0 — ∞ (*legal, postal, filial*)	20	Text
Working hours	0 — ∞ (*for physical addresses*)	4	Text
Phone	0 — ∞ (*cellular, plain, toll-free, VoIP*)	20	Text
Contact hours	0 — ∞ (*for phone numbers*)	2	Text
IM account	0 — ∞ (ICQ, *Skype*, MSN)	5	Text
E-mail address	0 — ∞	20	URI
Homepage	0 – 1	20	URI
Type of industry	0 — ∞	19	Text
Year established	0 – 1	11	Date
Annual revenue	0 – 1	7	Num/Text
Number of employees	0 – 1	5	Num/Text
Contact Person	0 — ∞ (*chiefs of departments*)	18	Text
Plain text description	0 – 1	20	Text
Geo-coordinates	0 — ∞ (*for physical addresses*)	12	URI/Text
Parking options for customers	0 — ∞ (*for physical addresses*)	1	Text
Accepted payment methods	0 — ∞	7	Text
Bank account numbers	0 — ∞	9	Text
Spoken languages	0 — ∞ (*for phone numbers or IM accounts)*	6	Text
Discount programs	0 — ∞	2	Text

Cardinality column shows, if named property is mandatory in the catalog (value "1") or optional, that may be present only once (value "0 – 1") or more than once (0 — ∞). *Occurrences* column gives an idea about how many catalogs offers their customers possibility to describe company information with particular property.

Most of the properties are optional and may be omitted. The name of organization is the only mandatory, but in practice there are no companies described with only its name – it accompanied by at least one communicational contact (address, phone, e-mail, URL) or legal identity (registration ID or VAT code). Despite there were no catalogs, with all mentioned properties used at once, the structure of the virtual business card may become very complex, and hardly any of existing specification can introduce all mentioned properties as built-in.

At the same time, the whole data structure is redundant and may not be needed for every CRM or application, handling data about organizations. This means, that the key feature of a proper standard would be a combination of a variety of built-in properties with the extensibility (how easily specification can be extended with custom properties).

3.3.2 Sample Business Card of an Organization

While previous chapter gave an overview about the structure of the possible properties, a sample organization should be taken to project real data to the structure and see, how actually looks virtual business-card in a human-readable way, before it will be encoded with some CIM standard.

Author decided to take Tallinn University as a sample of organization what may need a virtual business card. The fact, that for demonstration purposes was picked-up an Estonian organization, doesn't mean, that a specification, which is most suitable for Estonian organizations, will be selected as the base for prototype development. Ideal specification will fit almost any organization and, as was mentioned above, should be extensible to make possible definition of additional properties and attributes, to describe organizations of certain domain (country, industry, etc).

The available information about Tallinn University was described using data structure defined in Table 7 and presented in the Table 8. This data will be used later as sample data for reviewing and comparing specifications.

Table 8. Virtual Business Card Data Structure of Tallinn University.

Property	Value	Data Type
Legal name	Tallinna Ülikool	Text
Legal form	Non-profit organization	Text
Business name	Tallinn University	Text
Registration code	74000122	Text
VAT number	EE100251335	Text
Address	Narva mnt 25, 10120 Tallinn, Eesti	Text

Table 8 (*continued*).Virtual Business Card Data Structure of Tallinn University.

Property	Value	Data type
Logo	TALLINNA ÜLIKOOL	Bin/URI
Working hours	8.00-20.00	Text
Phone	+372 640 9101	Text
Contact hours	9.00-17.00	Text
E-mail address	tlu@tlu.ee	URI
Homepage	http://www.tlu.ee	URI
Type of industry	Education	Text
Year established	1952	Date
Number of employees	100-500	Num/Text
Parking options for customers	Paid parking area	Text
Accepted payment methods	Bank transfer	Text
Bank account	EE071010002006943007, SEB Pank	Text

3.3.3 Requirements for CIM Standard for On-line Business Cards

As an outcome of completed case-study, author defines the following ten aspects as important requirements for specification to become a standard for representing on-line business cards of organizations. This requirements are divided into two groups: critical requirements, that should be fulfilled to qualify specification as a standard for prototype development, and general recommendations.

Critical Requirements (order does not make sense):

1. open-source community support – indicates how specification is adopted by open-source community; the amount of available open-source projects, what use this particular specification for storing or exchanging data (import/export);
   or (at least one of them – either first one or next one – must be fulfilled)
2. business community support – indicates the adoption of specification by business community; the amount of vendors using this specification in their proprietary software; what businesses support the development of the specification;
3. compliance with Semantic Web standards – how closely specification follows the Semantic Web model (Figure 2);
4. extensibility – how easy (with less maintenance and agreements) developers can expand current specification with their own properties and attributes; how other developers can be

informed about changes (ideal standard is self-documenting standard, without need to store new semantics outside of extended specification);

5. level of potential – how long will be (*i*f can be predicted) the life-time of this standard? Does it have prospects or specification is futureless? Will it last for years (like *HTTP*[33]) or may be easily replaced by new trends (like *Gopher*[34] protocol)?
6. variety of built-in properties for description of organizational business cards – how suitable is particular specification without any significant modifications for describing organization business card, its contact details and other properties?

General recommendations:

1. maturity – how stable specification is, are there any global changes in active development, that may affect all previously developed software?
2. tractability – how complicated for an average developer, who is novice to the concept of on-line business cards, to adopt specification in his application?
3. support in developer tools – does most-popular web-development frameworks and programming languages (*Java*, C#, PHP, *Ruby*, *Python*) have a built-in tools to process data with respect of mentioned specification?
4. documentation quality – is specification well-structured and has exhaustive manuals; how informative is web-site of the standard; are there any other information sources about specification and it adoption (chats, forums, blogs)?

All named requirements and recommendations will be used later in this chapter for evaluation of short-listed specifications.

3.4 Overview of Existing Standards

To proceed with further exploration of distributed contact information management approach, an overview of existing machine-readable standards, suitable for representation of contact information and identity credentials (in other words, business-cards) should be introduced. Author has selected three open standards what have a history and significant community support: *vCard, Friend of a Friend (FOAF) and OASIS Customer Information Quality (CIQ)*. There are other standards concerning provision of contact information (like *eBiquity Contact Information Ontology*[35]*, SWAP*

33 http://en.wikipedia.org/wiki/HTTP
34 http://en.wikipedia.org/wiki/Gopher_(protocol)
35 http://ebiquity.umbc.edu/ontology/

Personal Information Markup[36] *or Pervasive Computing Standard Person Ontology*[37]) and perhaps even more, but they will be not discussed in this paper, because of their proprietary nature, lack of documentation or minor influence on the Internet space.

3.4.1 vCard

vCard format is the oldest open standard for business cards representation, being known since 1995. Responsibility for development and promotion of vCard standard lays on I*nternet Mail Consortium*[38] (IMC) – an international organization of vendors focused on expanding the role of the Internet mail and making it easier for Internet users to get the most out of this growing communications medium.

Today vCard is de-facto standard for exchanging contact details between many of devices and applications: files in vCard format (with .vcf extension) most often can be seen attached to e-mails or sent from one cellphone to other with SMS or *Bluetooth*. vCard has wide support of computer software of any kind, first of all in e-mail applications. The latest stable version of standard is 3.0 published in 1998. (Dawson & Howes, 1998).

Listing 3. Business Card of Tallinn University in vCard 3.0 format

```
BEGIN:VCARD
VERSION:3.0
FN:Tallinna Ülikool
ORG:Tallinna Ülikool
TEL:+3726409101
ADR:;;Narva mnt 25;Tallinn;Harjumaa;10120;ESTONIA
EMAIL:tlu@tlu.ee
URL:http://www.tlu.ee
LOGO:VALUE=uri:http://www.tlu.ee/files/design/96/logo.7593a328bcc259be7f
5400a65fca0765.gif
CATEGORIES:Education
UID:74000122
BDAY:1953-01-01
X-PARKING:Paid parking area
X-LEGAL:Non-profit organization
X-IBAN: EE071010002006943007, SEB Pank
X-VAT:EE100251335
X-BRANDNAME: Tallinn University
X-WORKINGHOURS: 8.00-20.00
END:VCARD
```

Listing 3 presents business card of Tallinn University encoded according to vCard specification. As can be seen, specification defines business card in a plain-text format with proprietary syntax and

36 http://www.w3.org/2000/10/swap/pim/contact
37 http://pervasive.semanticweb.org/ont/2004/01/person
38 http://www.imc.org

semantics. A special custom text parser should be applied to the document to get exact field values.

The most notable derivatives of vCard standard are hCard microformat, what is 1:1 representation of vCard 3.0 properties and values published in (X)HTML (and thus simultaneously visible and accessible by both people and machines), and *Portable Contacts*[39] – an easy-to-implement, vendor-neutral API to enable CIM for any website that stores address books of its users. *Portable Contacts* is built on top of existing open standards (vCard as data model, *JavaScript Object Notation*[40] – JSON – as data interchange format, OAuth[41] as authentication method, and XRDS-Simple[42] as metadata discovery technology). *Portable Contacts* are used by *Google Data Contacts API, Plaxo*[43] social service and others.

vCard and the Semantic Web

Currently the existing vCard 3.0 standard has a little official support of Semantic Web essential features. RFC2426 prefers UTF-8 for character encoding of vCard data, but any other character set may be also specified. The use of URIs is limited to pointing at vCard source location in the Web and linking to binary data, like photo or logo. There also were numerous of XML and RDF representations of vCard data created by enthusiasts and discussed in W3C (Iannella, 2001), but no one of them were recommended by W3C or any other organization as valid representation.

