

European Polytechnical Institute, Ltd. in Kunovice

Study field: Electronic computers

**IMPLEMENTATION OF SOFTWARE
ENGINEERING IN A PROJECT OF A WEB
DATABASE**

(Bachelor thesis)

Author: Dušan JANOVAČ
Head of BT: Ing. Jindřich PETRUCHA, PhDr.

Kunovice, May 2006



Evropský polytechnický institut, s.r.o.

Osvobození 699, 686 04 Kunovice
☎ a fax: 572549018, 548035, e-mail: epi@vos.cz
<http://www.vos.cz/epi>

Student(ka)
Dušan Janováč
Zábrani 1369
763 61 Napajedla

VÁŠ DOPIS ZNAČKY / ZE DNE

NAŠE ZNAČKA
BP_EP05/06

ZODP. VEDOUCÍ/VYŘÍZUJE
Ing. Dušek/Čápková

KUNOVICE
5.10.2005

Zadání bakalářské práce

Vážený studente, vážená studentko,

jako téma Vaší bakalářské práce ve studiu oboru Elektronické počítače Vám
zadávám


Aplikace metodiky softwarového inženýrství v projektu webové databáze

- Osnova:
1. Analýza současného stavu metodiky UML
 2. Praktické nástroje používané pro model UML
 3. Analýza datové struktury pro aplikaci webové databáze
 4. Návrh datového modelu pro firmu hřebčín Napajedla
 5. Implementace do prostředí www stránek
 6. Ekonomické zhodnocení projektu

Bakalářská práce bude zpracována pro: Hřebčín Napajedla

Tento dokument je součástí Vaší bakalářské práce.

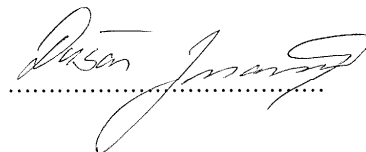
S pozdravem


Ing. Oldřich Kratochvíl, Dr.h.c.
rektor

Evropský polytechnický institut
s. r. o.
Osvobození 699
686 04 KUNOVICE

I declare that I myself worked out my bachelor thesis headed by Ing. Jindřich PETRUCHA, PhDr. and I mentioned all used literary and technical sources in the bibliography.

Kunovice, May 2006

A handwritten signature in black ink, appearing to read "Jindřich Petrucha", written over a horizontal dotted line.

I would like to thank the head of my bachelor thesis, Ing. Jindřich PETRUCHA, PhDr. for his patience, specialist leading, professional guidance, help, constructive comments and retrieval of other themes and information.

Kunovice, May 2006
Dušan Janovác

Content:

INTRODUCTION	7
1 THEORY OF DATABASE SYSTEMS	8
1.1 Development of the database systems	8
1.2 Database creation	9
1.3 Relational database	10
1.4 Terminology of relational databases	11
1.5 Normal forms	14
1.5.1 First normal form	14
1.5.2 Second normal form.....	15
1.5.3 Third normal form.....	16
1.6 SQL language	17
2 THE UML LANGUAGE THEORY	18
2.1 The UML notation	18
2.2 Semantics of the UML language	19
2.3 The OCL language.....	19
2.4 Specifications of exchange formats	19
2.5 The UML diagrams.....	20
2.6 Diagram of a model of usage	20
3 ANALYSIS OF THE STUD FARM NPAJEDLA, INC.	24
3.1 Communication with user	24
3.2 Analysis of paper database.....	25
3.3 Analysis of paper documentation.....	25
3.3.1 Current documentation.....	25
3.3.2 DFD diagrams of current system of records.....	26
3.4 Determination of the ways of data gaining	27
3.5 Conversations	28
3.6 Creation of table structure	29
3.7 Names of tables	30
3.8 Designed database.....	30
3.9 Designed names of entities	31
3.10 Sample of table in MySQL creation	35
3.11 Prototyping	36
4 PHP SCRIPTS + MySQL	37
4.1 Summary of all scripts	37
4.2 index.php.....	39
4.3 evi_*.php.....	39
4.4 date_decode.php.....	40
4.5 pripousteni.php.....	40
4.6 fn.php	41
5 DESCRIPTION OF A MODULE OF HORSES' REGISTRATION DATABASE	42
5.1 Introductory page	42
5.2 Registration	43
5.3 Surveys	47

5.4	Monitoring	50
5.5	History.....	50
5.6	Testing	51
5.7	Implementation of the project.....	52
6	ECONOMIC EVALUATION OF THE THESIS	53
	CONCLUSION	54
	SUMMARY	55
	BIBLIOGRAPHY	56
	LIST OF USED ABBREVIATIONS.....	57

INTRODUCTION

I would like to acquaint readers with the theme of my bachelor thesis: Record of horses in the stud farm Napajedla. On the basis of theoretical knowledge gained at school I am going to apply some parts of the UML (Unified Modeling Language) methodics in the creation of my bachelor thesis. According to me an important part is the part of the analysis in which advantages of the UML are shown in a graphic way.

As the title indicates the scope of my thesis will be an implementation of the database information system in electronic form which should replace paper records. The aim of my project is not only the replacement of paper cardfile for electronic one but on the basis of primary entries is to optimize the running of the stud farm.

It is a database project and for that reason in the next part the theories of database systems, which are applied in my bachelor's thesis, will be explained. It is important for the reader to be familiarized with them.

1 THEORY OF DATABASE SYSTEMS

The name database is familiar to all people in these days. People have had a need to record and gather information since earliest times. The whole modern society is put into database systems, from the registration of citizens, through new created health care, economy, school system, banks, research to a network of mobile phones.

In next lines of my thesis I am going to acquaint you with the SQL (Structured Query Language) language. I am going to introduce some basic terms from the database system field but also some concrete features of the SQL language, as tools for creating the database or manipulation with the data. The language SQL is a standard tool from which the most of the used database systems original and these standards are respected at different levels.

1.1 Development of the database systems

Now we are going to go back to about 40 years ago. In the 60s of the last century some well-known terms were created, e.g. *database*, *entity*, *attribute*, and *entity relation*. I would like to mention about these terms.

We can imagine the database as a collection of data which serves for an exact copy of our real world (e.g. record of a school library, a store of metallurgical material, record of employees etc.). Entity can be an element of the real world (e.g. machine, computer, school subject, town) which is described with its typical characteristics (features). These features are usually determined as attributes (e.g. name, surname, marital status, salary, weight).

The other important term in the world of database is an entity relation. Individual entities which are pieces of elements of the real world have certain relation among themselves. For example, each person has his personal details kept in a municipal government, in a department of identification cards. This is relation of type 1:1. Another type is 1:N relation which is of a piece in reality one stud farm can own more horses, but one horse can have only one owner. The last type of relation is M:N. There is no limitation and for example, a university student can enroll several subjects and at the same time the subject can be enrolled by several students.

In this era a new term is created – *database model*. It was established by mathematicians as a means for describing the database. At the beginning, two types of models were used: *hierarchical* (based on modelling hierarchy among the entities in relations of subordination and superiority) and the *network* (it comes out of graph theory, knots in the graph correspond to entities and oriented edges define relations among the entities). In the 70s actual database models showed as insufficient (new problems arose with the realization and also with implementation of the relation M:A) and that is why the *relational model* was created, became a standard and it is used nowadays. [5]

1.2 Database creation

If we define database as an information store in which the fastest, the most reliable, the perfect access to information is required, then a database creation is similar to building of such a store. We can choose the following procedures in creating it:

- Copying and modification of the existing database;
- complete creation of a new database. [2]

Although the first procedure may seem as the fastest and the most feasible, after some time it can be ineffective and incomplete. Especially when a nonstandard requirement appears in the database. On the other hand the second procedure offers absolute freedom in creating a general structure of the database. We are not dependent on a certain delineated framework and therefore we can shape it.

