Development of an ImmunoCAP® ISAC database application

Lennie Fredriksson
Abstract

Development of an ImmunoCAP® ISAC database application

Lennie Fredriksson

Over the last decades the number of persons suffering from allergy has increased significantly. As many as 30-40% of the world’s population suffers from allergy of some kind. Allergy is also a costly post for the society, and for the individuals, in loss of work and over- or wrong- medication. The tools available for clinicians to diagnose allergy are; taking a thorough patient history and taking an allergy test. However, most allergy diagnoses are only based on the clinical history without any allergy tests being performed.

The most commonly used allergy tests are skin prick test (where an allergen is introduced into the skin and a skin reaction is registered as a positive test) and blood tests (where IgE antibody binding to an allergen is measured). One drawback with today’s allergy tests is that they are a relatively high percentage of false positive test results (i.e. patients having a positive test without allergic symptoms), and false negative test (patients with clinical symptoms but a negative test result).

Phadia’s molecular allergy diagnostic system, ImmunoCAP® ISAC, produces a lot of valuable data, when diagnose patient’s allergy status. The purpose of the project was to develop an application for collecting this data as optimized as possible using a SQL Server database. Also to visualize the data in the database in a way that compares patients and allergen components. To facilitate the comparison of allergens a feature called Rules were developed into the application.
Table of Contents

1 INTRODUCTION .................................................................................................................. 8

1.1 AIM OF PROJECT ............................................................................................................ 8

1.2 BACKGROUND .................................................................................................................. 8

1.2.1 ALLERGY DIAGNOSTICS .......................................................................................... 9

1.2.2 IMMUNOCAP® ISAC ................................................................................................. 9

1.2.3 DECISION SUPPORT SYSTEMS ................................................................................. 10

1.2.4 AIM OF DATABASE APPLICATION ......................................................................... 10

1.2.5 OTHER DATABASE EXAMPLES ................................................................................. 11

1.2.5.1 KIGS and KIMS ..................................................................................................... 11

1.2.5.2 Allergome ............................................................................................................... 11

2 METHOD .................................................................................................................................. 12

3 REQUIREMENTS ...................................................................................................................... 12

3.1 STORING IMMUNOCAP® ISAC DATA ......................................................................... 12

3.2 VISUALIZING IMMUNOCAP® ISAC DATA ................................................................... 13

3.3 STATISTICS OF DATABASE CONTAINING IMMUNOCAP® ISAC DATA ..................... 14

3.4 DEVELOPMENT TOOL ..................................................................................................... 14

3.5 SUMMARY OF REQUIREMENTS .................................................................................. 15

4 DESIGN ................................................................................................................................... 15

4.1 ARCHITECTURAL DESIGN ............................................................................................. 16

4.2 IMPLEMENTATION DESIGN ............................................................................................ 16

5 IMPLEMENTATION ............................................................................................................... 17

6 STRUCTURE AND TECHNICAL SOLUTIONS OF THE APPLICATION ....................... 18

6.1 ARCHITECTURE ............................................................................................................... 18

6.1.1 DATABASE .................................................................................................................. 18

6.1.2 THE WCF SERVICE .................................................................................................... 20

6.1.3 SERVICE PROVIDER – THE HOST ............................................................................. 21

6.1.4 SERVICE CONSUMER – THE WPF CLIENT ............................................................... 21

6.1.5 SETUP AND INSTALLATION ...................................................................................... 22

6.1.6 APPLICATION TESTING AND EVALUATION ......................................................... 22
7 GRAPHICAL USER INTERFACE ........................................................................................................ 22

7.1 STORING IMMUNOCAP® ISAC DATA .......................................................................................... 24
7.2 VISUALIZING AND ANALYZING IMMUNOCAP® ISAC DATA ..................................................... 26
7.3 STATISTICS OF DATABASE CONTAINING IMMUNOCAP® ISAC DATA ..................................... 28

8 VERIFICATION .................................................................................................................................. 28

9 EVALUATION ..................................................................................................................................... 29

10 FUTURE DEVELOPMENT ................................................................................................................ 29

11 FINAL CONCLUSIONS ................................................................................................................... 29

12 REFERENCES ..................................................................................................................................... 30

APPENDIX A APPLICATION FIGURES .............................................................................................. 31

APPENDIX B APPLICATION SOLUTION STRUCTURE ........................................................................ 33
1 Introduction

1.1 Aim of Project

The aim of the project is to investigate and develop an (demo) application for storing and visualizing ImmunoCAP® ISAC data. From a number of requirements and specifications, design of application architecture and implementation the design was then performed. This application was then tested and evaluated by Phadia employees.

The project involved development of a database storing of ISAC result data (csv data files), development of a client graphical user interface (GUI) for uploading csv data files to the database, visualization of data stored in the database, and finally some evaluation of useful, future-database statistics.

Visualization of data should be done in a way that will help the users to get an overview of the results. From this overview the application should facilitate the user to identify different IgE patterns, and to export the data to other data mining softwares for more advanced data analysis.

This application is not the final ISAC database system but should be regarded as a first step where ISAC data can be stored and it can also be used as input when planning for a future world-wide database system in routine use.

1.2 Background

Over the last decades the number of persons suffering from allergy has increased significantly. As many as 30-40% of the world’s population suffers from allergy of some kind. Allergy is also a costly post for the society, and for the individuals, in loss of work and over- or wrong-medication. The tools available for clinicians to diagnose allergy are; taking a thorough patient history and taking an allergy test. However, most allergy diagnoses are only based on the clinical history without any allergy tests being performed. It has been shown though that testing can save money; for example [1] estimates that the costs per patient over a 2 years period decreased from 784€ in the no-test strategy to 535€ in the test strategy, and this will also improve the quality of life for the patient. The most commonly used allergy tests are skin prick test (where an allergen is introduced into the skin and a skin reaction is registered as a positive test) and blood tests (where IgE antibody binding to an allergen is measured). One drawback with today’s allergy tests is that they are a relatively high percentage of false positive test results (i.e. patients having a positive test without allergic symptoms), and false negative test (patients with clinical symptoms but a negative test result). Recent development in the field of molecular allergology indicates that a molecular allergy diagnostics approach have the possibility to improve the clinical usefulness of allergy testing.