To provide valid XML and RDF encoding for vCard format, the work on the next version of the standard (vCard 4.0) was started by vCardDAV[44] working group. As for spring 2009, working group delivered a draft format specification for the new vCard version, which will obsolete RFC 2425, 2426 and 4770 if approved (Perreault & Resnick, 2008). This new format assumes, that vCard data will be still represented in plain text format, but with support of XML encoding in NG Relax scheme syntax (Perreault, 2009). vCardDAV group does not provide a RDF ontology for vCard object yet, thus latest available RDF scheme is Harry Halpin, Brian Suda and Norman Walsh initiative (2006) to use RDF best practices while modeling a subset of vCards.

Summarizing aforesaid, there is confidence that vCard standard is upcoming for the Semantic Web.

vCard Data Model

All major contact details can be described in vCard format. According to the version 4.0 draft,

39 http://portablecontacts.net/
40 http://www.json.org/
41 http://oauth.net/
42 http://xrds-simple.net/
43 http://www.plaxo.com/
44 http://www.vcarddav.org/

vCard features built-in support for personal name and nickname, photo, date and place of birth and the same for death, gender of the person, his addressing and communicational properties (phone numbers, IM accounts, e-mail and ground mail addresses) and geo-location.

The data model of vCard can be extended with the use of so-called "*X-properties*" (property names starting with "X-" characters) if both exchanging parts agree on their names and possible values. It is no possible to define the exact list of *X-properties* within the specification, so developers must solve this issue separately, providing external documentation for all extensions they agreed to use.

vCard for Organizations

As noted in RFC2426, vCard standard was created first of all to exchange information about private persons (Dawson & Howes, 1998), that's why until version 4.0 it lacks possibility to distinguish organizational business cards from personal ones. This issue was corrected in the latest 4.0 drafts, when a `KIND` property (with possible values "`individual`", "`org`", "`group`" or "`location`") was introduced (Perreault & Resnick, 2008).

Despite of this, there are still very few predefined properties to describe organizational business card. Besides those properties, what is in common both for individuals and organizations (like addressing and communication), the additional properties available for the organization are limited to logo and business category only. If organization would like to share any other data, for example, working days and hours (important information for museums, banks, etc), what is not defined by vCard standard, the X-properties should be used. This approach makes impossible any interoperability of these properties, unless both exact syntax and semantics of X-properties agreed among all parties who will exchange this data.

3.4.2 Friend of a Friend (FOAF)

Next contact information exchange standard was developed in mid-2000 with the expansion of social networks. The aim of FOAF is to describe people and their relationships in a machine-readable way by a set of numerous aspects, like location, interests, membership in organizations and participation in projects, schools and publications, etc. The contact details and personal credentials are added to the personal profile also. (Brickley & Miller, 2007).

The main advantage of FOAF is ability to refer to people someone knows, works or studies with and to build one's own decentralized social network and to get actual information about their activities, new friends (or even friends of friends, as standard names stands), changes in contact details and so on. This is done by linking their FOAF-files or pointing to any other Internet-resources.

FOAF initiative got a wide support from IT industry organizations, like *Google*, who implemented proposed mark-up in its *Google Social Graph API*, *IBM* who published promoting articles[45], *Plaxo* social service who included FOAF support to its service and others.

Listing 4 demonstrates, that FOAF document is a XML/RDF document based on Semantic Web principles.

Listing 4. Business Card of Tallinn University in FOAF format.

```
<rdf:RDF xmlns:foaf="http://xmlns.com/foaf/0.1/"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:loc="http://simile.mit.edu/2005/05/ontologies/location#">
   <foaf:Organization rdf:about="#TLU">
     <foaf:name>Tallinna Ülikool</foaf:name>
     <foaf:mbox rdf:resource="mailto:tlu@tlu.ee" />
     <foaf:homepage rdf:resource="http://www.tlu.ee" />
     <foaf:logo rdf:resource =
      "http://www.tlu.ee/files/design/96/logo.7593a328bcc259be7f5400a65f
ca0765.gif" />
     <loc:address>Narva mnt 25</loc:address>
     <loc:city>Tallinn</loc:city>
     <loc:postal-code>10120</loc:postal-code>
     <loc:country>Eesti</loc:country>
   </foaf:Organization>
</rdf:RDF>
```

FOAF and the Semantic Web

FOAF standard was designed from scratch to be used in the Semantic Web, thus its output document is Unicode-encoded, written in XML syntax and adopts conventions of RDF. FOAF-documents link to other resources by means of URIs.

Moreover, FOAF with help of Web Ontology Language (OWL) defines its own vocabulary (ontologies) that may be re-used elsewhere in any other Semantic Web applications. The FOAF vocabulary is pretty simple, pragmatic and designed to allow simultaneous deployment and extension. FOAF is intended for broad use, but its authors make no commitments regarding its suitability for any particular purpose. (Brickley & Miller, 2007)

45 http://www.ibm.com/developerworks/xml/library/x-foaf.html and http://www.ibm.com/developerworks/xml/library/x-foaf.html

FOAF Data Model

FOAF offers users a standard attributes for description of mostly electronic identity. User can describe his name (in various formats), e-mail and homepage addresses, URLs of workplace or school, numerous IM accounts (ICQ, MSN, *Jabber, Yahoo*) as well as accounts for on-line gaming, e-commerce or chat. There are also quite unusal attributes for description user's personality, like Myers Briggs taxonomies[46] or *Geekcodes*[47].

On the other hand, FOAF default vocabulary lacks of properties for description off-line details, like physical address or phone numbers. All absent properties may be added to the FOAF document using either existing RDF ontologies or defined by user himself. In the Listing 7 author extended FOAF basic properties with an ontology, describing physical address (defined in `<loc:>` namespace). Offered approach is rather complex, thus who wish to expand FOAF, should start with learning RDF/OWL, then determine, if any of already existing ontologies can be applied or new vocabulary should be defined for specific purposes. This task seems to be very complicated, slow and resource-consuming approach even for experienced developer, as needs analytic skills and system thinking in a domain context.

FOAF for Organizations

At a glance, FOAF vocabulary "as is" is suitable for representing contact details of only these "virtual" organizations, that does not need to share their physical contact details (like phone numbers and post addresses). To get a support of such essential elements for most organizations, FOAF should be significantly expanded by adding new vocabularies, what is, like mentioned above, rather complex task.

One outstanding experiment to extend FOAF to represent in more detail the structure and relations between organizations is *FOAFCorp*[48] effort. This ontology is build on top of FOAF to describe in more detail the structure and relationships of organizations. *FOAFCorp* is current the most popular extension of FOAF, but it is still focused on projects, people, their role and position in the company (etc.), but not on a excessive representation of contact details.

3.4.3 Customer Information Quality (CIQ)

The brief analysis of two major Internet standards for exchanging contact details demonstrated a

46 http://en.wikipedia.org/wiki/Myers_Briggs
47 http://www.geekcode.com/geek.html
48 http://rdfweb.org/foafcorp/intro.html

lack of global application-, vendor-, industry- and technology-agnostic open standard, what is focused on providing an excessive data model for contact information exchange and what is suitable for both private persons and organizations. An effort to fill up the niche was made by *The Organization for the Advancement of Structured Information Standards*[49] (OASIS) late in 2000, when *Technical Committee (TC) for Customer Information Quality (CIQ)* was established. The word "Customer" in the name of the committee has a much broader meaning: it used as a common name and can represent any party such as: citizen, client, contact, employee, organization, partner, person, provider, subscriber, supplier (etc).

The goal of CIQ TC is to develop generic set of XML specifications for defining, representing, interoperating and managing party-centric information, that can be used in global context – in any of more then 200 world countries, independent of their particular culture, language or geopolitical aspects. The standards should be applicable to any industry (government, e-commerce, logistics, manufacturing, services and other), any applications (CRM, Human Resources, sales support, billing, etc.) with scalable level of details, starting from abstract (human-readable text format) to well-structured (machine-readable). (OASIS, n.d.)

To achieve the goal, OASIS TC divided the customer information domain into four entities and implemented each entity as an extensible mark-up language, based on XML-scheme:

- names (*extensible Name Language*, xNL) – a standard for representation party names, based on about 30 possible name formats;
- addresses (*extensible Address Language*, xAL) – a standard for representation party physical addresses and geographical locations, based on about 130 possible address formats;
- party Information (*extensible Party Information Language*, xPIL) – a standard for representation party-centric credentials, based on attributes, collected from more then 200 countries;
- party relationships (*extensible Party Relationships Language*, xPRL) – a standard for representation affiliation between any two parties.

(CitCity.ru, n.d.)

The latest version of CIQ standard is 3.0, released in November 2008, includes only three schemes mentioned above. The xPRL scheme was not approved by TC[50] and there is no information about how long does it take to release the final component.

49 http://www.oasis-open.org
50 True on May 1, 2009, updates check at http://www.oasis-open.org/committees/ciq/

CIQ is widely used across the globe in almost any sector: government, finance, health, automotive (etc), but mostly in proprietary environment. The few open standards using CIQ are, for example, *Google Maps*[51] and *Google Earth*[52] and open-source *HR-CRM*[53]. (OASIS, n.d.)