Though choosing the first or the second possibility, people with no practise within this field, will not avoid two basic kinds of problems in database creation: *the application* and *the data*.

Application problems can be characterized as visible deficits in the program. It is especially about unnecessarily complicated forms in data assignment and data modification, confusing or disorganized menu, bad graphic processing, absence of some functions and the like. These problems usually are caused by a low level of experience and knowledge with the database creation, insufficient cooperation with user or some other reasons.

On the other hand *data problems* seem to be imperceptible at the first sight and can arise in a longer time period. There are errors inside the system, e.g. missing, incorrect or repugnant data.

Another typical feature of the data failures, which apt to a spontaneous finish of the application or a closing of the computer, is an instability of the database system. Typical reason of these problems is a bad database design. If database is bad structured, the requirements on information set by an organization cannot be fulfilled.

1.3 Relational database

Now we are going to look closer to a relational model of database. The basic term is relation. Although I could introduce some mathematical definitions, we can imagine relation as a table which consists of columns and lines. The columns correspond to individual features (attributes) of the entity. The data in one line of the table represent contemporary state of the world. Now I have a table OWNER, which contains an entity description of an owner of a horse in the stud farm. The columns of the table are: ID_OWNER, NAME, SURNAME, STREET, TOWN, TOWN CODE, STATE, TELEPHONE and E-MAIL.

Table is a basic building stone for building the whole database. The relation corresponds to the whole table and one concrete line corresponds to an item of the relation. One line is usually called a database record. A set of tables then forms the whole database – a relational scheme. One table describes an entity. Behind the columns of the table we usually choose those attributes of the entity which we want to record or those in which we are interested.

And now the time has come to say something about formation of relational tables. From a long-term viewpoint, formation of well-designed tables is an important key task. If we found an essential mistake in a database system running in a strict process for some time and we discovered that it lies in a bad designed database, it would have serious consequence not only in financial side but also in proper functionality.

1.4 Terminology of relational databases

It is important to understand a terminology of relational databases because the methodology of design will be studied later. Why is terminology important? There are three good reasons why we should learn the terms.

- They are used to express and define specific thoughts and concepts of the relational-database module;
- they are used to define and express their own process of design;
- they are used wherever it is said about relational databases or about the RDBMS (Relational Database Management System).

Data

Data are pieces of information which we store in the database. The data are static because they remain changeless until we change them either by hand or by automatic process.[5]

Information

Information is the data which we prepare to be understandable and applicable when shown and worked with them. Information is dynamic, it changes continually according as we change the data stored in the database. And they can be processed and presented in unlimited number of ways. The important idea is that before the data are turned to a meaningful information they have to be prepared.

Null

Null presents a missing or an unknown value. We need to emphasize that null does not represent a zero or a text chain compound of one or more gaps![5]

Table

According to a relational module the data in the relational database are stored in relations which users see in tables. Every relation consists of n-tic (records) and of attributes (arrays) too.

Tables are the main pillars in database and each of them represents one entity. Series of records and arrays in the table has no importance. Every table contains an array known as primary key which determines all its records. The data in relational database can exist independently of their physical position in a computer. An advantage for users is that they do not have to know a physical data location.

Entity represented by the given table can be an object or an event. If the entity is the object, then the table represents something concrete, e.g. a person, a place, or a thing. Then each object has certain features which we can be later stored as the data. We can transform these data in an unlimited number of ways.

If the entity of table is the event, then the table represents something which happened in a certain time and has certain features to be recorded. These events can be recorded by the data. Later on they are processed as information in the same way like table representing an object.

The table determined to supply information is called a data table and it is the most common type of tables in relational database. The data in that table are dynamic because we work with them permanently and we process them as information.

Another type of a table is a validation table. It stores data which serve for implementation of data integrity. The data in this table are static because they are changed only seldom. [2]

Array

Array is the smallest structure in database. Array represents a feature of the entity corresponding to the table to which it belongs. Arrays are structures which store the data in

reality. The data presented in them can be gained in a whatever configuration we can imagine.

Record

Record represents a unique instance of entity in the table. It consists of arrays in the table regardless if these arrays contain some values or not. Any records in database are identified by a unique value in primary key of the given record.

Keys

Keys are special arrays which determine its meaning in the table. We know several kinds of keys but for us the most important one is a primary and a foreign key.

Primary key is an array which clearly identifies the given record in the array. If the record consists of two or more arrays it is marked as a complex primary key. Primary key is the most important part of the whole table.

Index

Index is a structure which is provided by the RDBMS to improve the data processing. Index has nothing in common with a logical design of the database.

Data integrity

Data integrity determines validity, consistency and accuracy of the data in database. It is one of the most important aspects in process of logical design of the database which cannot be underestimated, overlooked or neglected. This peep causes accumulation of mistakes which are hard to find or discover.

There are four levels of integrity which are implemented in the whole process of designing the database. Three of them are based on different aspects of a database structure. They are determined according to an area they appear in. The fourth type of the data integrity is based on using and understanding the data by an organization. Here are all four types:

- Integrity at the at the level of the table;
- integrity at the level of the array;
- integrity at the level of the relation;
- Business rules. [2]

1.5 Normal forms

The term normal form is used in connection with a well-designed tables. Well-designed tables accomplish three basic normal forms.

1.5.1 First normal form

The first and the simplest normal form (marked as 1NF) says that all attributes are atomic, i.e. further indivisible (in other words, relation cannot be the value). We have e.g. a table ADDRESS which has columns NAME, SURNAME and RESIDENCE. Content of the table corresponds to the real world:

NAME	SURNAME	RESIDENCE
Jan	Novák	Ostravská 16, Praha 17000
Petr	Nový	Chmelnice 23, Zlín 65432

If we wanted to write all workers whose city code corresponds to a specific value, we would get into a trouble or we could not discover it directly. Attribute RESIDENCE is not atomic because it consists of several parts: STREET, NUMBER, TOWN and CITY CODE. A correct design of the table which should respect 1NF will look as follows:

NAME	SURNAME	STREET	NUMBER	TOWN	CITY_CODE
Jan	Novák	Ostravská	16	Praha	17000
Petr	Nový	Chmelnice	23	Zlín	65432

In general we should try to have just one value (of a specific database type) contained in one database item.

1.5.2 Second normal form

A table is 2NF in case it is 1NF and moreover each attribute, which is not the primary key, is completely depended on the key. It means that in a line of the table cannot appear an item which would be depended on one part of the primary key. As the definition says a problem of 2NF relates only to the tables where more than one item is chosen as the primary key. If the table has one column as the primary key, 2NF is trivially accomplished. Now we have a table WORKER which will look as follows:

NUMBER	NAME	SURNAME	WOR_NUMBER	WORKPLACE_NAME
1	Ivan	Kučera	86	workshop
2	Petr	Zátopek	92	office
3	Jan	Marek	45	central

If only NUMBER is chosen as the primary key, it is wrong because a worker's workplace is for sure not depended on a worker's number. Then a pair NUMBER, WORKER'S NUMBER has to be taken as the primary key. But another new problem arises here. Items NAME, SURNAME and WORKPLACE are not completely depended on the pair of the primary key. We can do anything but it is not possible to choose such a primary key to accomplish 2NF. How can we get out of this problem? In general transferring to the table which accomplishes 2NF. It means to separate it into two or more tables in which each table would accomplish 2NF. This disintegration of the tables is professionally called a decomposition of relational scheme. Well-designed tables accomplishing 2NF will look as follows:

NUMBER	NAME	SURNAME	WOR_NUMBER	NUMBER	NAME
1	Ivan	Kučera	86	86	workshop
2	Petr	Zátopek	92	92	office
3	Jan	Marek	45	45	central

We can notice that if the table does not accomplish 2NF, almost in every case it leads into a redundance. Redundance is a phenomenon which accompanies not accomplished 2NF. There is no need to doubt that redundance is undesirable.