Phadia AB [2, 3] is a global leader in allergy, asthma and autoimmunity diagnostics. Phadia develops, manufactures and markets complete blood test systems to support clinical diagnosis and management. Phadia is owned by the leading European buyout firm Cinven. The head quarter is located in Uppsala, Sweden, including production, research and development, and management. Some 98% of Phadia AB production is exported to more than 3,000 laboratories in 50 countries.

Phadias diagnostic technology is based on the discovery of the connection of immunoglobulin E (or IgE) antibodies and allergic symptoms, made at Uppsala University Hospital in 1967 (S.G.O Johansson, Hans Bennich, Leif Wide). The discovery was made during the development of a method to measure the levels of IgE in blood samples. These findings were presented to Pharmacia with the intent of creating a commercial allergy blood test. With the help of the Sandwich technology,
developed by Leif Wide, the product Phadia RAST was established, and with it, the leading position Phadia still holds in allergy testing.

1.2.1 Allergy diagnostics

The aim of allergy testing [4] is to identify the sources to which a person binds IgE antibodies. Traditional testing (both SPT and blood tests) is extract based, which means that the allergen test substance contains basically all molecules in allergen source like peanut or timothy grass pollen. The extract based test has the drawback of not identifying the trigger molecule, and the advantage of covering most molecules in the allergen source.

The new era in allergy diagnostics is molecular allergy diagnostics. This technology can identity triggering molecules giving information about;

a) Cross-reactivity: Do I need to consider one or several allergen sources, i.e. is the molecule species-specific or cross-reactive, and

b) Risk assessment: Is there a risk of having a more severe reaction or only mild symptoms.

Molecular Allergy (MA) Diagnostics can be performed in singleplex (one-test-at-the-time approach) with Phadia 250/1000/2500/5000 systems or in multiplex (>100 of tests in parallel) on a biochip with ImmunoCAP® ISAC system. ImmunoCAP® ISAC is the first and only available micro-array allergen-molecule chip on the market. With this system the IgE antibody concentration for 103 allergen components is measured. In general, low IgE antibody levels indicate a low probability of clinical disease, whereas high antibody levels to an allergen show good correlation with clinical disease, but exceptions occur.

1.2.2 ImmunoCAP® ISAC

Phadia is developing and marketing ImmunoCAP® ISAC (Immuno Solid-phase Allergen Chip) [5, 6, 7]. This system is an array-based, biochip technology, and it is the most advanced in vitro diagnostics test for measurement of specific IgE antibodies to allergen components. All the steps in the testing with ImmunoCAP® ISAC are shown in figure 1. ImmunoCAP ISAC® is a multiplex diagnostic tool, for the moment analyzing 103 different allergen components in one test. The allergen components are spotted on the array (the glass plate) in triplicates.

Only 30 μl of serum or plasma, from the patient is used for a single test and the biochip delivers results of a total of 103 different answers, one from each triplicate. When the test answer is of this type, multiplex and high dimensional, help to interpret the result is almost always necessary. Therefore a decision support system has been developed at Phadia to guide personnel and doctors to interpretate the ISAC result. The doctor then needs to take the test result and interpret it in combination with other clinical data and information for the patient.
The number of steps in ISAC testing can be approximated into six major. The patient gives a blood sample, the sample is applied to the ISAC chip, staining reagents is added, the chip is scanned by a laser, the chip image is analyzed, and a report with result is created.

### 1.2.3 Decision Support Systems

The field of Decision Support System (DSS) is gigantic and has been applied in many different areas like economics, healthcare, military, transportations and many more. When it is used in the field of healthcare then it is called Clinical Decision Support System (CDSS).

Knowledge from the literature and clinical experience are implemented into the knowledge base, which for example can be defined by a rules-based system. However, ISAC is a novel system and knowledge regarding many molecules present on ISAC is still missing.

### 1.2.4 Aim of Database Application

By storing world-wide ISAC data in one place this data can be used to speed up the knowledge discovery for further data mining. Knowledge extracted and published from the database can then be implemented into the CDSS to provide the clinician with personalized interpretation support in the daily routine work.

Since ImmunoCAP® ISAC results, is complex and multidimensional it is often troublesome for the clinician/researcher to evaluated the data and expert support is often required to extract as much information as possible. That is why an ISAC database application would come in handy to gather as much ImmunoCAP® ISAC data as possible, build up expert knowledge on how to handle this kind of data to be able to support the researchers owning the data. The data can then be visualized and analyzed, for input to a CDSS for ImmunoCAP® ISAC.
1.2.5 Other Database Examples

There are some similar databases to compare with: For example the KIGS & KIMS database, where data on children and adolescents with growth disorders treated with Genotropin®/Genotonorm® respectively gathering data on adults patients with growth hormone deficiency (GH), are collected, stored and analyzed. Another example in the allergy area is the Allergome database, which contains information and data about allergen components, research articles and some analytical tools.

1.2.5.1 KIGS and KIMS

KIGS – Pfizer International Growth Database [8] is the largest and most comprehensive international pharmacoepidemiological survey on growth hormone (GH) therapy in children and adolescents. And KIGS is a tool for monitoring data on children who are receiving GH therapy with Genotropin®; the data are entered into the database at the discretion of the attending physician. The project is a collaborative effort between hundreds of growth specialists worldwide and Pfizer Endocrine Care that aims to improve the understanding of the safety of GH treatment in children. The main goal of KIGS is to generate information that will lead to new insights into the effects of GH treatment. For 20 years has KIGS gathering data, and contains now data on the safety and efficacy of treatment with GH from more than 60,000 patients in over 50 countries. For more information about KIGS, visit their web site; www.medicaloutcomes.pfizer.com.

KIMS [9] has been collecting information since 1994 about adult patients with GH deficiency both who receive and do not receive GH replacement therapy. To date it contains data on more than 13,000 patients from 31 countries. Data from KIMS are being used to optimize GH replacement therapy in adult patients with GH deficient. For more information about KIGS, visit their web site; www.medicaloutcomes.pfizer.com.

1.2.5.2 Allergome

An organization involved in allergen knowledge is Allergome. Allergome is a mixture of a lot of different parts. On their website, www.allergome.org, [10] it can be read that Allergome is a project, a web based platform for allergen knowledge and a database of allergenic molecules.

The Allergome project is currently managed by “Allergy Data Laboratories s.c.” in Italy, where the person most connected to the Allergome project, Adriano Mari, also is involved.