Listing 5. Data about Tallinn University encoded with CIQ-specification.

```
<p:OrganisationDetails>
  <n:OrganisationName>
    <n:NameElement n:ElementType="FullName">Tallinna Ülikool
    </n:NameElement>
  </n:OrganisationName>
  <p:Addresses>
    <p:Address a:Usage="Postal">
     <a:FreeTextAddress>
      <a:AddressLine>Narva mnt 25, 10120 Tallinn, Eesti</a:AddressLine>
     <a:FreeTextAddress>
    </p:Address>
  </p:Addresses>
  <p:ContactNumbers>
  <p:ContactNumber p:CommunicationMediaType="Telephone"
   p:ContactHours="9.00-17.00">
    <p:ContactNumberElement p:Type="LocalNumber">640 9101
    </p:ContactNumberElement>
    <p:ContactNumberElement p:Type="CountryCode">372
    </p:ContactNumberElement>
   </p:ContactNumber>
  </p:ContactNumbers>
  <p:ElectronicAddressIdentifiers>
    <p:ElectronicAddressIdentifier p:Type="URL">http://www.tlu.ee
    </p:ElectronicAddressIdentifier>
    <p:ElectronicAddressIdentifier p:Type="EMAIL">tlu@tlu.ee
    </p:ElectronicAddressIdentifier>
  </p:ElectronicAddressIdentifiers>
  <p:OrganisationInfo p:OperatingHourStartTime="8:00"
   p:OperatingHourEndTime="20:00"/>
</p:OrganisationDetails>
```

Listing 5 shows, that typical CIQ-encoded data is XML-based document, where all elements have a prefix for scheme identification. In this particular example `<n:>` prefix stands for xNL scheme and `<p:>` for xPIL scheme.

CIQ and the Semantic Web

Customer Information Quality set of standards utilizes only few components of the Semantic Web model: Unicode, URI and XML (with namespaces and schemes). Currently CIQ 3.0 has no support of RDF, as the level of complexity of schemes, the variety of attributes and properties described in specifications put obstacles on the way of defining ontologies. (D'Arcus, 2004). As CIQ standard is

51 http://maps.google.com/
52 http://earth.google.com/
53 http://hr-crm.sourceforge.net/

based on XML, there is no problem to expand it with RDF in the future.

CIQ Data Model

Customer Information Quality set of standards offers developers a great number of predefined attributes and properties for describing their data in almost any possible way used in the World. As already mentioned above, names can be expressed with more then 30 different formats, addresses in more then 130 formats and party information itself may be described using about 30 predefined classes (for example, Visas, Vehicles, Occupations, Membership, Hobbies, Events, Documents) accompanied with a set of properties for describing electronic means of communications and so on. These classes have nested properties and in total form a comprehensive yet flexible structure.

The key features of CIQ data model are its extensibility and a selection of special attributes, aimed to measure data quality.

As the standard element names say, CIQ is extensible specification. Every user may extend default lists of types, that are offered by specification, with own types. To make it possible, each specification is divided into two separate XML-schemes. One is used for defining common (or parent) classes and properties. It is not supposed to be changed by developer, as specification may be updated by the next version of CIQ standard. The other scheme is designed to store application-specific (nested) properties and their attributes. They have exact meaning in the context of this application and therefore should be distributed among all developers, who are also using extended type's lists.

For example, in the most western cultures a person is identified by his full name, composed of first- and last name and, sometimes, middle name. By default, CIQ xNL offers nine predefined properties, what may be used for formal representation of personal name: preceding title (His Excellency, Honorable, Dear), title (Mr, Mrs, Ms), first name, middle name, last name, other name, alias (nickname), generation identifier (Junior, Senior, The Second, III) and academic degree (BSc, MA, PhD). But if CIQ is used in Island, where the full name is composed of the first name and a *patronymic* – a father's (or *matronymic* – a mother's) name, thus a family has no common name[54], it would be incorrect to force Icelanders to fill-in their patronymics as last name (because some Icelanders may have a classic last name also).

54 http://en.wikipedia.org/wiki/Icelandic_name

Listing 6. New parameter definition in CIQ specification.

```
<xs:simpleType name="PersonNameElementList">
  <xs:restriction base="xs:normalizedString">
    <xs:enumeration value="Patronymic">
      <xs:annotation>
        <xs:documentation xml:lang="en">Father's name</xs:documentation>
        <xs:documentation xml:lang="et">Isanimi</xs:documentation>
        <xs:documentation xml:lang="ru">Отчество</xs:documentation>
      </xs:annotation>
  </xs:restriction>
</xs:simpleType>
```

CIQ xNL offers two possible solutions for this problem: developers may use "`other name`" property for patronymics and ignore the semantics. More advanced developers may define in addition to existing properties a new one – called "`patronymic`" (Listing 6). In this case, it can be, first of all, properly translated if application is being localized, and, secondly, may be re-used in other cultures, who use patronymics – Slavic, Arabian and a few others. New property gives a meaning to the data and saves "`other name`" property for it intended uses.

The same way almost any element in all four extensible languages may be expanded with user-defined options.

The second key benefit of CIQ standard in data quality attributes, what may be applied to any element or piece of data. Currently, CIQ 3.0 defines three attributes to measure quality of data:

- `DataQualityType` indicates the level of reliability of the data. Attributes has two possible values: "Valid" (means that data was validated and is considered to be true and correct) and "Invalid" (Indicates that at least some part of the content is known to be incorrect). Attribute can be used for example to exchange data, what is known in advance to be false, or to make difference between data obtained from reliable source ;
- `ValidFrom` indicates the date and time data is considered to be valid from. Attribute can be used if piece of data it is connected to has a life-time or must be used only after the indicated date. This is useful, for example, for making exceptions in working hours, to define in advance new last name for a woman after a marriage, to inform about address change during relocation, etc.
- `ValidTo`, like name says, indicates the date and time data is considered to be valid until. The usage scope of the attribute is in general the same like in the previous case. (OASIS, 2008).

CIQ for Organizations

Contact Information Quality standard was developed with respect of many different party types, but two main classes are `Person` (or physical body, depending on the context may be referred as Citizen, Customer, Friend, Member or any other possible role) and `Organization` (or legal body – an enterprise, a club, a non-profit association in any role – Consumer, Supplier, Customer, Partner, etc).

Organization properties are separated in xNL and xPIL specifications. Specification for names (xNL) defines the following properties for describing organization names:

- organization name, what can be of different types (legal name, former name, unofficial name – customizable list);
- organization name element, what can be of different kind (full name, name only, legal form only – customizable list);
- subdivision, what can also be of selected type (department, division, branch, section – customizable list).

(OASIS, 2008)

There is no cardinality set how many elements can be use for entity description. As well, common XML attributes, like ID or language, can be included to uniquely represent specific organization with desired localizations.

Party Information specification (xPIL) introduced even more broad choice of elements supposed for decomposing organizational data. The main element of xPIL in organizational context is `<OrganizationDetails>` container, used to all the characteristics of specifying organization. It can hold organization name (with its own structure mentioned above), addresses, bank accounts, contact numbers, documents, electronic identifiers, events, legal identifiers, membership, revenues, stocks, vehicles and miscellaneous organizational info (working hours, amount of employees, etc).

3.5 In the Search of the Advanced Standard

To proceed further with prototype development, shortlisted standards should be compared and only one selected as a basement for on-line business card exchange system.

3.5.1 Comparison of Specifications

While comparing specifications, author used for evaluation his experience as a software developer and knowledge obtained during university master's study. The results are shown in the Table 9, the order of the aspects has no significant meaning. More detailed information about requirements are given in chapter 3.3.3.

Author used for evaluation three ranks with the following meanings:

- high – specification completely satisfied defined requirement;
- medium – specification mostly (with rare exceptions) satisfies requirement; minor improvements should be done before it could get the highest mark; improvement are possible and does not contradict ideology of specification; this rank considered to be positive;
- poor – the only negative rank; it says, that specification has a major drawbacks and these weak points can not be easily improved.

Color-coding is used (high marks are green, poor marks – red) to distinguish outstanding results.

Table 9. Comparison of Analyzed CIM Specifications.

Aspect	vCard 3.0	FOAF	CIQ 3.0
Critical requirements			
open-source community support	high	high	poor
business community support	high	medium	high
level of potential	high	high	high
compliance with Semantic Web standards	medium	high	medium
extensibility	poor	medium	high
variety of built-in properties	poor	poor	high
Recommendations			
support in developer tools	medium	high	medium
documentation quality	medium	medium	high
maturity	high	medium	high
tractability	poor	high	medium

As seen from the table, two standards – FOAF and CIQ – got mostly positive marks and both only one negative in the requirements section. Outsider is the vCard standard, as the current plain-text version (defined in 1998, as was mentioned before) with “hard-coded” properties can not compete with modern specification, created special for the Semantic Web epoch – this specification got two negatives for requirements.

In the recommendation section leaders (FOAF and CIQ) got both two high ranks and two medium ranks, while vCard was “honored” with one negative mark.

3.5.2 Selecting a Standard

The results of comparison shown in Table 9 are quite self-explanatory, especially in the context of their analysis done before, but still they should be explained in more verbose manner. Table 10 summarizes *pro et contra* for each standard, what helps to choose between leaders of comparison table – FOAF and CIQ.

Table 10. Advantages and Disadvantages of Analyzed CIM Specifications.