1.5.3 Third normal form

Relational tables accomplish the third normal form (3NF) if they accomplish 2NF and no attribute, which is not the primary key, is not transitionally depended on any key. The best is to explain it on an example. We have a table SALARY which looks as follows:

NUMBER	NAME	SURNAME	FUNCTION	SALARY
1	Ivan	Kučera	technician	14500
2	Petr	Zátopek	chief	18000
2	Jan	Marek	manager	15000

Now we forget the fact that this table does not accomplish 2NF which is the basic condition for 3NF. At the first sight we see that concrete attributes NAME, SURNAME and FUNCTION depend on the attribute NUMBER. Next we see that the attribute SALARY is functionally depended on the attribute FUNCTION and if we take into account that NUMBER - FUNCTION and FUNCTION - SALARY we get thanks to transitivity of NUMBER - SALARY. The procedure of getting the tables to 3NF is similar as in 2NF, i.e. we will make a decomposition again: (table FUNCTION and SALARY).

NUMBER	NAME	SURNAME	FUNCTION	SALARY
1	Ivan	Kučera	technician	15000
2	Petr	Zátopek	chief	14500
3	Jan	Marek	manager	18000

In terms of three basic normal forms these two tables are all right. In terms of practicality is appropriate to use a function dial to accomplish the condition that primary key in the tables should be of the shortest length. The best record is as follows:

NUMBER	NAME	SUR.	NUM_FUN	NUMBER	FUNCTION	SALARY
1	Ivan	Kučera	127	121	manager	15000
2	Petr	Zátopek	156	127	technician	14500
3	Jan	Marek	121	156	chief	18000

1.6 The SQL language

History of the SQL language originated in 1970 and 1980. The first standard was received in 1986 (marked as SQL86). After some time some deficits appeared. A renewed version is from 1992 and it is marked as SQL92. This standard is used in an area of relational databases. The language contains tools for database (table) creation and tools for data manipulation (data insertion, actualization, deletion and searching new information).

The SQL belongs to a category of so-called declarative programming languages which means that a code of the SQL language is not written in an independent program but we insert it into another programming language. We can work with the SQL language only when we link to SQL server and in a command line we assign commands of the SQL language directly. [5]

2 THE UML LANGUAGE THEORY

Software engineering methodics represents a set of presentation techniques, methodical procedures, which support a systematic development and a program work maintainance, including planning and managing a program project. Methodics can differ in presentation techniques, methodical procedures and available tools.

Development of consolidated methodics is one of the main aims of an unnamed company which was established by Rumbaugh and Booch. Later on Jacobson with his methodics joined this team and they together made a first design of the UML language.

The UML is meant as a universal standard for record, construction, visual view, but also for an artifact system documentation with software characteristics. Definition of the UML language contains four basic parts:

- Definition of the UML notation (syntax);
- the UML metamodel (semantics);
- the OCL (Object Constraint Language) for a description of other features of the model;
- specification of transmission into changed formats. [6]

2.1 The UML notation

Modelling is usually used in the analysis of requirements as a communication tool between a designer and an assigner. Different models of a planned system are made to be tested, verified and modified. After achieving an agreement between an analyst and the assigner a model can be decomposed and transferred into a target implementary language.

Used models can be of different types. In general a model of software system is defined as all information structured in a specific way. The model can include any gathered information about the system.

Diagram is a grafically depicted view at the model. It describes a specific part of the model by graphic symbols. One diagram rarely describes the whole system in contrast

to the module. Mostly more views are used to describe the system where each of them concentrates on one aspect of the model.

The UML tries to summarize the best known views and it defines eight basic kinds of diagrams which are described below. The UML does not keep from creating other possible views or diagrams but this process is a little bit against unification. On the contrary, the importance of unification is that the same views on the system model would be understandable. UML notation defines how basic views at the model system are written.

2.2 Semantics of the UML language

Records in the UML based on the syntactic rules has to have a strict defined meaning. Description of the UML semantics deals with it. It is divided into four layers which at a different level of abstraction describe features of the UML diagram, including possible extension.

2.3 The OCL language

Not all features of the model can be expressed by a diagram. For example a data system model which deals with record of employees. It is needed to remember different attributes about each employee, e.g. employee's salary. In the same way we must remember hierarchical structure among the employees. It is easy to express requirements on the data structure. It is hard to attach such requirements as „a superior must have a higher salary than his subordinate“. The UML allows applying the OCL language to express these limits. The OCL serves for expressing more complicated integrate limits, description of operational features and it can be after all used also for expressing the semantics of the UML.

2.4 Specifications of exchange formats

A part of the UML is also a specification of exchange formats used for example for transporting the records in the UML among different tools.

2.5 The UML diagrams

The UML documentation does not have to consist only of texts. Two-dimensional diagrams may be created from basic lexical elements. On a sheet we lay particular package of elements and we connect them by connectors. It is possible to use arbitrary elements and connectors. In terms of unification it is appropriate to select a specific package of elements suitable for expressing a purposeful view on a simulated system. For that reason the UML includes a definition of eight types of diagrams, i.e. eight different views at the system model:

- Class diagram;
- use case diagrams;
- sequence diagrams;
- cooperation diagram;
- state chart diagrams;
- activity diagrams;
- component diagrams;
- deployment diagrams. [1]

2.6 Diagram of a model of usage

Because of implementation by PHP (Hypertext processor) program without using the OOP (Object Oriented Programing) version only a usage diagram will be shown – it is used in a first phase of the analysis of development. The main purpose of the diagram of usage will be explained in next paragraphs.

Model of the diagram of usage serves for a basic determination of limits between the system and its surroundings. Diagrams of usage give us a fast conception about the individual system functions but exact procedures, extended and alternative scripts have to be recorded in a text form.

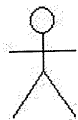
In the UML diagrams we try to record cases of usage, participants and also their relationships. During making a model of usage it is needed to take into account that there are so called secondary participants, i.e. user's roles or cooperative systems for which the

system is determined but also they are needed for its function. When we want a direction of communication, we draw it by oriented arrows. Among the cases of usage can exist relationships. Components of the model are:

- Actor – user's role or a cooperative system;
- system limits –determination of the limit of the system;
- case of the usage – documentation of events on which the system must react;
- communication – relation between the participant and an event of the usage.

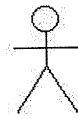
Actors

Record
actor



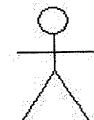
Record actor – responsibility for the data insertion;

system
aministrator



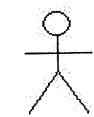
system administrator – responsibility for the program solution;

veterinary
surgeons



veterinary surgeons – responsibility for sanitarian cards of horses;

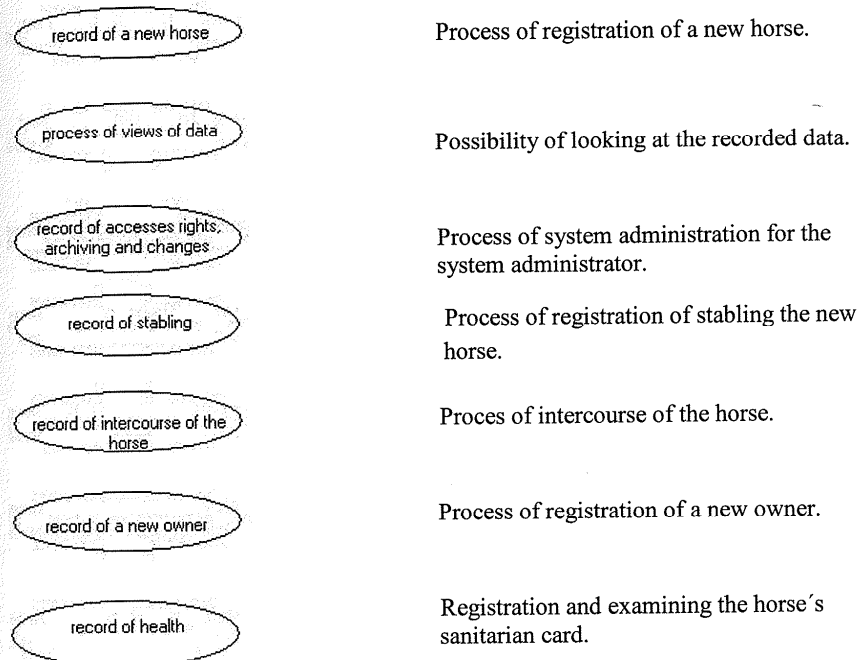
other users



other users – looking at information without specific responsibility.