The Allergome platform contains four modules; RefArray, ReTiME, InterAll and AllergomeConsumer. The modules RefArray and ReTiME can be used from the Allergome website and delivers information about different allergenic molecules, their sources and references to international scientific journals. On the website statistics of information stored in Allergome is presented as monographs, which is a long list of information. This way of presenting statistics is colorless and uninspiring. InterAll cannot be used from the website, but information on the website says that it is an allergy electronic record tool for allergy/immunology clinical centers and research laboratories willing to manage patients’ clinical and diagnostic data in a very easy and modern way.
2 Method

The process used in the project follows the software development principles including; the requirements engineering process, the design process, continued with the implementation process, and as a final step a demo of the application will be installed on a computer for (staging) verification, testing and evaluation.

The project contains a number of different problems to investigate appropriate solutions to. First is an application architectural design investigated, explaining what major pieces the application should be built from. And secondly the design of these individual parts. The parts can be revealed to be the design and creation of; a database for storing data, a service handling the data transfer between database and clients and also the hosting of the service in an appropriate way, and a client graphical user interface (GUI) for uploading data to the database, visualizing data from the database, and investigating what kind of database statistics that can be interesting to present in a document or as a feature in the application.

During the development of softwares usually a lot of different artifacts are produced to document the process, as a supporting process for the project owner and the developers. In the process of system modeling many documents containing diagrams are produced; case diagram, class diagram, sequence diagram and state diagram. To make this documentation hundred percent will take a lot of time and effort, which is better used in solving the problems in the project. That is why this documentation part has been left a side, and because this project only involves one person and limited time.

3 Requirements

The major user requirements of the application can be divided into three parts; storing ImmunoCAP® ISAC data, visualization of ImmunoCAP® ISAC data and statistics of database containing ImmunoCAP® ISAC data. The part of storing data, also involves the development of the database, and the development of the service and the service host.

The number of requirements for this kind of application can be very extensive, but because the time limit of the project is just twenty weeks, only the most important requirements will be implemented in the resulting demo. The left-over requirements and the extra requirements or features that might be discovered during the implementation process will not go wasted; they will be documented for future application development.

3.1 Storing ImmunoCAP® ISAC data

The first part is the application storing data into a centralized database. A centralized database gives the advantage of having all data in one location which is favorable when you want to look at as much data as possible. But it can also be a disadvantage if it turns out to be a bottleneck, which is most likely to happen when many users want to get data fast, or when the amount of data requested from the database is large. The guess is that maybe the last case can be of importance in this application. A distributed database can also be used, but such an approach with data stored at many different locations requires a much more complex solution. Storing data in a database also provides the possibilities to search for data, and thereby selecting specific kind of data sets based on some search options.
The data that is handled by the application, as said before, comes from the ImmunoCAP® ISAC system. It has some specific features which will affect the way it is both handled and stored. First, the data comes in the format of csv files which follows a certain protocol, i.e. it has a predetermined structure, which the application is able to work with. Secondly, in the csv files the number of samples, which each constitutes of one row, can vary, so the application is able to sense that and extract each sample one by one. A sample can be seen as two parts, one part contains information about the sample, the head, and the second part contains the allergen components and their measured values, the tail. During the parsing of the csv file the data head is presented in an editable grid, compared to the tail which is not editable and thereby not showed in that grid. Third, to the head part some additional data information is added, like; sample date, run date, date of birth, gender, country etc.

One thing to keep in mind that will affect the structure and design of the database is the number of allergen components that will likely increase from today’s 103 to 111 in the future. That is why this property cannot be static; instead it has to be handled dynamically.

The kind of information that is saved to the database is important, because that information is the base of the different criteria for the selection of data for visualization. And then it is of great importance to connect data in an appropriate way to optimize the selection of data from the database. That is done using a relational database system, where connection is set up by different keys.

The database is centralized, but the clients on the other hand are distributed, i.e. the clients are able to connect to the database from computers with this application installed. That is why the connection between the database and the users, clients, is over the internet. And the two major technologies for that are services and web sites.

3.2 Visualizing ImmunoCAP® ISAC data

The second major part of the requirements of the application is visualization of data sets from the database. Visualization of data is a fast growing field right now and the complexity can be very comprehensive. That is why this part of the requirements can be heavily extensive and time-consuming. The first approach is to just visualize the data in a plain and simple way, and also make the analysis of the data more transparent. The specific structure of the data, in this case the data constitutes of a long tail with 103 allergen component values, contributes to find a way that shows the data in a more colorful way.

It is possible for the users to make a selection of what set of data the users are interested in to see. That is why the data is best saved in a database which enables the selection of data from different criteria. The number of criteria for selection of data can almost be infinitive, to ease for the users to select data some predefined criteria is created. To begin with the following selections are created; selection by investigator and related projects, selection by projects, and selection by country of patient. During the development of the application it is much likely that a lot more selections would be desirable. The question is how dynamic the selection of data can be, without having to learn the users SQL querying from scratch.

Each time a selection of data is done the result ends up in the visualization area. To be able to compare and look at two or more different selection at the same time, the visualization area is able to merge selections. For example, if a selection by country like China is first made, and then a selection by country of Japan is followed, both results are presented in the visualization area. To start all over to select data, a clear function resets the visualization area.
The data presented in the visualization area is also able to be exported back to a csv file, if the user likes to work with the data in another tool.

Working with the ISAC data, one approach is to compare different allergen components to each other. In this case when the number of allergens is quite a lot, a reminder 103, this will be tedious to do, especially if you need to copy and paste data next to each other during the process. To make this process work much faster and simpler, a feature is developed to align selected allergen components next to each other for simpler analysis.

One way to facilitate the analysis of a large grid of data is to color the values based on their size, i.e. some kind of color scheme. Low values can have one color and high values another color. This way to translate a value into a color most often creates patterns in the data set.

Working with a large number of values, sorting the values can also bring information into the analysis of data. That is way sorting of columns is a feature put into the visualization.

### 3.3 Statistics of database containing ImmunoCAP® ISAC data

The third and last major part of user requirements is the statistics of the database. This is the part with least known requirements. The total number of samples in the database is always basic statistic knowledge. Another statistical feature is to represent each allergen component with a pie chart, to which a color scheme is connected, and then each allergen component has a pie chart with colored pieces representing the number of samples within that color interval. If both the number of allergen components and the number of samples are large this process takes a fair amount of time to do, so maybe this feature is better ran on the server instead of in the application. This kind of process should just be done after checking that the database has been changed or if the change in the database is large enough, having like a threshold. If different statistical processes will be ran during the night on the server, maybe there should be a specific script containing instructions for what kind of statistical analysis that should be done. The statistical result can be summarized into a report in a pdf format and stored at a specific place containing all statistical reports.