Standard	Pro	Contra
vCard 3.0	• very good support from both open-source and business communities; • mature standard with a high level of potential, as will be updated to the version 4.0 that should get official Semantic Web support.	• outdated specification that would be replaced by new standard, but not earlier than in autumn 2009; • limited extensibility: added properties should be agreed outside of the specification; • a few built-in properties for organizational business-cards; • data interchange needs text-file parsing: lack of standard tools for this task; • no standard compliance with Semantic Web model, as all existing XML and RDF representations are unofficial.
FOAF	• very good support from open-source community, significant support from business community; • true support of Semantic Web model, thus well-supported in developers tools; • standard can be easily integrated into custom application, as it has simple structure and based on XML.	• Developing specification that may be updated, – the compatibility risk for commercial deployment; • lacks of built-in properties for representing contact- and organizational details; • defining new properties is rather complicated, as needs understanding of RDF approach;
CIQ 3.0	• very strong support from business community and OASIS organization, what is in charge of standard development and adoption; • enough support of Semantic Web model to utilize benefits of XML processing tools; • mature standard (current version approved in 2008) with no information without upcoming global changes; • has almost all needed properties built-in to define contact details and various organization information details. • Specification has a great potential due to good documentation and logical structure;	• There is a few open-source projects using this specification, – time is needed to adopt this standard by open-source community; • developers should spend much more time (compared to other standards) on studying the specification, as standard is verbose and not very clear for newbies (particularly, this limits spread in open-source community).

Based on conducted analysis, author selects *OASIS Customer Information Quality 3.0* specification as a basement for further prototype development.

Counterarguments for CIQ disadvantages mentioned in Table 10 are:

- current research will make its contribution into popularization of CIQ specification among open-source developers, as prototype, demonstrating the power of CIQ will be freely available. On the other hand, since the approval of 3.0 standard passed only six months, what is not enough for open-source community to adopt it and deliver new projects using this specification;
- complexity of CIQ standard to some extent caused by lack of open-source applications based on this specification. Author is sure, that when there will be more examples of successful usage of CIQ in open-source projects, the community will help newbies to adopt the standards without need to thoroughly study the volumes of documentation, but will get quick and competent help from other sources, like forums, blogs, newsgroups or chat. On the other hand, the existing documentation is the best in its class with lots of examples and manuals, so with some readiness, there is no big problem to an average developer understand the basics of CIQ and to integrate it to own application. This will be proved with prototype development.

The selection of CIQ doesn't mean that distributed system for on-line business cards exchange can not be implemented with any other standard. Quite the contrary: it can be implemented, as other shortlisted standards (FOAF and vCard) are extensible in principle. Still, as both the analysis and the comparison proof, the CIQ specification is the most efficient standard for “painless” development of extensible on-line business card exchange system at a desired altitude.

3.6 Conclusion from Chapter 3

Key points derived by author from the above exploration of contact management approaches and specifications of the contact details exchange standards are:

1. The distributed approach to the contact management allows users to have their contact details available on-line and share them with partners. In a distributed system, users manage their data by themselves, what grants, that they record is always up-to-date. This method may be combined with storing some information about customer in local centralized database, while the other part is obtained directly from the customer.

2. Internet catalogs describe organization's record with about 30 properties (contact details,

credentials and other information useful for their partners). This means, that desired specification for the on-line business card should support as much of them as possible, or to be easily extensible by software developers.

3. Besides the amount of by default supported properties and their extensibility, there are other requirements for specification: support from either open-source or business community, compliance with Semantic Web standards and general level of standard's potential. Maturity, tractability, good support in developer's tools and high quality of the documentation are the main recommendations for the searched specification.

4. There are three standards, that have a significant spread in today's Internet space: vCard, Friend of a Friend (FOAF) and OASIS Customer Information Quality (CIQ). These standards do not have much in common, as currently they are used in quite different industry segments. Still, only one of them – OASIS CIQ – is suitable for correct representation of both personal and organizational information. Moreover, CIQ specification may be quite easily extended with additional attributes, has enough support of the Semantic Web model, contains special attributes to control data validity and, besides representation of customer information, can also represent relationship between parties. The OASIS CIQ specification was selected as a standard, to be used in the prototype development.

Chapter 4
Prototype Development

As the name says, this chapter focuses on the development of the prototype of distributed contact management application. Prototype consists of who modules – business card generator and software agent for reading business card from the customer's web-site. Both modules should be implemented to start the evaluation of the proposed system on end-users. The exact details of implementation (programming and coding stages) are out of scope of this chapter, as main objective is to provide an example of distributed contact management application, in other words, *to show how does system work*, rather then to teach how to build-up such system using specific programming language. Prototype development consists of three phases: design, implementation and self-evaluation of the prototype by its author himself.

4.1 Development Cycle

Software development is not a solid process, but a set of activities, performed by a developer in a certain sequence. Over the time, there were created a variety of frameworks, that consolidate the best practices and knowledge about how these activities should be performed. These frameworks are known as software development methodologies and typically consist of the following stages:

- Initial investigation – gives a high-level view about entire application, it objectives and their feasibility. In terms of current work this was done in chapters 1 and 2;
- System Design – Requirements Analysis. It describes features of the application, starting from the logical model of application behavior and finishing with particular properties of each module. The overall requirements for the distributed contact management system was discussed in chapter 3 of this paper, while the user requirements for the prototyped modules will be discussed in this chapter.
- Implementation – during this stage an application gets its shape: program code and user

interface are created. This paper described the environment where application is created and evaluated and lists the developer's tools used by author in this phase.

- Evaluation – typically, this is the final stage of the development cycle. It starts with software testing for possible errors and correspondence to the requirements, then continues with deployment, and, finally, end-users get access to the applications and may report a feedback about their experience. This stage is the final point of current research.
- Operation – the last stage, perhaps it is not connected to the development cycle as straight as all others are, but it is still important and may not be ignored, as software needs maintenance and users need support.

(Morris, 2005)

A prototype being developed is not a complex computer program, that's why author does not select any traditional software development methodology, but simply keeps on iterative model. When end-user will get access to the prototype, their feedback may affect the initial system design or cause modifications in the prototype. In this case, Design-, Implementation- and Evaluation- stages will be repeated and end-users will get an updated prototype for the next evaluation. The next three chapters describe the main points of the every of these three stages.

4.2 Design Phase

The idea of the prototype is to demonstrate the proposed system in action, thus giving to other developers a working example of that, how distributed contact management approach may be implemented in their applications. To make it possible, a technical description of the proposed system architecture should be given. The simplest way to do it is providing a graphical model of the prototype and its modules with brief comments about their functionality.

Author selected an *Agile Modeling* (AM) approach to the system design. AM is a practice-based methodology for effective modeling of software-based systems. AM offers a developer a collection of principles and best practices for modeling software that can be applied on a software development project in an effective and light-weight manner. (Ambler, 2007a). This is especially useful for rapid prototyping, when the application is being developed without gathering requirements from real users, but based on pure theoretical investigation of the problem. Lightweight approach helps developer saves resources, as the probability to iterate through the design phase after prototype evaluation stage is rather high.

The key principles of the AM used in the design phase of the current research are:

- Simplicity – the simplest solution is the best solution. No need to overbuild an application, as it may be refactored later when user requirements evolve. Models should be kept as simple as possible.
- Working software is the primary goal; the goal of software development cycle is to produce working software that solves desired problem in an effective manner. The primary goal is not to produce loads of documentation, management artifacts, or even models. That's why any activity, which does not directly contribute to the primary goal, should be avoided.
- Content is more important than representation; any model could have several ways to represent it (sketch on paper or a whiteboard, a "traditional" model built using special prototyping tools, etc). Developers should select the representation, that is sufficiently accurate and sufficiently detailed for understanding the meaning behind it, regardless the tools used for it production.

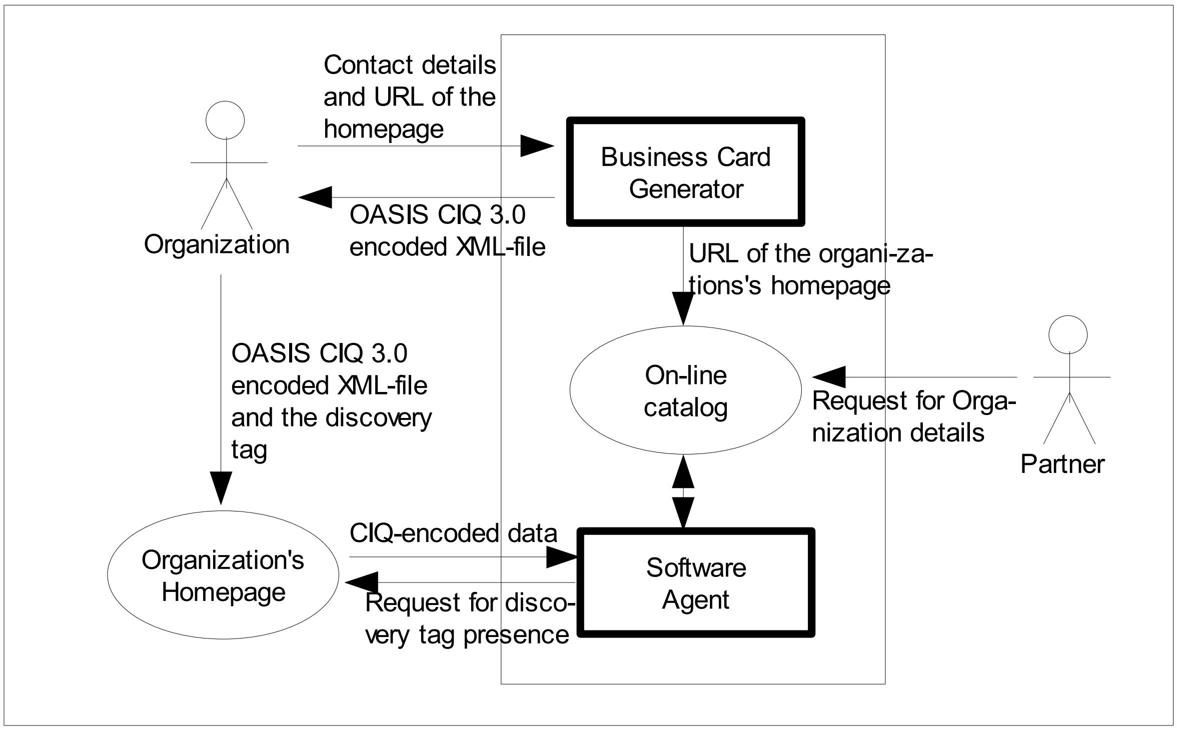
The named principles may be considered valid, as they have been adopted from *eXtreme Programming* (XP) methodology, which in turn adopted them from common software engineering techniques. (Ambler, 2007b).