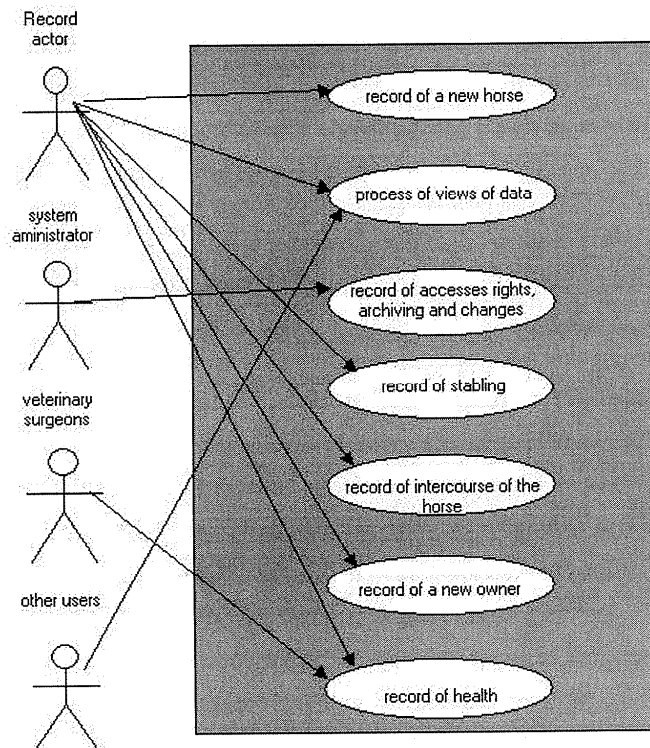
pic. 2.1 Survey of all participants in the stud farm.

Processes and its processing



pic. 2.2 Survey of all processes in the stud farm.

Final diagram of usage



pic. 2.3 Model of diagram of usage.

It is clear from the picture that using the diagram of usage during the project analysis provides a good survey about particular processes and it is possible to follow other parts of the project analysis which will elaborate particular processes into functional parts of the chosen language. In our case we will deal with the implementation by the PHP language which is shown in the final chapters.

3 ANALYSIS OF THE STUD FARM NAPAJEDLA, INC.

Before starting of new database creation it is necessary to make proper analysis of the given problem, and to start building the desired system on its basis. There exist a lot of possibilities how to proceed appropriately when analyzing the given problem. In my thesis I am going to try to suggest one of the procedures.

3.1 Communication with user

Mutual communication with the stud farm personnel was very important source of information for me not only for the database creation. Thanks to it I gained lots of formal and also informal data about the everyday operation of local stud farm. Next important step was to acquaint myself with their current system of records, which was kept only in written form. Director of the Stud Farm Napajedla, Inc. (onward referred to in this text as "stud farm") Mr. MVDr. Filla helped me with this, for what I would like to thank him this way.

Our mutual consultations took place in the stud farm area in almost regular 14-days intervals. Mr. Filla has been elected a director of the stud farm only recently. I myself didn't have any experience with creating the database, and therefore to specify our priorities of electronic records wasn't an obvious matter.

I carefully prepared different questions and clues for each face-to-face encounter, which could clarify the problems of the whole structure of horse records to me. Design of this database proved itself to be very difficult, because none of us had particular idea of what should the final result look like. However, in the time passing, the door to final result was starting to open slowly to us, and the structure of database was getting a clearer form. Here are the clues that I tried to keep to at our sessions with my assigner:

- To prepare the asked questions in advance;
- to create homely atmosphere;
- to keep a pace of conversation;
- to keep control over conversation.

3.2 Analysis of paper database

Paper database is the one that has been used in the stud farm by now. It concerns different forms written by hand. There are the data that were gathered and assorted and kept on the paper.

They who have already made such an analysis would confirm that this way of analysis was a very demanding task. The most important thing was to arrange someone who would describe the paper database itself. The personnel were newly elected, and so I didn't know which paper-evidenced data would they consider to be the most important.

Goal of this analysis is to determine the type of data used in the stud farm. Furthermore, how are the data administered, kept, and used. During this analysis I examined different ways of how the stud farm gathers and presents the data. I used these gained data for predefining of arrays and defining the tables, which should be in the primal structure of database. If our database reveals that the current database is badly designed, we will be able to prevent repeating of the same mistakes in the design of new database. The basic rule that we should remember when designing the database is: *"We mustn't base the structure of new database on the structure of already existing database"*. [2] If we keep this rule, we will avoid unnecessary mistakes and it will enable us to maximize our effort in development.

3.3 Analysis of paper documentation

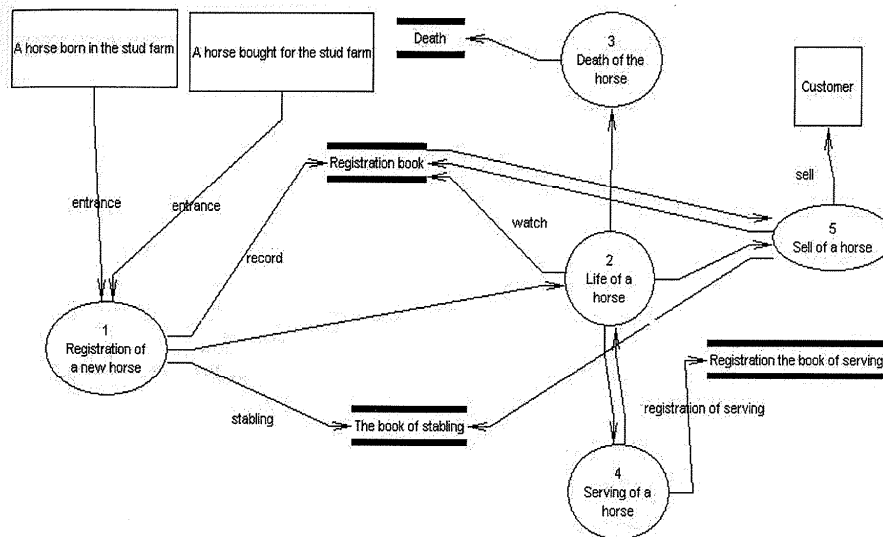
Current paper documentation is based on constant writing of particular data about a horse. Records made in this way were and still are made in different books.