Minor statistical analysis could be done in the application without using much time and power, like presenting pie chart statistics for just one selected allergen component.

In the future the patients may be grouped into different categories, and also different symptoms will be connected to each sample, then different statistics based on that information can be developed.

Some information about geographical differences in allergy distribution has been mentioned, so maybe some statistics based on country could be found. Either showing differences or not. This information can be valuable, especially when Phadia’s products are sold all over the world, and the question about specifying their products based on geographical location.

### 3.4 Development Tool

Phadia has its own software developing group working in the programming environment from Microsoft, called Visual Studio 2010. That is why the IDE Visual Studio 2010 is used as the development tool for this project. The latest .NET Framework 4 is used, the database is a SQL Server Express, the service managing the database is a WCF Service, Microsoft’s Windows Communication Foundation Service, and development of the client is done in WPF, Windows Presentation Foundation.
3.5 Summary of Requirements

Here are all requirements gathered, grouped into three sets, each representing one major requirement.

- Handling ImmunoCAP® ISAC data in the format of csv files.
- Show data head in editable grid with addition of data; sample date, run date, date of birth, gender, postal code, country, and for future choice; patient type, diagnosis, symptoms.
- Create and add new investigator information containing the following information; name, address, city, region, postal code, country, phone, mail.
- Create and add new project information; project title, and connect responsible investigator to the project from the investigators in the database.
- Upload of data set presented in the grid, connected to a project title in the database.

- Visualization of data sets in database selected by using some predefined queries. The first set of selection queries will be; selection by investigator and projects, selection by project, selection by country of patient, and finally a wild selection, the whole database.
- To facilitate the visualization of the different values of the allergen components, different colors are set depending on the value. A default color scheme can be developed as a first step, which can be followed by a feature for creating own color schemes.
- To facilitate comparison between different allergen components a feature for creating groups of selected allergen components is developed.
- It is also be possible to add different selections of data to the grid.
- The samples in the visualization set can also be exported to a csv file.
- Each allergen component column in the visualization set contains a sorting feature.

- The total number of samples in the database.
- Pie charts reporting different fractions of allergen components, and so on.
- Statistical report (pdf document).
- When this area has least requirements all possible ideas of how data can be statistically presented, should be documented for further investigation.

4 Design

Developing an application the design process is usually broken down into different levels or principles. The architectural design describes what kind of components the application is be build from, and how these components are organized in relation to each other and also how the communication between them occur. Next step is to make the design on a more detailed level. This is the implementation design, and describes how the different components is structured and implemented.

The overall design solves the requirements in such a way that the application is optimal both in functionality and in usability.
4.1 Architectural Design

The current requirements and the future ideas for the application are affecting the architectural design of the application. One of the major requirements is the storage of data into one place, no matter where the user uploading the data is located in the world. The best way to do this is to use a centralized database.

When having a centralized database, with users or clients working with it internally, it is enough for the application to use the ADO.NET technology. In contrast, working externally the database must be supervised by a service, web service, getting request from the users via the internet using http protocol. In this case a Window Communication Foundation (WCF) Service is a perfect solution, which gives the advantage of having one service to work both internally using tcp protocol and externally using http protocol. How the service is distributed is just a configuration setting, the service is always the same.

A database solution also gives the benefit of being searchable based on different search criteria. This feature is handy when data is going to be visualized.

Using a service it cannot work alone, it must be hosted, and that is be done in different ways, all depending on how the service is used. During the development of the service the hosting is best done using a console application host giving the advantage of easy to start, stop and edit. When the service is tested on a local computer the service is best hosted by the windows services, this way the computer’s operating system takes care of the service and manages its start and stop. Editing is much more troublesome when it now involves uninstallation and reinstallation of the service. Finally, when the service is finalized the service is best hosted by an IIS server. An IIS server brings a lot of features to the service like; connection to the internet, security, and much more.

The users connect to the service host, also called service provider, by a client GUI developed with Microsoft’s Windows Presentation Foundation (WPF). WPF is a computer-software graphical subsystem for rendering user interfaces in windows based applications. With the GUI users are able to upload data to the database and visualize data stored in the database. The GUI also gives the users some statistical information of the database. The component connecting to the service host/provider is called the service consumer, and it can be like in this case an application GUI, but it can also be another service or a web site.

4.2 Implementation Design

All components from the architectural design, has their own individual design, optimizing their requirements and functions to full fill their requirements for the application.

Starting with the database design [11], this process itself can be further divided into three main steps; conceptual design, logical design and physical design. The conceptual design is about founding “real world objects” used in the organization, which is going to be stored in the database. In this process an Entity-Relationship (ER) model is used to clarify objects and their relationship types. The logic design is how the conceptual schema is translated to a logical schema that is stored by the Database Management System (DBMS). In this project the database is a relational database, and then the logical schema is a relational schema describing all tables in the database. Finally the physical design is managed mainly by the DBMS. In this design things like what data structures are used for data storage and are the database in need of a indexing for speeding up the requests. This part is handled by the DBMS.
Developing a WCF Service, each service is best designed by using two separate files. One is an interface containing the declarations of the service functions, called OperationContracts, and declarations of classes used for data transfer, called DataContracts. The second is an implementation file, containing all implementations of the OperationContracts. The data that is being transfer between the host and the consumer is actually transformed into XML before it is sent, and also the data that is received is also XML. The WCF service is made into a library, which is built into a dll file. The WCF service task is to be the connection between the client and the database, and that is why the service has some settings of the database, like the connection string, used for the database connection.

The service provider hosting the service is as mentioned before of different kinds. Note that, the service is always the same; it is just the way to host it that differs. The host has a reference to the service dll file, and sets up a service object and opens it for requests. This is why the host has to have a configuration file with all settings a service and its clients need to communicate to each other. In the configuration file information about the service like, its address i.e. web address and its port number, the binding i.e. http or net.tcp, and contract i.e. the service and its classes and functions, and also a lot of other information like logging, service-configurations, data sizes, are set.