4.2.1 Workflow of the Prototype

Prototype workflow is depicted on the Figure 4. This diagram extends in a free form use case diagram of the prototype (Figure 1). Detailed workflow diagram introduces a new component inside the system boundary: "On-line catalog". This catalog simulates actual CRM application (or any other application on the Partner's side), that stores data about organizations (and other customers).

In the terms of Prototype, on-line catalog works as public catalog of the "white pages", it is an address book, which stores names and contact details about participants. Similar catalogs were analyzed in chapter 3.3.1 with one difference: those catalogs used centralized approach to the contact management. White pages catalog of the prototype introduces distributed approach, as it stores only URLs of the homepages of participants, who decided to evaluate the prototype. The URL of the homepage is enough for Software Agent to check if CIQ-encoded business card is available and read contact details from it. The remote data are read every time any visitor addresses to them and data are not cached in the catalog (except of the URL).

Figure 4. Workflow diagram of the distributed contact management system prototype.


Contact details and URL of the homepage
Business Card Generator
OASIS CIQ 3.0 encoded XML-file
Organization
URL of the organi-za-tions's homepage
OASIS CIQ 3.0 encoded XML-file and the discovery tag
On-line catalog
Request for Orga-nization details
Partner
CIQ-encoded data
Organization's Homepage
Software Agent
Request for disco-very tag presence

The URLs are stored in an XML-file with the structure shown in Listing 7.

Listing 7. Structure of the on-line catalog XML-database.

```
<?xml version='1.0'?>
<participants>
  <url id='1'>http://www.example.org</url>
  <url id='2'>http://www.example.com</url>
  <url id='3'>http://www.tlu.ee</url>
</participants>
```

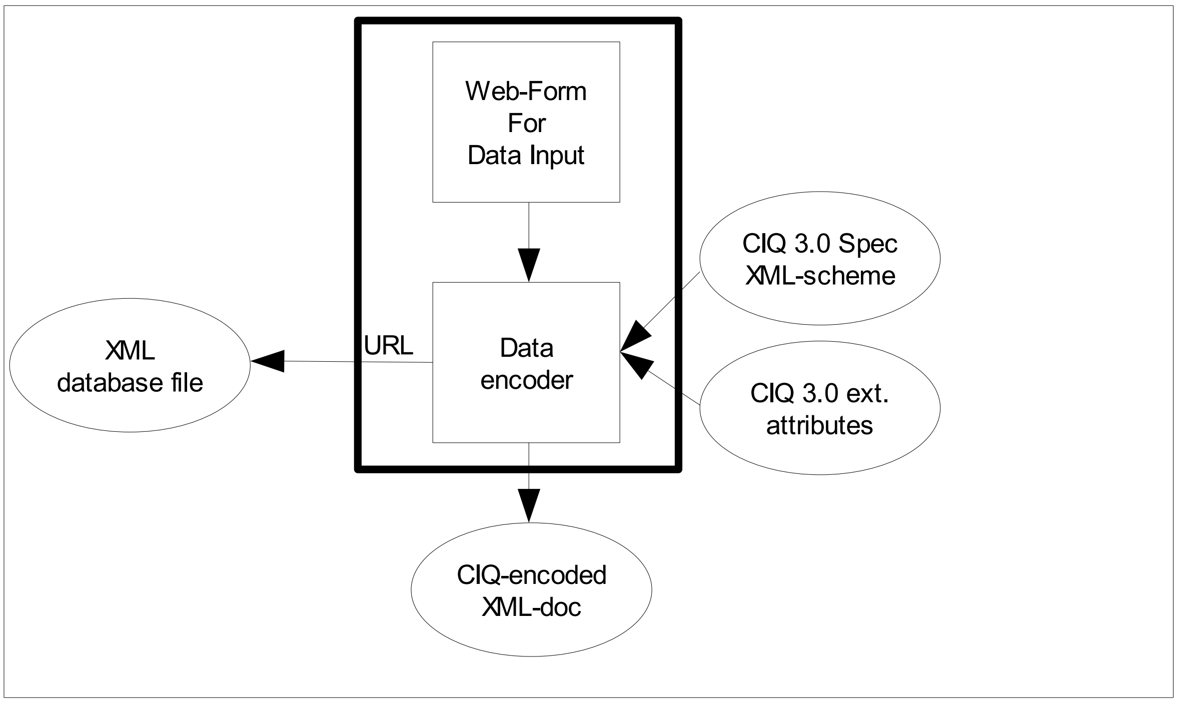
Despite the higher level of details, two major components of the prototype remain on the diagram as a “black boxes”. They should be decomposed in order to provide developers with exhaustive details about the functionality of the prototype.

4.2.2 On-line Business Card Generator

The workflow diagram of the Generator module is depicted in the Figure 5. The main component of the Generator module is Data Encoder. The Encoder gets data from the user input through the HTML web-form. The fields of the form correspond to the contact data model of the organization (see chapter 3.3). In the real-life application, derived from the prototype, fields represent the actual properties of the data being exchanged. For the purpose of the prototype application, author chose a list of properties, most often used in the analyzed Internet catalogs (see chapter 3.3.1):

- organization name (consist of legal name and legal type),
- physical location (consists of country, administrative- and sub-administrative areas, locality, premises and a postcode),
- contact numbers (for phone, fax and cellular phone, including country code and contact hours),
- electronic address identifiers (e-mail and a homepage),
- legal identifiers (VAT code and Registration ID),
- bank accounts (consists of type, account number and name of the bank),
- other organization details (working hours).

Figure 5. Decomposition of the Generator module.



The Encoder processes submitted form with respect of the OASIS CIQ 3.0 specification XML schema and optionally a list of extended attributes (if defined). While processing the form data, an URL of the participant's homepage is stored in the XML database file (see Listing 7) for future use by on-line catalog (if participant allows to do it).

The content of the generated digital business card (a CIQ-encoded XML-document) for Tallinn University is included in Appendix B. The generated file can be saved by participant to its computer or may be sent to a person, responsible for the participant's homepage (usually named web-master). After business card will be uploaded to the homepage, it becomes accessible to the software agent of the distributed contact management system. The only issue that needs to be resolved before software agent can read data from the on-line business card is the discovery of the business card on the remote web-server.

4.2.3 Discovery of the On-line Business Card

When the business card is uploaded to the web-site of its owner, it may be accessed from the Internet. What is the URL of the on-line business card? The simplest way is to agree, that the filename of the business card should be, for example, `businesscard.xml` and it should be stored in the root directory of the web-server. In this case software agent, who knows the URL of the homepage can construct an access URL and get the business card from the path:

`http://www.example.org/businesscard.xml`

Unfortunately, this would be incorrect approach, as any hard-coding of file named and paths is not a good software development practice. A user should have a possibility to change the filename (for example, if there is any other file with the same name on the web-server, originated by any other application) and place it into any of the subdirectories. For example, each department of the large enterprise may have its own on-line business card on the department homepage:

- http://www.example.org/department1
- http://www.example.org/department2
- http://www.example.org/department3

To avoid possible collisions and let users place file wherever they want, a software agent should be capable to discover business card by provided URL of the homepage only. This solution is known under "auto-discovery" term and is successfully used by many existing standards, for example, by FOAF (Brickley & Miller, 2007). The idea behind auto-discovery is to put a meta-information about the file name and path in the `<head> section` of HTML homepage (so-called index-file). The markup used by FOAF specification is shown in Listing 8.