3.3.1 Current documentation

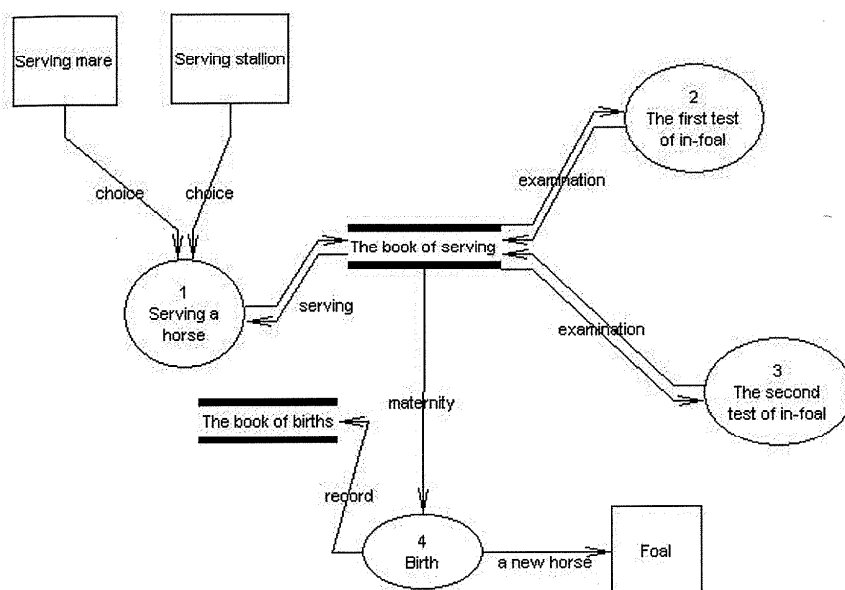
Current records are made in different evidentiary books, which keep different records about manipulation with a horse. The data are:

- Records of a new horse;
- stabling of a horse;
- serving a mare;
- examination of a horse;
- vaccination;
- sale;
- death.

3.3.2 DFD (Data Flow Diagram) diagrams of current systém of records



pic. 3.1 Process of recording of a new horse and its life in the stud farm.



pic. 3.2 Process of serving.

3.4 Determination of the ways of data gaining

The first step of the process of our paper analysis is determination of the ways of data gaining. In our case it contains everything from information cards and hand-written pre-printed forms to very complicated extensive hand-written records.

I started to examine all records on papers in this step. From each whole record there was created one sample. This was placed into the folder for later use at detail conversations with assigner. I also made copies of all forms, which have been used in the stud farm so far. For usage of table program Excel there was always made one screen up to present records.

	A	B	C	D	E	F	G	H
	Jméno klisny	Věk	Po	Z	Břeží po	Má se hřebit	Majitel	Adresa
1	SHADOW RIVER	24.5.83	Sharpen Up	Happy River	JALOVA		Alexandra Fillová	Pánné 106
2	LUCRETIA DE BORGIA	20.3.96	Woods of Windsor	Social Butterfly	BC	26.1.2006	Beck International s.ro.	Blatecká 3
3	NO FINESSE	12.3.88	Daring March	Hound Song	YO	8.6.2006	Borut Bernik Bogotaj	Kidričeva 1
4	RIWANA	3.5.00	Lavico	Regal White(USA)	????????	????????	Borut Bernik Bogotaj	Kidričeva 1
5	SARATOGA	13.5.89	Sectori	Some Diamond	HR	30.3.2006	Borut Bernik Bogotaj	Kidričeva 1
6	NON PAREILLE	17.2.95	Law Society(USA)	Noor Jamalik(FR)	LH	24.2.2006	Herbert Irsigler	Quadenstr
7	NATIONAL GALLERY	18.5.99	Lomitas(GB)	Noor Jamalik(FR)	LH	6.4.2006	Herbert Irsigler	Quadenstr
8	ARABESKA	25.2.97	Woodman	Aragon	SR	6.6.2006	Herbert Irsigler	Quadenstr
9	BRIGHT ANGEL	27.4.87	Antuco	Babsy	SR	3.2.2006	Hřebčín Napajedla, a.s.	Zámecká 5
10	COSIMA	22.3.98	Arcane(USA)	Comitia	SR	22.3.2006	Hřebčín Napajedla, a.s.	Zámecká 5
11	VAROSIA	27.3.99	Arcane(USA)	Vačka	LH	Potrat	Hřebčín Napajedla, a.s.	Zámecká 5
12	SHERCANA	25.4.97	Arcane(USA)	Shehrama(RE)	SR	18.4.2006	Hřebčín Napajedla, a.s.	Zámecká 5
13	SHERPA	2.5.98	Arcane(USA)	Shehrama(RE)	LH	31.3.2006	Hřebčín Napajedla, a.s.	Zámecká 5
14	LAKONIE	9.1.84	Arcaro	Latina	MP	8.1.2006	Hřebčín Napajedla, a.s.	Zámecká 5
15	FLAME BLOOD	24.3.02	Beccari(USA)	Flame Leaf	SR	24.3.2006	Hřebčín Napajedla, a.s.	Zámecká 5
16	LA MAGNIFICA	1.4.00	Beccari(USA)	Lady Sarah	LH	13.1.2006	Hřebčín Napajedla, a.s.	Zámecká 5
17	VÁČKA	29.1.80	Behietoun	Valentina	JALOVA		Hřebčín Napajedla, a.s.	Zámecká 5
18	QUEEN OF SHUFFLE	27.2.99	Big Shuffle	Quiteria	LH	30.1.2006	Hřebčín Napajedla, a.s.	Zámecká 5
19	LAURENA	18.1.90	Brustolon	Come Un Garcon	MP	29.4.2006	Hřebčín Napajedla, a.s.	Zámecká 5
20	MARKOTA	24.2.92	Carmelite House	Mullaghroe	BC	25.3.2006	Hřebčín Napajedla, a.s.	Zámecká 5
21	OLUJA	26.1.89	Celestial Storm	Sharp Run	BC	17.3.2006	Hřebčín Napajedla, a.s.	Zámecká 5
22	AMADARA	28.3.98	Dara Monarch	Amanda Praliné	BC	6.2.2006	Hřebčín Napajedla, a.s.	Zámecká 5

pic. 3.3 Sample of table records in the stud farm.

As you can see in the picture of current table records there are several data, which are important for mare survey:

- name of mare
- date of birth
- parents
- with foal after
- when it is supposed to foal (plan of birth)
- owner

All documents, copied forms, and given screens get in the stud farm were saved in to one file for later detailed analysis, and mostly for creation of questions for later conversation with the assigner.

3.5 Conversations

After detailed examination of all samples conversations with the assigner began. Our mutual conversation was based on giving open and also closed questions. The open

question gave me possibility to gain the most transparent possible information about the stud farm. The closed questions were given to gain different necessary details.

When giving the questions with open ending I tried to identify topics for next discussion from the answer. On the basis of particular techniques for database creation it was important to look up nouns in the answers. These nouns can substitute the entity list. In my case I found in the answers entities like: Horse, Owner, Stable, Box, State, and Manipulation as well. Manipulation entity is supposed to describe movement of a horse in the stud farm, and also to follow the length of stay for summation of feeding days. Total feeding days of particular horses are then given to financial department.

When my stage with entity identification finished, immediately I started the second stage – identification of characteristics of particular entities. This identification brought me the most detailed information about every found entity. I noted down all these characteristics on the special paper. Both these papers, with entities, and with characteristics, became the basis for creation of table structure.

3.6 Creation of table structure

My process started with definition of list of table database, which explores preliminary list of arrays. This process allowed me to identify entities in the arrays of the list. While exploring the list of arrays another entities may appear, which are not included in the list created in advance. In my case another identities, like type of manipulation and type of horse, were discovered. Later, both of the lists were joined together into one complex list of entities.

Next step in defining tables was the necessity to provide control of every item in entity list. In the case that there would exist duplicate items, it would be necessary to remove them and to replace them by other ones. In the case of concord of the entity names, however the necessity to have both entities, it is desirable to rename one of the. In my design of database there wasn't any such data.

3.7 Names of tables

Matter of naming the tables was quite difficult as far as I was concerned because this project is my first fruit in the world of databases. It was quite complicated to find proper names. I tried to create the names according to some specific rules.