The client GUI design is constructed to handle the requirements it is set to meet. The smaller features like storing projects, investigators, rules and so on are solved by using forms. The larger features like upload of csv files, and the visualization of data requires larger parts of the GUI. Even the statistical part needs its own structure in the GUI. To be able to handle both upload and visualization at the same time, they need their own working areas, windows, and this is why a structure with tabs is in handy solution. Each tab representing its feature; upload, visualization, and statistics.

This GUI also has a menu containing different options to apply to the application. The menu contains options for adding projects, investigators, rules, showing rules and the export of data to a csv file.

5 Implementation

The tool used for implementation is Microsoft’s Visual Studio 2010. Microsoft Visual Studio is an integrated development environment (IDE). This tool helps the developer with a lot during the development. For more basic information about the tool, take a look at Wikipedia/Microsoft Visual Studio.

The programming language used for this project is C#, and the Framework used is the latest version from Microsoft, .NET 4.

Working with Microsoft SQL Servers, a useful tool is the SQL Server Management Studio (SSMS). This tool is used during the project, to make scripts creating the database; tables, foreign keys relations, stored procedures, and also when executed queries against the database, both when data is looked at in the database but also when stored procedures are tested.

The communication between the database and the GUI is done with a WCF service and its provider, the host. C# is used also here.

The GUI is made with the technology WPF, which contains the programming markup language called XAML, eXtensible Application Markup Language. XAML and C# is used in the development of the GUI, and XAML is very effective “programming language” for creating GUIs.
When the design process has been examined, the big picture of how the application is structured is given. The design process is not static, because during the implementation process it happens that it turns out that another design or solutions might work better in combination with other solution, or the project customer had some new ideas. This is an agile process. The development process is here presented and the solutions that are chosen to full fill the requirements.

6.1 Architecture

The structure and the solution for each of the components in the application are here presented component by component. All figures used in this chapter are presented in the appendix A in the end of this report, as a consequence of their sizes.

6.1.1 Database

In the development of the database, prototype name ISACdb, the first step is to go through the conceptual design. The real world objects most interesting are; Investigator, Project, Patient, Sample, Spot, Allergen Component, Patient Type, Symptom, Rule and Rule Allergen Component. Next step is to find relations between these objects, and the relations are presented in figure 2, showing the ER diagram. The last step is to find all attributes that describes each object, which is the information that is saved to the database.

The ER diagram shows that an Investigator represents one or more Projects, a Project contains one or more Patients, a Patient has one Sample, and a Sample contains many Spots. Each Spot represents an Allergy Component and its value. The first approach is to reference each Spot to an Allergen Component. But the gain of separating the Spot from its Allergen Component name, making a reference to a table representing the Allergen Component is insignificant, and it is excluded. The Patient also has a Patient Type, which for the moment can be either of two kinds, patient or control. It has been some indication of that more types is maybe used in the future, but for the moment just
two different kinds are available. If this changes in the future, the application is able to add, and maybe even edit and remove patient types with a form. As soon as the possibility to add something to the database, its siblings edit and remove must be allowed for, (or not).

The Patient has a third relation to one or many Symptoms. The other way is also true; a Symptom can also belong to one or many Patients. When a situation like that happens, the need of a third object is added between Patient and Symptom and that object is named Patient Symptom. Patient Symptom keeps track of all relations between Patients and their Symptoms.

Some of the objects are for the moment not as important as others, e.g. Patient Type, Patient Symptom and Symptom. This is because it has not been decided how to transfer this data in to the application. It is not part of the csv file, so there are still some issues to be considered.

Looking at the relation between Sample and Spot, these two entities are presented in separated tables. The reason is that for the moment each Sample has 103 different spots, but in the future this will most likely change to 111 or more. Having two separated tables, a change in number of spots on the chip is not affecting the database design, it can handle all cases.

The last part of the ER diagram is a stand-alone structure, Rules and Rule Allergen Components. A Rules (the name Rule is already taken by the Microsoft SQL Express application, this is why Rule becomes Rules, and notes “a Rules”) contains one or more Rule Allergen Components. A Rules is an object that groups one or more allergen components, by that facilitates the comparison between them. One more thing to notice is that, by storing the Rules in the database, all Rules are available to all users. If the meaning is that each user has its own Rules, then the Rules must be either stored locally on the user’s computer or the Rules must be related to some User object, which we do not have in this database.

After all objects, relationship types and attributes are found, this ER diagram, the conceptual schema, is translated into tables. Using a relational database, the relations schema is represented by tables, figure 3.

![Figure 3. Relational Schema](image)

Figure 3. Relational Schema. This project will be using a relational database, then the logical schema is a relational schema describing all tables in the database.

Working with a database, the best way to do this is by stored procedures (SPs). SPs are predefined queries, procedures or functions, which are called when a request to the database is made. Using SP gives some benefits, SPs are stored in the SQL server, less communication between DB and client, the SP is pre-compiled, reducing the run time of the query, SPs can also be optimized,
query optimization, by the DBMS. Using SPs also contributes to merging code into functions and modules, making the solution nicer and more reusable. Because of the advantage of using SPs, these are also created in this solution. In the project nine SPs are created, seven Add-SPs, and two Get-SPs. In the project a number of select queries are created but they are never translated in to SPs.

When a change is done to a table, either adding or editing data, and an error occurs in the process the change is never done at all, i.e. the table remains unchanged. During the upload of a csv file, three different tables are changed at the same time, and if an error occurs here the risk of ending up with inconsistent data is major. This is why the service’s function managing this transaction is using a transaction command to group this three SPs; AddPatient, AddSample and AddSpot, into one over-all transaction. Now, if an error happens during one of these SPs the whole transaction is rolled back and no change is done at all.

When querying a database table with a huge number of entities/posts, or if the query is really complex, it can take a fair amount of time to do the query. To reduce or minimize the time for querying, some actions can be taken. One is to change the physical design of the database by adding an index table, based on the query that is being optimized like a register in alphabetical order. A second action is to work with Query Optimization Techniques. No time will be scheduled to look at these issues in this project. The prospects for this application is that the number of patients in the future can be substantial, if or when this happens actions above must be taken.

Adding data to a database it has to be considered, if the data being stored is put in manually by the user by hand i.e. typing a value in a textbox, or does the user has to select a value from a combobox, with already predefined values. The last option is the one to be preferred, then no misspelling can be done and the misinterpretation is minimized, i.e. is it the Netherlands or is it Holland? It can be mistakes even when comboboxes are used but the risk is minimized. If these values are used as database search criteria, then it is important that these values are correct. Otherwise the result from a search is missing some data in the request result.