Listing 8. Auto-discovery element according to the FOAF specification. (Brickley & Miller, 2007).

```
<link rel="meta" type="application/rdf+xml" title="FOAF"
 href="http://example.com/~you/foaf.rdf" />
```

Author prefers not to reinvent the wheel and acknowledges the auto-discovery element offered by the FOAF specification as de-facto standard for auto-discovery approach. Based on this decision, an auto-discovery element for OASIS CIQ standard is introduced in Listing 9 (currently OASIS CIQ 3.0 specification does not have one):

Listing 9. Auto-discovery element for the OASIS CIQ.

```
<link rel="meta" type="application/xml" title="CIQ"
 href="http://example.org/department/ciq.xml" />
```

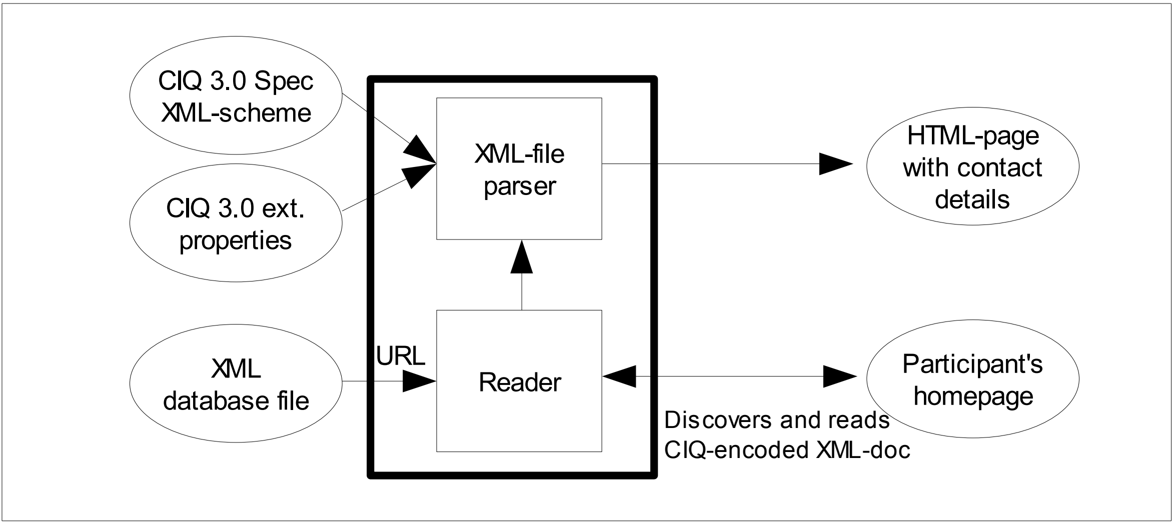
The difference between FOAF and OASIS CIQ auto-discovery elements is in the type of the linked file. While FOAF uses RDF-encoding (with corresponding file extension and MIME-type), the OASIS CIQ must use ones for XML encoding. It doesn't really matter what filename participant chooses for his on-line business card, although `ciq.xml` is a common choice and this filename is given by the Generator. Of course, user must change the URL in the `href` property to point to his own business card name and location. Both absolute and relative paths are allowed.

After the on-line business card document is uploaded to the web-server and auto-discovery element is added to the homepage, a Software Agent may read the data from the business card.

4.2.4 The Software Agent

The workflow diagram of the Software Agent module is depicted in the Figure 6. The module consists of two components: a reader, and a XML-parser. The main task for the reader is to discover and read-in an on-line business card from the remote web-side using provided URL. The reader passes obtained XML-structure to the XML-parser that uses the same OASIS CIQ 3.0 specifications as the Generator to validate the file of the business card. If XML-structure was parsed without any errors, parser returns to the user's web-browser a HTML-document, which will be rendered in a human-readable way.

Figure 6. Decomposition of the Software Agent module.



4.3 Implementation Phase

Author implemented the designed prototype as a web-application with *Ruby*[55] programming language (version 1.8) using *Ruby on Rails*[56] (RoR) software development framework (version 2.3). During the implementation phase the following tools and technologies were applied:

- *Asynchronous JavaScript and XML technology*[57] (AJAX) in the Software Agent module to discover and read on-line business cards on the remote homepages without annoying a user by extra refreshed on the web-page;
- (X)HTML and CSS to serve the content to a user;
- REXML[58] API for RoR to generate and parse XML-files;
- *e TextEditor*[59] to write the code;
- *TortoiseSVN*[60] to manage code versions;
- *Altova XMLSpy*[61] 2009 (trial version) for validating XML-files and schemes.

The implementation process was guided by the *Getting Real* agile development methodology, that helps developer to focus on building a successful web application. *Getting Real* methodology is ideal for software rapid prototyping, as it requires developer to start implementation from the user interface, allows him to skip documentation process in favor of in-line code documentation and prepares programmer for constant improvement of the created prototype with low cost of change. (37signals, n.d.).

4.4 Evaluation Phase

After creating a working prototype, author evaluated it by generating on-line business cards for different organizations, publishing them on-line and then accessing contact details with the Software Agent and the White pages catalog. This chapter exposes only negative aspects of the evaluation: shortcomings, "bottlenecks", limitations and drawbacks of the current design and implementation.

- The main drawback of the OASIS CIQ specification from the developer's point of view is the extreme verbosity of the specification (it can be seen in the Appendix B). The necessity of the extra XML elements and additional attributes is caused by the extensibility of

55 http://www.ruby-lang.org/en/
56 http://rubyonrails.org
57 http://www.w3schools.com/Ajax/Default.Asp
58 http://www.germane-software.com/software/rexml/
59 http://www.e-texteditor.com/
60 http://tortoisesvn.tigris.org/
61 http://www.altova.com/xml-editor/

specification and there is no possibility to overcome this shortage. Developers should do their best while encoding files with OASIS CIQ specifications to expel missing of essential elements, attributes or block nesting. Otherwise, the XML-document would not be valid. The same goes for expanding XML-scheme with custom properties.

- The work with XML-scheme and complex XML-documents derived from more than one scheme is complicated due to the absence on handy web-based open-source tools for validating an XML-document against multiply XML-schemes or for validating child scheme against the parent one. The only fully functional solution capable of doing it is commercial product *XMLSpy*, which still does not have an adequate alternative open-source application.
- OASIS CIQ 3.0 standard currently does not include xPRL specification; as a result, relationships between business cards cannot be defined. This shortcoming significantly limits the adaptation of contact exchange system based on OASIS CIQ, as many mid-sized organization (and definitely all large organizations) are divided into departments or have branch offices, which have some contact details in common. Current version of the prototype requires each branch to generate its own business card, but for this case there should be option to extend the existing business card by filling additional properties or overwriting existing attributes.
- The auto-discovery feature limits the audience of the system with experienced computer users only, as typical Internet user may know nothing about how to add extra tag into the index file on his homepage.
- Currently the only possible way to edit the data is to open business card in a text editor and replace the old text with the new one. It is also realizable by experienced users, who have at least basic web-development skills. Other people are forced to enter all the data again and to generate new business card, if they just wish to change the phone number. The prototype should be extended with the editing feature (Software Agent module reads remote business card and fills data entry form with obtained information, which can be edited by the user).
- Current implementation has no mechanisms for authentication and authorization of software agents, which read the on-line business cards. This circumstance can bring to the data leak to *spammers* and other unwanted parties, who may get access to the contact details.
- Business cards cannot be authenticated in the current system design. As a result, anyone may make a false business card of any other organization and register it as valid one. In the same time, the organization does not have any methods to get know about misuse of its

credentials.

- Developed contact management system may lose track of participant, if its homepage URL changes. In this case participant should manually re-register his new URL (in terms of the prototype he should generate a new business card and save his new URL in the White pages catalog).

The list above names only the main problems of the developed prototype. Some of them can be quite easily resolved during the next iteration of the development cycle. The others need more thorough design and even in deep study of possible solutions. In no doubt, the evaluation of the prototype by the end-users will expose even more defects, that's why some kind of bugs reporting tool should be provided users on the project homepage.

4.5 Conclusion from Chapter 4

Key points derived by author from the above exploration of prototype development process are:

- Despite the developed prototype was not a complex software application, a system approach to the design and implementation phases is a must in order to deliver a fully-functional application. The use of design- and implementation methodologies is important, and the selection of the proper one is crucial task.

- Author successfully developed a prototype of the distributed contact exchange system using *Ruby on Rails* web-development framework and innovative *Agile Modeling* and *Getting Real* methodologies.

- The developed prototype was evaluated by author without assistance of end-users. This exposes a number of cases that need more thorough investigation, in order to make improvements to the prototype functionality.

Chapter 5 **Conclusions**

Every year organizations are losing money due to outdated contact information about their customers and partners. The purpose of this research is to offer organization of every size and kind a new contact information management approach, which keeps their contact details up to date. Thus, it may become a solution of the data quality problem.

The cornerstone of the proposed approach is an on-line business card that would help participants to obtain perfect contact details and identity credentials about each other in no time, with low setup costs and maintenance expenses. The main parts of the on-line business card are its data model (the list of properties about card's owner, that card holds) and the encoding of this data in a machine-readable form. The data encoding with the proper specification is important, as on it right selection depends the adoption of the approach, it spread and contribution into the problem domain.

The first goal achieved in the current research is the knowledge, how far existing web-standards related to contact information management, to be applicable for on-line business card exchange system. The exhaustive information about contact management specifications was combined in one document and analyzed from the viewpoint of the organizations, taking into account requirements and the recommendations of the Semantic Web.

Among three standards, participating in the comparison of their features and potential, – vCard, *Friend of a Friend* and OASIS *Customer Information Quality* (CIQ), – the last one was selected as best suitable specification for representing business cards of the organizations. OASIS CIQ proved to be flexible and extensible set of specifications, capable to present a great number of customer's details in the form of machine-readable XML-document.

The developed prototype of the proposed contact management system is the second goal achieved by this research. The prototype is designed as a web-application to visually demonstrate benefits of distributed approach to the contact management. The prototype introduces a distributed White pages directory, which generated and reads on-line business cards encoded with OASIS CIQ specification. Once published on the participant's homepage business card may be integrated with any other applications that get support for OASIS CIQ standard.

Author shares research results on the project homepage[62] under *Creative Commons Attribution* license and invites all persons who are not indifferent to the problems of contact details exchange and data quality, to evaluate the prototype and share their ideas about it improvements and further development.

During the self-evaluation phase of prototype development cycle author nominated a few problems to become topics of future study. These are authentication and authorization of data in distributed contact management systems, as well as necessity to build up a more robust data model and system specification on top of OASIS CIQ, to utilize all potential of this standard at the service of distributed contact information management system.

In no doubt, the discussion about distributed contact management approaches is not finished. The development of the specifications is going further, as every analyzed standard has a strong user's community behind. Perhaps, even more new specifications and ontologies for party information management domain will emerge. There is still no single standard, today's Internet society is on the way towards standardization of on-line business cards, and author hopes his research takes another step on this way.