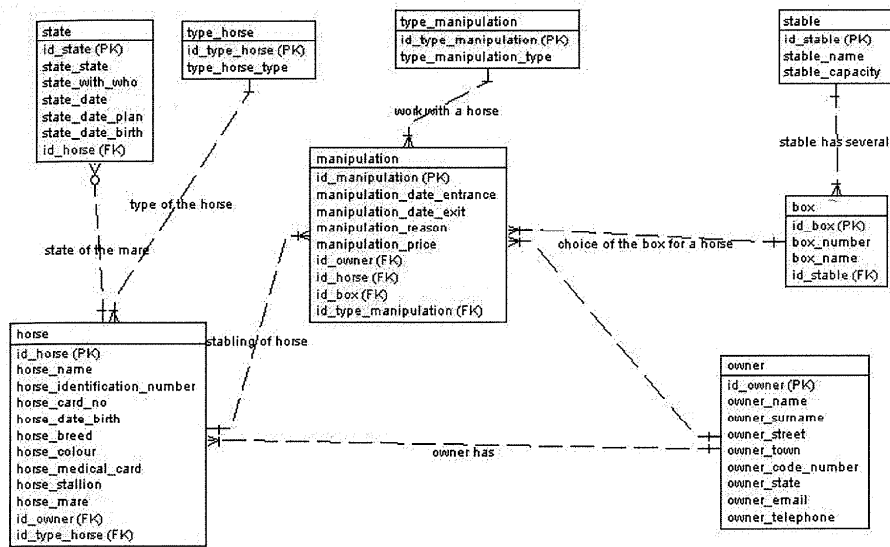
Rules for naming the tables:

- To create unique descriptive names clear to everyone in the organisation;
- to create names, which accurately, clearly, and unambiguously identify the entities represented in the table;
- for conveying of what kind of entity the table represents to use the smallest number of words possible;
- not to use words that express physical characteristics;
- not to use acronyms and abbreviations;
- not to use proper names and words, which limit data that can be put into the table;
- not to use names which implicitly or explicitly identify more than one entity. [2]

When satisfying all the rules I succeeded in matching names to all my tables. Names of tables were written in small letters, as well as items in other arrays. Items in particular entities were chosen for transparency in following PHP scripts so that they would transparently give meaning, what table I am just working with.

3.8 Designed database

I created the graphic design of the database (see pic. 3.5) from previous rules and instructions for creation of proper database. I paid attention to keeping all previous rules so that the given database would be designed properly. Badly designed database is definitely the sticking point of the whole design, and in later stage it could prove not to be sufficient function system of the whole database. With this fails the whole project. Attributes accomplish the first two normal forms and designs of primary keys accomplish the third normal form (see chapter 1).



pic. 3.5 Designed database in CASE studio program.

For my own creation of database I used the trial version of Case Studio 2 program, in which I designed all tables and their mutual relations, with caption of particular relations.

3.9 Designed names of entities

From the created model of database I can develop particular entity types into the table of attributes. I am trying to introduce the names, under which will be represented the data types in my design, key and NULL argument. Every table has its so-called primary key. Value of primary key must be unique within the table. Primary key must always be stated. With the assistance of database model and the table of attributes it is possible to create the appropriate database.

Table: horse

horse
id_horse (PK)
horse_name
horse_identification_number
horse_card_no
horse_date_birth
horse_breed
horse_colour
horse_medical_card
horse_stallion
horse_mare
id_owner (FK)
id_type_horse (FK)

pic. 3.6 Sample of horse entity table imported from CASE studio.

Table's task is to register all horses in the stud farm. It considers horses born in the stud farm, and also the ones bought for the stud farm, or stabled for temporary period of time by external owner. Primary key here is id_horses attribute, which positively identifies each horse and is connected with the State and Manipulation table. Furthermore, in the table there are attributes of array positively describing the horse, such as: horse_name, horse_identification_number, horse_card_no., horse_date_of_birth, horse_breed, horse_colour, horse_medical_card, horse_stallion (father of horse), horse_mare (mother of horse). Unfamiliar keys here are id_owner providing connection with owner table, id_type_horse providing connection with type_horse table.

Table: state

state
id_state (PK)
state_state
state_with_who
state_date
state_date_plan
state_date_birth
id_horse (FK)

pic. 3.7 Sample of state intity imported from CASE studio.

I am trying to describe the monitoring of mare gravidity in state table. Primary key here is id_state for positive identification of given couple. Unfamiliar key id_horse provides connection with horse table and following possibility to choose mares stabled in

the stud farm, and state_with_who attribute provides possible choice of stallion for mating. Other attributes like state_date, state_1_scan, state_1_result, state_2_scan, state_2_result, state_date_plan, stae_date_birth, and state_date are and will be important information for assigner.

Table: type_horse

type_horse
id_type_horse (PK)
type_horse_type

pic. 3.8 Sample of type_horse entity table imported from CASE studio.

In the type_horse table the user fills particular types of horses, which may appear in the stud farm. It considers particular types of horses, such as: Foal, Gruntling, Weaned, Stallion, Mare, and Others. Table is connected with horse table through primary key id_state.

Table: type_manipulation

type_manipulation
id_type_manipulation (PK)
type_manipulation_type

pic. 3.9 Sample of type_manipulation entity table imported from CASE studio.

Table type_manipulation gives the user possibility to add through type_manipulation_type attribute different types of manipulations, which can happen with the horse in the stud farm. Primary key id_type_manipulation is connected with manipulation table where will be the type_manipulation attribute monitored.

Table: owner

owner
id_owner (PK)
owner_name
owner_surname
owner_street
owner_town
owner_code_number
owner_state
owner_email
owner_telephone

pic. 3.10 Sample of owner entity table imported from CASE studio.

Owner table saves all necessary records about owners, where primary key id_owner is connected to horse and manipulation tables.

Table: stable

stable
id_stable (PK)
stable_name
stable_capacity

pic. 3.11 Sample of stable entity table imported from CASE studio.

Stable table serves as register of all stables in the stud farm. Through stable_name attribute name of stable will be saved, and through stable_capacity attribute the user will fill the total possibility of number of horses to be stabled in the given stable. Primary key id_stable is connected with box table.

Table: box

box
id_box (PK)
box_number
box_name
id_stable (FK)

pic. 3.12 Sample of box entity table imported from CASE studio.

Box table serves to register all boxes in the given stable, where there is recorded ordinal number of the box through box_number attribute, and name of box through box_name attribute. Primary key id_box is connected with manipulation table. Unfamiliar key id_stable provides connection with stable table.

Table: manipulation

manipulation
id_manipulation (PK)
manipulation_date_entrance
manipulation_date_exit
manipulation_reason
manipulation_price
id_owner (FK)
id_horse (FK)
id_box (FK)
id_type_manipulation (FK)

pic. 3.13 Sample of manipulation entity table imported from CASE studio.

Manipulation table is the most important of all tables. Here are monitored complete records about motion of horse in the stud farm through primary key id_manipulation. Attributes kept as manipulation_date_entrance and manipulation_date_exit monitor the length of stay of horse in the stud farm. Unfamiliar keys are here id_owner providing connection with owner table, id_horses providing connection with horse table, id_box providing connection with box table, id_type_manipulation providing connection with type_manipulation table.

With database designed in this way I can slowly start to work. By this I mean creation file with sql ending in CASE studio and by the following import I will convert it into console phpmyadmin. However, I can't claim that database designed this way is the exact copy of customer's demands. While programming there will appear particular problems, which will have to be solved by adding other identity to our design of database, or possibly by adding array to entity.