6.1.2 The WCF Service

The developing process of the service is iterative, meaning; add a new operation contact or a new data contract each time it is needed. In this project the WCF service, is responsible for handling the data transfer to and from the database. Because of this a configuration file for the service is set, with information about the connection string, information used to connect to the database. This connection string can be a default or a more specific one with more specific information such as usernames, passwords and so on. The default connection string is used during the development process, but a change is necessary when it is time for staging the application on another computer, where different user rights are setup.

Depending on the number of tasks the service is managing, the number of operations is different. The demo developed in this project has 18 different operations. The operations the service has is adding information/data to the database; adding investigator information, adding project information, upload data from a csv file and storing created Rules. Meanwhile during the visualization the service is sending requested data from the database to the client. In the client a number of query criteria are used to request for different data. Data can be requested based on investigator and its projects, or just projects alone, or as a last request based on the country of the patients.
During testing of the application, some features of the service have to be more carefully investigated. The features have to do with the size of the data that is being sent from the service to the client, and also the size of the data that is received by the service from the client. Because of security reason a service can not send or receive unlimited large data sets. How large a data set can be is managed in the configuration file for the object hosting the service.

The default approach a service sends data is buffered. When this is the case the client configuration sets the maxReceivedMessageSize and maxItemsInObjectGraph, to larger values then the default values. How large these values should be is difficult to approximate, especially when the size of data being sent had no limit. The solution is to set the number large enough, and hopefully that will work. When the service receives data, the host has its maxItemsInObjectGraph set to a larger value than default.

The problem of sending and receiving large data set is not yet solved. Maybe there is a solution to the problem by using steamed data transfer instead of the default buffered. If this strategy is used instead, some configuration changes have to be done, and also some changes in the code have to be made. Because the project time has run out, this strategy cannot be implemented and tested.

The WCF service is created like a library, which package the service into a dll file when built. This dll file is being added and referenced to the host that is going to host the service.

6.1.3 Service Provider – the Host

A WCF service can never be stand-alone; it has to be hosted by another component, called service provider. The way the service is hosted can change, and depends on how the service is going to be used. The service provider’s task is to receive requests from the client, also called service consumer when used together with a service. The service provider contains the configuration settings for the service. A part of the settings, the endpoint of the service is called the “ABC” of the service. “A” stands for the address of the service, which is the URI the provider is listening to. During the development the address is http://localhost:9000/ISACDataService/. “B” stands for the binding used by the service. If the client is communicating through internet the binding is http-related, or if the client is communicating internally, the binding is net.tcp-related. The advantage of using a WCF service is that both communications are allowed at the same time, just set up both ways in the host’s configuration file. In the project the binding used is a wsHttpBinding. “C” stands for the contract, and in this project the contract that is being used is also the one that has been created in this project, ISAC ServiceLibrary.IISACDataService.

The service is hosted in two different ways in this project. During the development of the application the service is hosted by a console application host. This gives the benefits of changes made to the service can fast be rebuilt and reintegrated into the project solution. When the application is ready for staging on another computer, for testing and validation, the service is hosted with Windows Services on the demo computer. This way the service is managed by the computer’s own service handling. If the application in the future will be used in production, then the service provider is usually an IIS.

6.1.4 Service Consumer – the WPF Client

To facilitate the use of the service and the database, a graphical user interface (GUI) is created to communicate with the service provider. This part is quite extensive giving the GUI its own chapter, see chapter 7, where a thorough description of the GUI is given.
The setting for the client component, the service consumer, is similar to the service provider. It also has endpoint settings with the ABC given, with the corresponding values, and also values for the maxReceivedMessageSize and maxItemsInObjectGraph.

6.1.5 Setup and Installation

After the application is developed to fulfill all requirements it has to be built into a setup and installation project. This is done by adding one setup project for the WCF service provider, and one setup project for the GUI, the WCF service consumer.

Before the installation of the service and the client, the Database Management System (DBMS), Microsoft SQL Server Express, is installed on the staging computer. Next, the software SQL Server Management Studio (SSMS) is installed. This SSMS software helps the developer with many important database related things. For example with SSMS the creation of the script making the database, ISACdb, the tables, foreign keys and the stored procedures is made. When the script is complete, this script is also run by the SSMS software. As a final modification to the created ISACdb database a database user is setup with the execution rights of the stored procedures getting and setting data to the database.

The ISACdb is now up running; next the installation of the ISAC service is done. The installation of the ISAC service works well and it is now installed as a Windows Service and is automatically started when the computer is turned on.

Because of the projects time limit the client is never installed on the staging computer. The client will be installed on the staging computer but this will happen after the project report is written. Instead the testing and evaluation is done on the development computer.

6.1.6 Application Testing and Evaluation

The installation process of the whole application (database, WCF service and WPF client) on the staging computer is complete just when this report is nearly finished. So, testing and evaluation is never made on the staging computer for this report. Instead some testing and evaluation is done on the developing computer, which raised some questions and new ideas.

7 Graphical User Interface

The requirements for the GUI consist of three main parts; the storing of data to the database, the visualization of data stored in the database and a part presenting database statistics. Beyond this the GUI also has the features of adding investigator information, project information, create visualization rules, and select rules for visualization.

Working with data in the GUI, it is possible to both upload data to the database, and at the same time visualizes data. Solving these requirements, the solution is to make two different tabs, one for each feature, figure 4. When visualizing data in one tab, a switch can be done to the upload tab where an upload of csv file can take place, and later that data can be visualized in the visualization grid. A third future tab is representing the database statistics.

The application GUI has a menu along the top of the application window. This menu list contains three different groups; File, Edit and Rules. Under the menu list there is an area where forms for adding investigator information, project information, new rules, and selecting rules for visualization of grouped allergen components, is placed, figure 7. The solution made here is to create forms
connected to the application window, instead of using a pop-up window with the form. The benefit of this is that the users do not have to handle more windows, the user is not locked with the pop-up window, and the user can easily switch to another tabs, or open up another form to create a new investigator. This situation is impossible to do with locked pop-up windows, or the user may have two separate windows to handle, which can be irritating.

Figure 4. GUI with tabs. The GUI is divided into tabs which separates the uploading area from the visualization area. This is a beneficial solution, making it possible to switch between uploading and visualization, that can be done at the same time.