62 http://ciq.bits.ee

Bibliography

37signals (n.d.). *What is Getting Real?* Retrieved May 1, 2009, from http://gettingreal.37signals.com/ch01_What_is_Getting_Real.php

Ambler, S. (2007a). *Agile Modeling. Effective Practices for Modeling and Documentation.* Retrieved April 27, 2009, from http://www.agilemodeling.com/

Ambler, S. (2007b). *Agile Modeling Principles v2*. Retrieved April 27, 2009, from http://www.agilemodeling.com/principles.htm

Anderson, N. (2006, September 1). *Tim Berners-Lee on Web 2.0: "nobody even knows what it means"*. Ars Technica. Retrieved April 10, 2009, from http://arstechnica.com/business/news/2006/09/7650.ars.

Antoniou, G., & Harmelen, F. (2004). A Semantic Web Primer. 2 ed. The MIT Press

Barab, S. & Squire, B. (2004). *Design-based reserach: Putting a stake in the ground.* The Journal of the Learning Sciences, 13 (1), 1-14.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001, May). *The Semantic Web.* Scientific American Magazine. Retrieved April 10, 2009, from http://www.sciam.com/article.cfm?id=the-semantic-web&page=2.

Betts, M. (2002, February 18). *Data quality: The cornerstone of CRM*. Computerworld, Inc. Retrieved April 20, 2009, from http://www.computerworld.com/action/article.do?command=viewArticleBasic&articleId=68270

Brickley, D., & Miller, L. (2007, November 2). *FOAF Vocabulary Specification*. Namespace document. Retrieved April 26, 2009, from http://xmlns.com/foaf/spec/.

CitCity.ru (n.d.). *CIQ. XML-стандарт для управления клиентской информацией*. Retrieved April 21, 2009, from http://citcity.ru/15257/.

Collins, A., Bielaczyc, K., & Joseph, D. (2004). *Design experiments: Theoretical and methodological issues.* Journal of the Learning Sciences, 13(1), 15-42.

D'Arcus, B. (2004, February 13). *xNAL: A Global Standard for Names?* Blog post. Retrieved April 27, 2009, from http://community.muohio.edu/blogs/darcusb/archives/2004/02/13/xnal-a-global-standard-for-names.

Dawson, F., & Howes, T. (1998, September). *vCard MIME Directory Profile*. RFC2426. Retrieved April 25, 2009, from http://tools.ietf.org/html/rfc2426.

Design-Based Research Collective. (2003). *Design-based research: An emerging paradigm for educational inquiry.* Educational Researcher, 32(1), 5-8.

Eckerson, W. (2001). *Data Quality and the Bottom Line.* Excerpt from TDWI's Research Report. Retrieved April 20, 2009, from https://www.tdwi.org/research/display.aspx?ID=6589

EENet (2009, March 18). *Ee tsoonis registreeriti 66 000. domeeninimi*. Retrieved April 20, 2009, from http://www.eenet.ee/EENet/uudised

Encyclopedia of Database Systems (n.d.). *Ontology (Computer Science), definition.* Retrieved April 10, 2009, from http://tomgruber.org/writing/ontology-definition-2007.htm.

eSight (n.d.). *Contact Management*. Retrieved April 14, 2009, from http://www.esight.org/OverviewByType.cfm?ovid=970

Future of the Internet (2009). *Conference materials. General information.* Retrieved April 12, 2009, from http://www.fi-prague.eu/info/.

Gartner, Inc. (2008a, April 22). *Gartner Says Worldwide CRM Software Market to Grow 14 Percent in 2008*. Retrieved April 20, 2009, from http://www.gartner.com/it/page.jsp?id=653307

Gartner, Inc. (2008b, July 7). *Gartner Says Worldwide Customer Relationship Management Market Grew 23 Percent in 2007*. Retrieved April 20, 2009, from http://www.gartner.com/it/page.jsp?id=715308

Halpin, H., Suda, B., & Walsh, N. (2006). *An Ontology for vCards.* Retrieved April 26, 2009, from http://www.w3.org/2006/vcard/ns.

Hansapank (2009, March 5). *Alates 17. märtsist on ASi Hansapank ärinimi Swedbank AS*. Press-release. Retrieved April 22, 2009, from http://w.hansa.ee/est/uudised_28826_2009_03_1.html.

Iannella, R. (2001, February 22). *Representing vCard Objects in RDF/XML*. W3.org. Retrieved April 26, 2009, from http://www.w3.org/TR/vcard-rdf.

Juran, J.M., & Godfrey, A.B. (1999). *Juran's quality handbook*. 5 ed. New York: McGraw-Hill.

Learning Theories. (n.d.). *Design-Based Research Methods*. Retrieved April 20, 2009, from http://www.learning-theories.com/design-based-research-methods.html.

McKendrick, J. (2008, December). *The 15 Top Information Management Trends for 2009.* Database Trends and Applications. The Journal of Information Integration and Management. Retrieved April 12, 2009, from http://www.dbta.com/e-edition/Dec08/2-contributed_article_mckendrick.html.

Morris, S. (2005, June). *Software Development Cycle.* Retrieved May 3, 2009, from http://www.tessella.com/wp-content/uploads/2008/05/softwaredevelopmentcycle.pdf.

OASIS (n.d.). *Customer Information Quality Technical Committee*. Retrieved May 2, 2009, from http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ciq.

OASIS (2008, September 20). Customer Information Quality (CIQ) Specifications Version 3.0. Retrieved May 1, 2009, from http://docs.oasis-open.org/ciq/v3.0/specs/ciq-specs-v3.html

Peelen, E. (2006, April 12). *Poor data quality costs businesses 400 million annually.* Human Inference. Data Quality Solutions. Retrieved April 10, 2009, from http://www.humaninference.com/company/press/262

Perreault, S. (2008, November 18). *vCard XML-schema.* Network Working Group Internet-draft (expires 2009, May 22). Retrieved April 26, 2009, from http://tools.ietf.org/id/draft-perreault-vcarddav-vcardxml-00.txt.

Perreault, S., & Resnick, P. (2009, March 5). *vCard Format Specification*. Network Working Group, Internet-draft (expires on 2009, September 6). Retrieved April 26, 2009, from http://tools.ietf.org/html/draft-ietf-vcarddav-vcardrev-06

RIK (2009, March 1). *Äriregister ning mittetulundusühingute ja sihtasutuste register maakonniti.* Registrite ja Infosüsteemide Keskus. Retrieved April 20, 2009, from http://www.rik.ee/stat/9_3mk.phtml

Spivac, N. (2004, November 15). *The Ontology Problem: A Definition with Commentary.* Blog post. Retrieved April 12, 2009, from http://novaspivack.typepad.com/nova_spivacks_weblog/2004/11/the_ontology_pr.html

Tauberer, J. (n.d.). *Quick Intro to RDF.* Retrieved April 11, 2009, from http://rdfabout.com/quickintro.xpd.

W3.org (2009). *W3C Semantic Web Activity.* Retrieved April 20, 2009, from http://www.w3.org/2001/sw/

Wang, F., & Hannafin, M. J. (2005). *Design-based research and technology-enhanced learning environments.* Educational Technology Research and Development, 53(4), p. 5-23.

Appendix A
List of On-line Catalogs

	URL	Country of Origin
1	http://www.yellowpages.com.au/	Australia
2	http://www.yellowpages.ca/	Canada
3	https://ariregister.rik.ee	Estonia
4	http://www.kontakt.ee	Estonia
5	http://www.infoatlas.ee	Estonia
6	http://www.yellowpages.ee	Estonia
7	http://www.yellowpages.co.id/	Indonesia
8	http://www.lursoft.lv	Latvia
9	http://www.visalietuva.lt	Lithuania
10	http://www.yellowpages.com.my/	Malaysia
11	http://www.kompass.md	Moldova
12	http://www.yellowpagesrussia.ru/	Russia
13	http://www.spr.ru	Russia
14	http://www.yellowpages.co.za/	South Africa
15	http://www.yellowpages.ae/	UAE
16	http://www.meta.ua	Ukraine
17	http://www.yell.com/	United Kingdom
18	http://www.switchboard.com	USA
19	http://www.superpages.com/	USA
20	http://buildinginfo.eu	Various Countries