3.10 Sample of table in MySQL creation

Smaller sample of CREATE TABLE command for creation of horse table, in which horses will be registered.

```

CREATE TABLE horse (
  id_horse INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  id_type_horse INTEGER UNSIGNED NOT NULL,
  id_owner INTEGER UNSIGNED NOT NULL,
  horse_name VARCHAR(40) NULL,
  horse_identification_number VARCHAR(20) NOT NULL,
  horse_card_no. VARCHAR(30) NULL,
  horse_data_of_birth DATE NULL,
  horse_breed VARCHAR(10) NULL,
  horse_colour TEXT NULL,
  horse_medical_card TEXT NULL,
  horse_stallion VARCHAR(45) NULL,
  horse_mare VARCHAR(45) NULL,
  PRIMARY KEY(id_horse),
  INDEX horse_FKIndex1(id_owner),
  INDEX horse_FKIndex2(id_type_horse)
)

```

pic. 3.14 Creation of table in MySQL (My Structured Query Language).

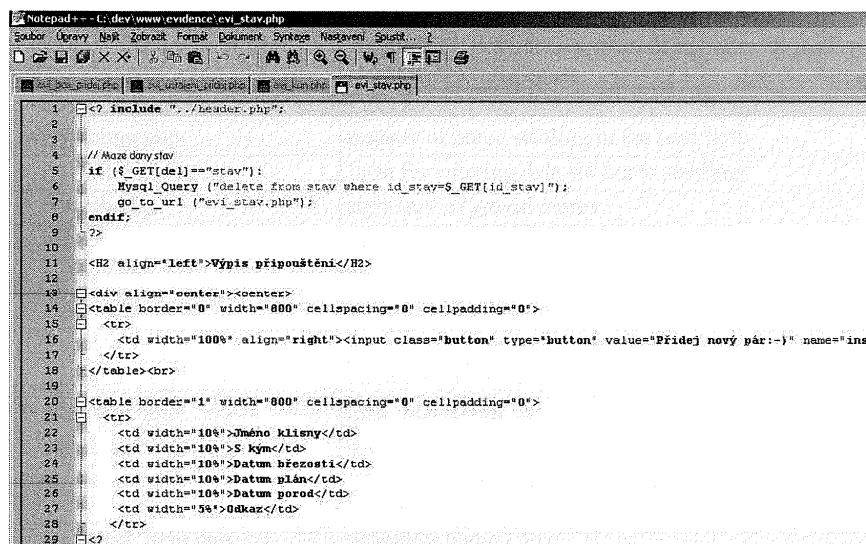
3.11 Prototyping

After table creating work on module was initiated. Main content was configuration and structure of the whole system. Every programmed part of module was presented to the user and gradually consulted. If the assigner was satisfied, programming of the rest of module still continued. [3]

4 PHP SCRIPTS + MySQL

After long thinking over created design of database I started the programming of the whole project itself. All register will be saved on free of charge server, which supports PHP as well as MySQL.

To operate on the interest as it should, I installed web server APACHE, server MySQL, programming language PHP, and phpMyAdmin on my local station, and every time I checked it all at my place in advance. Consequently, every script was copied to the server via FTP interface. Scripts were created with the assistance of freeware program Notepad++.



```
1 <? include "../header.php";
2
3
4 // Maza dany stav
5 if ($_GET[del]=="stav"):
6     Mysql_Query ("delete from stav where id_stav=$_GET[id_stav]");
7     go_to_url ("evi_stav.php");
8 endif;
9 ?>
10
11 <H2 align="left">Výpis připouštění</H2>
12
13 <div align="center"><center>
14 <table border="0" width="800" cellspacing="0" cellpadding="0">
15 <tr>
16 <td width="100%" align="right"><input class="button" type="button" value="Přidej nový pár:-)" name="ins
17 </tr>
18 </table><br>
19
20 <table border="1" width="800" cellspacing="0" cellpadding="0">
21 <tr>
22 <td width="10%">Jméno klienty</td>
23 <td width="10%">$ kým</td>
24 <td width="10%">Datum březosti</td>
25 <td width="10%">Datum plán</td>
26 <td width="10%">Datum porod</td>
27 <td width="5%">Odkaz</td>
28 </tr>
29 <?>
```

pic. 4.1 Sample of Notepad++ program.

4.1 Summary of all scripts

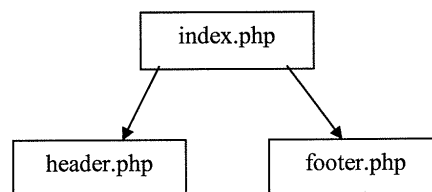
Table 4.1 Summary of all used scripts.

Name	Show srcipt
index.php	starting introductory site
header.php	reference menu

footer.php	header who created the program
calendar.php	calendar for transparent entering of dates
evi_horse.php	register of horse and summary of registered horses
evi_horse_add.php	form for entering data about horse
evi_owner.php	register of owner and summary of already registered owners
evi_owner_add.php	form for entering data about new owner
evi_stable.php	register of stables and summary of registered stables
evi_stable_add.php	form for entering data about stable
evi_box.php	register of boxes and summary of registered boxes
evi_box_add.php	form for entering data about box
evi_state.php	register of gravid state of mare and summary of already served mares
evi_state_add.php	form for entering data about serving
evi_type_horse.php	register of types of horses and summary of already registered types of horses
evi_type_horse_add.php	form for entering data about type of horse
evi_type_manipulation.php	register of types of manipulation and summary of already registered types
evi_type_manipulation_add.php	form for entering data about type of manipulation
evi_stabling.php	register of horse stabling in the stud farm
evi_stabling_add.php	form for entering data for horse stabling
p_maresb.php	summary of gravid mares
p_horses_all.php	summary of all stabled horses
p_owners.php	summary of all owners
p_dividedinto.php	summary of all horses divided according to types
p_freeboxes	summary of free boxes for stabling of new horse
s_feedingdays.php	possibility to monitor feeding days at stabled horses
h_gravidity.php	historical summary of all mares
h_stabling.php	historical summary of all stabled horses
data_decode.php	function for converting date into more user-transparent format
form_select_item.php	generates combo box
go_to_url.php	function for passing to other www site
feeding_days.php	function for counting feeding days
option_list.php	conversion of array into html choices
serving.php	function for number of days for planned foaling
fn.php	includes all functions

4.2 index.php

I included two other scripts to introductory script and simultaneously the introductory site according to quickening loading of sites. Script includes only single code of function for calling menu elaborated in the script header.php, and label, who made the program in the script footer.php. Both scripts are also included in rest of scripts because if they were written in every script separately, the script would become confused and very demanding utility. At every change in menu header each script would need to be explored and made change everywhere. Loading of sites would get very slow.



pic. 4.2 Sample of content of index.php script.

4.3 evi_*.php

Almost all scripts serving to register either horse itself or owner or stable have the same body of program, they differ only in small deflections.

```
<?
$result= Mysql_Query ("select * from manipulation where manipulation_date_exit='0000-00-00'");
while ($rec = mysql_fetch_array($result)):
$result1 = mysql_query("SELECT * FROM horse WHERE id_kone=$rec[id_horse]");
$rec_kun = mysql_fetch_array($result1);
$result2 = mysql_query("SELECT * FROM type_manipulation WHERE id_typ_manipulace=$rec[id_type_manipulation]");
$rec_typ_manipulace=mysql_fetch_array($result2);
$rec[manipulation_date_exit]=date_decode($rec[manipulation_date_exit],"sql", "normal");
?>
```

pic. 4.3 Sample of body of script with standard answering.

In the body of script you can notice standard answering on manipulation table, which is also treated by the condition for displaying statement of information on user's screen. It is clause WHERE. I pass through arrays with the help of condition WHILE and passing from line to line mysql_fetch_array function. I will provide passing through the whole table with it, and I will fill necessary information.[4]

4.4 date_decode.php

You have surely noticed in the previous sample of one script calling date_decode function. This function provides the change of standard format of date SQL. It examines if the entry format type 'YYYY-MM-DD' and if yes, it will convert them into user-friendly format, being 'DD-MM-YYYY'.