The forms are created from a base class, BaseForm, which derived from the UserControl class. The BaseForm has an event that removes the forms from being visible. The AddNewRule form has a dynamic solution, where the number of allergen components can vary. The selection of an allergen component is done from a combobox, and each time an allergen component is selected a new combobox is opened. This way the number of allergen components can be dynamic. The forms adding information about investigators and projects, AddNewInvestigator respectively AddNewProject, consists of textboxes which content is stored into the database. The form of ShowRule contains a combobox populated with the rules stored in the database. Select a rule and the rule with related allergen components shows up in the visualization grid.

In the application a lot of comboboxes are used. The population of the comboboxes is done from data coming from text files or from database requests. One example is the combobox containing countries in the upload tab, where the population of country strings is done from a text file. Another example is the combobox also in the upload tab, containing all projects that are stored in the database. This combobox has a variable number of values depending on the number of projects stored in the database. It is also important that this combobox remains updated all the time. When a project is added to the database this combobox must be updated. Having a combobox limiting both the number of and the value of the possible options. This is beneficial when the data selected is
stored to the database, and also later is used for database requests. This way misspelling and languages mixing are eliminated, which will affect the results of database requests.

From the menu the feature of saving the data in the visualization grid back to a csv file can be made. This way data from different ISAC projects can be merging into a new csv file and used by another analyzing tool.

7.1 Storing ImmunoCAP® ISAC data

The application is uploading data in the form of csv files containing results from ImmunoCAP ISAC instruments. The selection of the csv file is done by using an openfiledialogbox, figure 5, which opens the csv file and the file is parsed to get all data of interest. Not all data in the csv file will be stored in the database. The information in the file constitutes of sample information, formerly called the head data, and sample result, called the tail data. The data in the head for each sample is presented in a grid, where also more information about the patient can be added, like: patient’s date of birth, sample date, sample run date, patient country, postal code, gender, patient type, diagnosis and symptoms, figure 6. The three last entries are never implemented because of lack of time. The sample result, the tail, is not presented in the grid, because this data is not editable. Each uploaded csv file is connected to a project, one project from the database is selected, and the projects are presented in a combobox. The combobox is filled with projects when the application starts up, and the combobox is updated each time a new project is added to the database. To make the combobox up to date the application makes a request to the service to send the projects that exists in the database.

Figure 5. Selection fo CSV file. The CSV file that is going to be uploaded is selected through an Open File Dialog Box.

Because each uploaded csv file is connected to a project, a new project is created from a form. In the menu list a selection of add new project is done, which opens up a form sliding down at the top
of the application window. All projects are connected to an investigator. The form for creating an investigator is selected form the menu list and the form are placed in the area beneath the menu list, figure 7.

Figure 6. CSV file presentation in data grid. The data presented in the data grid are sample information and patient information, added manually by comboboxes.

Figure 7. Add New Project and Add New Investigator forms. The forms used for adding new projects and investigators are presented in the main windows in an area just under the menu list.
When the csv file is uploaded to the database it can be visualized, which is the next major part of the application.

7.2 Visualizing and Analyzing ImmunoCAP® ISAC data

The visualization of the data is done within the visualization tab, figure 8. For the moment the patient information that can be seen in the visualization grid are; sample name, patient label, country and gender, and all values for all allergen components, the spots. The allergen components are organized into four different levels. The first level contains three main groups; food components, cross-reactive components, and airborne components. The second and three levels contain different sources where the allergen components can be found, and the fourth and last level is the allergen component itself. The information setting up the allergen component header with four levels comes from an xml file. The xml file is parsed and the header is made based on that information. Using an xml file is beneficial because it is a dynamic solution. A change to the header, either rearranging the levels or adding new allergen components, is easily done by editing the xml file. The number of allergen components in the grid is 132, and number of spots on the chip is 103, which results in empty columns. In the future more allergen components will be added to the chip, which is why the total number of components is 132.

![Visualization grid](image)

**Figure 8.** The Visualization grid. An empty visualization grid, before the selection of data is done. The selection criteria is made from comboboxes at the bottom of the window. Each time a selection is made a new combobox is shown for further criteria.

The default way the data is visualized is quite simple; basically just present each sample with four different pre-selected patient information, followed with all values for each allergen component. The number of data in a sample is large, that is why making different coloring value intervals will distinctly increase the ease of looking at data. A set of four different color intervals are set up as default; [0,
The intervals are determined by experience of the personal at Phadia. In the future an away for the users to create own color intervals should be add.

Sorting an allergen components values in a column from high at the top to low at the bottom, also facilitate the analysis. This feature is implemented, and a user can make a column sort by pressing the allergen component label. The sorting algorithm used is a bubble sort. The sorting is done based on the elements in a column, but the elements that are rearranged are whole rows. When the sorting of the list containing all patient data is done, a redrawing of the visualization grid is done. That is why the sorting becomes a quite heavy process.

The selection of samples for visualization is done by querying the database with a number of predefined queries. To start with, three different sets of query criteria are created; investigator, project, and country of patient. Which selection criteria that is used, is selected from a number of comboboxes, figures 8 and 9.

**Figure 9.** The Visualization grid showing data. Once all criteria are made the data can be visualized in the visualization grid. The default coloring is white, yellow, orange and red. Because the grid has 132 different allergen components and the chip only contains 103 components some columns are empty.

There is also a special selection from the database, the selection of the whole database. The number of elements in the database can be very large, that is why a warning or a pre-notation popup window is first showed, giving the number of patients in the database. The user gets a choice to continue or abort the visualization of the whole database, which can take a while if the number is large.
7.3 Statistics of database containing ImmunoCAP® ISAC data

Having a database with a large number of data opens up the possibility to create statistics of the database. Implementation of database statistics is never done because of time restriction. Instead different number of valuable statistics is investigated.

The total number of patients in the database is of course basic valuable information. Statistics of each allergen component showing a pie chart of the percentage of each value interval, figure 10.

![Figure 10. Pie Chart distribution. The distribution of different allergen components in the database can be visualized as a pie chart. A lot of information and statistics from the database can be visualized by pie charts.](image)

Pie charts can also visualize the distribution of patient types, patient symptoms and so on. The level of statistical presentation by this application is basic. This application is not going to be a statistical program; there are a lot of other applications for that. The statistics in this application is there to give the users information of what kind of data and the number of data the database is containing. This information can in the future be something that attracts new users to the application.