Appendix B
Prototype output – ciq.xml

```
<?xml version='1.0'?>
<p:OrganisationDetails
   xmlns:xlink='http://www.w3.org/1999/xlink'
   xmlns:a='urn:oasis:names:tc:ciq:xal:3'
   xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
   xmlns:n='urn:oasis:names:tc:ciq:xnl:3'
   xmlns:c='urn:acme.org:corporate:customers'
   xsi:schemaLocation='urn:oasis:names:tc:ciq:xpil:3
                       http://ciq.bits.ee/schemes/xPIL.xsd'
   xmlns:p='urn:oasis:names:tc:ciq:xpil:3'
   xmlns:ct='urn:oasis:names:tc:ciq:ct:3'>
  <n:OrganisationName>
    <n:NameElement n:ElementType='NameOnly'>
      Tallinn University
    </n:NameElement>
    <n:NameElement n:ElementType='TypeOnly'>
      Non-profit organization
    </n:NameElement>
  </n:OrganisationName>
  <p:Addresses>
    <p:Address a:Usage='Postal'>
      <a:Premises>
        <a:NameElement>
          Narva mnt 25
        </a:NameElement>
      </a:Premises>
      <a:Locality>
        <a:NameElement>
          Tallinn
        </a:NameElement>
      </a:Locality>
      <a:Country>
        <a:NameElement>
          Estonia
        </a:NameElement>
      </a:Country>
      <a:PostCode>
        <a:Identifier>
          10120
        </a:Identifier>
      </a:PostCode>
    </p:Address>
  </p:Addresses>
  <p:Accounts>
    <p:Account>
      <p:AccountElement p:Type='AccountType'>
        IBAN
      </p:AccountElement>
      <p:Organisation>
        <n:NameElement n:ElementType='FullName'>
          SEB Pank
        </n:NameElement>
      </p:Organisation>
      <p:AccountElement p:Type='AccountID'>
        EE071010002006943007
      </p:AccountElement>
    </p:Account>
  </p:Accounts>
  <p:ContactNumbers>
    <p:ContactNumber p:CommunicationMediaType='Telephone'
p:ContactHours='9.00-17.00'>
      <p:ContactNumberElement p:Type='LocalNumber'>
        640 9101
      </p:ContactNumberElement>
      <p:ContactNumberElement p:Type='CountryCode'>
        +372
      </p:ContactNumberElement>
    </p:ContactNumber>
  </p:ContactNumbers>
  <p:ElectronicAddressIdentifiers>
    <p:ElectronicAddressIdentifier p:Type='URL'>
      http://www.tlu.ee
    </p:ElectronicAddressIdentifier>
    <p:ElectronicAddressIdentifier p:Type='EMAIL'>
      tlu@tlu.ee
    </p:ElectronicAddressIdentifier>
  </p:ElectronicAddressIdentifiers>
  <p:Identifiers>
    <p:Identifier p:Type='TaxID'>
      <p:IdentifierElement p:Type='Identifier'>
        EE100251335
      </p:IdentifierElement>
    </p:Identifier>
    <p:Identifier p:Type='RegistrationID'>
      <p:IdentifierElement p:Type='Identifier'>
        74000122
      </p:IdentifierElement>
    </p:Identifier>
  </p:Identifiers>
  <p:OrganisationInfo p:OperatingHourStartTime='8:00'
p:OperatingHourEndTime='20:00'/>
</p:OrganisationDetails>
```

KOKKUVÕTE

Tänapäeval on üha rohkem ettevõtteid hakanud kasutama kliendihaldustarkvara, mis võimaldab salvestada kõiki vajalikke andmeid (sh. kontaktandmed) ettevõtte klientide, tarnijate jt. partnerite kohta. *Gartner'i* analüüsid (2008a, 2008b) kinnitavad, et kliendihaldustarkvara turg on 2007 ja 2008 aastal kasvanud ning taoline kasvutrend jätkub vähemalt 2012. aastani.

Koos kliendihaldustarkvara leviku ja töödeldava andmemahu kasvuga, on aga tekkinud vajadus kanda hoolt säilitatavate andmete, eelkõige kontaktandmete kehtivuse eest. Isegi stabiilses majandusruumis aeguvad andmebaasi kontaktandmed ca 2 protsendil kirjetest kuus. Tingitud on see inimeste loomulikust migreerumisest, perekonnaseisu või töösuhete muutumisest vms. tavapärastest põhjustest. Ebakindlates majandustingimustes tuleb aga mistahes muutusi palju rohkem ette ning kogutud andmete kasutatavus aegub veelgi kiiremini. Teave ühe või teise isiku kontakt- või isikuandmetes toimunud parandustest ei jõua aga küllalt kiiresti kõikide koostööpartneriteni, kes on seetõttu sunnitud vahel üsnagi pikka aega opereerima vananenud ülestähendustega.

Ebaõigetele kontaktandmetele tuginemine on enesestmõistetavalt kahjulik nii ettevõttele kui ka kliendile. 2006. aastal põhjustas üksnes Hollandis saadetiste järelesaatmisest, valesti adresseeritud arvete kordustrükkimisest ja vanadele telefoninumbritele helistamisest tingitud ülemäärane kulu materiaalset kahju ca 400 miljonit eurot (Peelen, 2007). Seejuures ei kajasta mainitud summa vahetult kliendile tekitatud kahju, näiteks saadetiste hilinenud kohaletoimetamise eest.

Käesolev magistritöö uurib, kuidas saaks parandada andmebaasides olevate andmete kvaliteeti ning tagada kiire teave nende muutumisest. Ühe võimaliku lahendusena pakub autor virtuaalseid nimekaarte, mida iga soovija saaks endale luua lihtsate IKT vahenditega, avaldades seejärel oma digitaalse nimekaardi on-line keskkonnas (kas oma kodulehel, ajaveebis või mujal Internetis), kus see oleks kliendihaldustarkvara agentidele loetav. Sel juhul piisab kliendil kontakt- või isikuandmete muutumise korral vaid andmete uuendamisest oma isiklikul on-line nimekaardil ning need saavad otsekohe kõigile partneritele kättesaadavaks. Kaob vajadus toimunud muudatustest kõiki ükshaaval teavitada.

Töö eesmärgiks on selgitada välja, kuivõrd olemasolevad kontaktandmete esitamise veebistandardid sobivad on-line nimekaartide vahetussüsteemi rajamiseks, ning luua selle süsteemi prototüüp. Prototüübi loomisel piirab autor süsteemi kasutajaskonda esialgu vaid organisatsioonidega, olles veendunud, et eduka käivitamise korral võib väljapakutud lahendust laiendada, kaasates virtuaalsete nimekaartide vahetamisele ka eraisikuid.

Sisuliselt kujutab töö endast arendusuuringut, mille teoreetiline osa põhineb kvalitatiivsel dokumendiuuringul. Saadud teadmised läbivad proovitsükli ning autor täiendab neid veebirakenduse loomise käigus, mis moodustab töö empiirilise osa.

Dokumendiuuring algab Semantilise Veebi mudeli analüüsimisega, mis on paljude kaasaegsete veebirakenduste alus. Seejärel kirjeldatakse kolme põhilist lähenemisviisi kontaktinfo töötlemisele ja selgitatakse käsitletud mooduste eripära – plusse ja miinuseid. Peatähelepanu dokumendiuuringu raames on aga pühendatud kontaktinfo haldamiseks mõeldud veebistandardite analüüsimisele ja võrdlemisele, eesmärgiga selgitada välja, milline spetsifikatsioon sobib kõige paremini on-line nimekaartide süsteemis andmekandjaks.

Võrdluses osales kolm spetsifikatsiooni: *vCard 3.0, Friend of a Friend (FOAF)* ja *OASIS Customer Information Quality 3.0 (CIQ)*. Asja objektiivsuse tagamiseks olid defineeritud nõudmised, millele peab sobivaks tunnistatud standard maksimaalsel võimalikul määral vastama. Töö käigus selgusid iga standardi tugevused ja nõrkused ning lõpptulemusena osutus parimaks *OASIS CIQ*.

Tuginedes valitud standardile, on loodud süsteemi prototüüp, mis võimaldab genereerida tavakasutajatele digitaalseid nimekaarte, sisestades asutuse põhilised andmed veebivormi kaudu. Nimekaart on kodeeritud *XML*-vormingus vastavalt *OASIS CIQ* versioon 3.0 spetsifikatsioonile. Peale nimekaardi avaldamist asutuse veebilehel on sama prototüübi agendiga võimalik lugeda nimekaardilt asutuse kontaktandmed ja kuvada neid prototüübi kataloogis.

Kokkuvõtvalt on magistritöö tulemused järgmised:

- seni killustatud teave kontakt- ja isikuandmete kirjeldamise spetsifikatsioonide kohta on koondatud ühte dokumenti ning on tehtud nende analüüs ja võrdlus;
- on loodud ja käivitatud on-line nimekaartide vahetussüsteemi prototüüp, mis võib olla

eeskujuks nii tarkvara-arendajatele, kes soovivad lisada oma tarkvarasse on-line nimekaartide tuge, kui ka asutustele, kes soovivad lihtsate meetmetega teavitada oma partnereid kontaktandmete muutumisest.

Töö tulemused, sealhulgas ka veebirakenduse lähtekoodid, on kõigile huvilistele vabalt kättesaadavad projekti kodulehel Internetis aadressil http://ciq.bits.ee. Autor levitab neid *Creative Commons Attribution* litsentsi alusel ning huviliste igakülgne osalemine koos kriitika ja ettepanekutega asja täiustamise ja arendamise kohta on teretulnud.

Kahtlemata ei ole diskussioon kontaktandmete standardiseerimise ja levitamise üle veebi kaudu veel kaugeltki lõppenud. Pigem vastupidi – tänapäeval pole sedalaadi ettevõtmistel kindlat liidrit, kuna iga kirjeldatud standardi taga seisab tegus kasutajaskond, kes omalt poolt tagab spetsifikatsiooni edasiarenduse. Sugugi pole välistatud ka uute standardite ilmumine, üritamaks lahendada konktakt- ja isikuandmete infovahetusprobleeme. On-line nimekaartide standardiseerimine pole veel jõudnud sihtpunkti ning autor loodab, et ka tema töö annab sellel teel oma panuse.

Magistritöö on kirjutatud inglise keeles, koosneb 5 osast, sisaldab üle 18 000 sõna, 6 joonist ja 10 tabelit ning kokku omab 59lk sisulises osas.

Võtmesõnad: *Web2.0, Semantiline Veeb, kliendihaldus, kontaktihaldus, prototüüp, kontaktandmed, tarkvaraarendus, Ruby, Ruby on Rails, RoR, CRM, XML, vCard, OASIS, CIQ, FOAF, ontoloogiad.*