Calculating statistics is a process that can take a lot of time, especially if the number of data is large. So, the question is if the statistics is going to be a part of the application, or if it is going to be something that is outside the application. Smaller statistics can be part of the application, and large statistics can be a stand-alone part just working with database statistics. It can for example be a script going through the database calculating a number of pre-determined statistics. The results can be presented in a pdf report, or a newsletter.

8 Verification

The verification process is to investigate if the application is meeting the requirements, and if the application is function in a flawless manner. The number of processes that is verified here are; (1) the upload process, to check that right information is saved to the database, (2) that right data is shown in the visualization grid after selection, (3) that the sorting algorithm is sorting in a correct way and that the following rearrangement of patients is correct and (4) that the export back to a csv file from visualization grid is correct. There can be even more and thorough verification processes done, but the project’s timeframe limits the extent of the verification process.

The uploading process seems to work very well. By comparing the csv file with the data saved to the database, using the SQL Server Management Studio, no differences is discovered.

The visualization of data is almost correct, right data, and also right number of patients is in the project. The problem is that the sample names in the csv file, and in the database is, for example “9I51324_2”, but when it is visualized, the sample name is “9I513242”. The underscore has been omitted. But when the visualization grid is exported back to a csv file, the underscore is back again. Strange error, wonder if it is just a configuration error.

The sorting function works correct when testing and comparing before and after. Both the sorting and rearrangement of rows are correct.
One thing that needs some work in the application is how the file structure is set up. In the application some files, one xml file for example, are used to build up the visualization header and some other text files to fill comboboxes. When a file is uploaded, the position of where in the file structure the application is, is changed, which will affect the visualization redrawing, when sorting or selecting new data. This must be taken care of. The solution right now is to use absolute paths to files used in the application. This will not work for the application used for staging on another computer.

9 Evaluation

A minor evaluation of the application has been done by Phadia employees, and resulted in following comments; the upload section, adding data manually using comboboxes and datepickers take too much time and effort, and should instead be set in the csv data file. By adding some extra data columns for sample date, run date, date of birth, and so on, that will eliminate the manually setting of data.

The use of patient types, diagnosis and symptoms, which now is not implemented, should be added to the application as soon as possible. Before that is done some thorough reviews on how this is best done should be analyzed. Where should the extra information come from and the numbers of symptoms, and so on.

The visualization part also had some comments, for example, now the application cannot export rules to a csv file, just the regular allergy components, that was one thing that would be useful. Also the possibility to select multiple data sets should be a feature in the future.

10 Future Development

The future work will be to implement statistics of the database. To decide what kind of statistics that will be most valuable and how the statistics are going to be done, in the application or outside the application, or maybe both.

After some testing and evaluation by Phadia employees some features to the uploading section was desirable. Adding data manually to the patients during upload is too much work and should instead be added to the csv file. That is sample date, run date, patient’s date of birth, country, gender, and also symptoms and diagnosis.

Also the visualization section had some features to be changed or added. Selection of data should be easier to do, especially selection of multiple data sets, which cannot be done in the application today. Multiple selections of data sets can be solved with a checkbox feature, where the user can check which data sets to show. When the amount of data increases the way to work with it also changes, that is one thing to observe.

The amount of data in the database will increase all the time, which is for sure. That will affect the service, both in capacity and speed. The service can be modified to instead of buffered transfer, use steamed transfer, which will increase its performance.

11 Final Conclusions

This thesis resulted in a demo application for handling ImmunoCAP® ISAC data; storing csv data files to a database, the ISACdb, a database designed to store ISAC data, and visualization of ISAC data.
from the database. Hopefully this demo will be found useful for analyzing ISAC data. Maybe this will be a first step of a greater application used by Phadia employees and Phadia customers around the world.

By introducing rules for grouping allergen components, the analysis of comparing allergen components to each other is facilitated, resulting in founding interesting relations between them.

In working with the project I have developed an advanced knowledge of database design, Microsoft’s Windows Communication Foundation (WCF) building services and Microsoft’s Windows Presentation Foundation (WPF) building Graphical User Interface (GUI). Also the knowledge of putting all this technologies together into one working application, learning about difficulties and benefits when developing an application. During the project I have also get knowledge about Phadia as a company, its research and products.

Finally I would like to thank Annica Önell at Phadia for letting me do this thesis project, for guidance, help and encouragement, and Mikael Alfven at Phadia for giving me feedback when I got stuck in different problems.

12 References

8. Published by Oxford PharmaGenesis™ Ltd, 1 Tubney Warren Barns, Tubney, Oxford OX13 5QJ on behalf of Fizer Endocrine Care. © 2007 Pfizer Inc.; KIGS medical outcomes;
9. Medical Outcomes website; About KIMS; http://www.medicaloutcomes.pfizer.com/MedicalOutcome.aspx?id=734; 30 may 2011
10. Allergome website; Allergome Home; http://www.allergome.org; 4 may 2011
Appendix A  Application Figures

Figure 11. The creation of a Rule. To create a rule a number of allergen components must be selected. The number of components are dynamic, each time a component is selected a new combobox is shown.

Figure 12. Adding a Rule to the Visualization grid. To add a rule to the grid the show rule form must be selected. Then from a combobox all rules in the database are presented. Select a rule and the rule will be added to the visualization grid.
**Figure 13.** Export data in Visualization grid. All data in the visualization grid can be exported back to a CVS file. Maybe the data can be analysed with another tool that way.
Figure 14. ISACDataProject Solution. This is a representation of the whole project. The project contains of six different subprojects. The ISACServiceLibrary containing the service implementation, with the interface file, IISACDataService.cs and implementation file ISACDataService.cs. The ISACConsoleHost for the service hosting during the development process. The ISACWindowsServiceHost for the hosting during the staging. The ISACClient containing all files building the GUI. In this project there is a reference to the service, Service Reference / ServiceReference. The PresentationLogic contains three grids for the visualization of sample head, HeadGrid.xaml, for sample tail, TailGrid.xaml, and for the rules, RuleGrid.xaml. In the Forms catalog there is one BaseForm.cs which is the base class for the other forms. In the catalog Text files there are five files, to xmlfiles and three text files, with data for building content of comboboxes and grid headers. The last two projects are the Setup projects for the Service, SetupISACService, and the Client, SetupISACClient.