4.5 pripousteni.php

Another interesting script, or better said function is serving.php. This function provides counting of days from the serving of mare to the first test of gravidity of mare, the second, and next date of birth. From serving of mare due to the 16th day the gravidity test is made. Next testing comes on the 42nd day. Plan of foaling is arranged according to standard length of mare gravidity, i. e. 336 days.

```
$stav_datum_time=mktime
(0,0,0,substr($rec[$stav_datum],5,2),substr($rec[$stav_datum],8,2),substr($rec[$stav_datum],
0,4));

    $scan1                = 16 * 86400;
    $stav_1_scan           = date ('Y-m-d',($stav_datum_time+$scan1));
    //echo $stav_1_scan."<br>";
    $scan2                = 42 * 86400;
    $stav_2_scan           = date ('Y-m-d',($stav_datum_time+$scan2));
    $scan3                = 336 * 86400;
    $stav_datum_plan       = date ('Y-m-d',($stav_datum_time+$scan3));
    $aktual_time           = mktime (0,0,0,date('m'),date('d'),date('Y'));
```

```
$stav_dny      = round (( $aktual_time - $stav_datum_time)/3600 ) /  
24);
```

pic. 4.4 Sample of body of serving.php function.

Function also monitors number of days from serving to planned foaling. It is information for monitoring the vaccination. During gravidity vaccine is given to mare in particular intervals.

4.6 fn.php

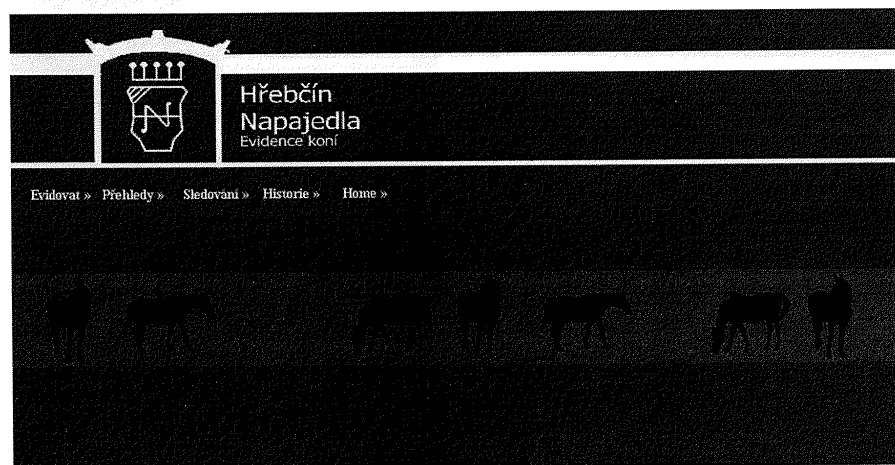
Very inconspicuous and at the same time useful script. I include completely all used functions to the only one, being fn.php, not to have to call the used functions separately. This function is repeatedly included in the script header.php. I have activation of all functions provided by this.

5 DESCRIPTION OF A MODULE OF HORSES' REGISTRATION DATABASE

An important aspect of the thesis was to fulfill requirements on the system given by an assiner. The most important requirement was monitoring of so called feeding days. Feeding days are those days which horses spend in a stable. From other requirements there was monitoring the mare gravidity from its primal serving until its delivery and also a possibility of selecting a mare to a stallion.

5.1 Introductory page

At an introductory page a very simple graphic interface is introduced to the users which consists of well arranged CSS (Cascading Style Sheets) of an unrolled menu and a simple universal graphics (picture 6.1). The user did not have any demanding requirements on graphics but the most of requirements were assigned to a functionality of the whole system.



pic. 5.1 Sample of the introductory page of horses' registration.

Evidovat = Registration
Přehledy = Surveys
Sledování = Monitoring

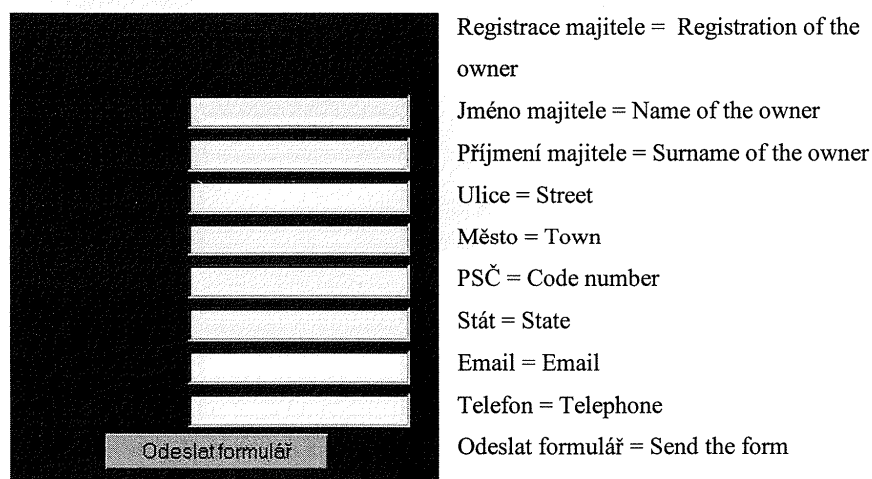
Evidence koní = Record of horses
Historie = History

5.2 Registration

In the menu item Register we find all concrete items which are needed to be recorded when a horse enters a stud farm, it stables and it serves. In all recorded items it is possible to record and delete the item.

Registration of the owner

In the first phase the owner is registered if he has not been registered before. After clicking the button add a new owner a well arranged form is shown. Into this well arranged form the data about the owner are adopted and they are sent after clicking the button send the form. Immediately a page of owners' survey is switched over where we make sure that the owner was added in registration.



Registrace majitele = Registration of the owner

Jméno majitele = Name of the owner

Příjmení majitele = Surname of the owner

Ulice = Street

Město = Town

PSČ = Code number

Stát = State

Email = Email

Telefon = Telephone

Odeslat formulář = Send the form

pic. 5.2 Sample of the form for registration of owners.

Registration of horses

Here a horse is registered. The principle of registration of horses is the same as registration of owners. Only some things in the form are kept in a different way. At the form of registration of horses (pic. 5.3) we can see various items which are used as a

source of information. In the item Owner we look for all registered owners in the stud farm.

Registrace koně = Registration of horses

Kategorie = Category

Jméno majitele = Name of the owner

Jméno koně = Name of the horse

Identifikační číslo = Identification number

Číslo půkazu = Card No.

Datum narození = Date of birth

Plemeno = Breed

Barva = Colour

Zdravotní karta = Medical card

Otec = Father

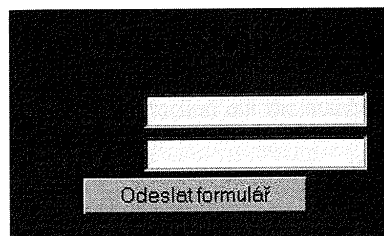
Matka = Mother

Odeslat formulář = Send the form

pic. 5.3 Sample of the form for registration of horses.

Registration of a stable

At filling the stables in the stud farm also a relevant stable with capacity is recorded first. Each stable has a given number of boxes which are recorded in another item.



Registrace stáje = Registration of a stable

Název stáje = Name of the stable

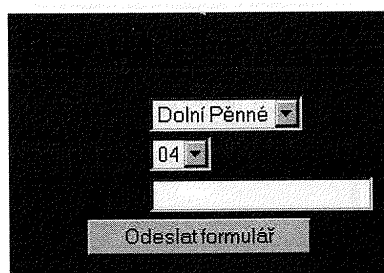
Kapacita = Capacity

Odeslat formulář = Send the form

pic. 5.4 Sample of the form for registration of the stable.

Registration of boxes

Here particular boxes in stables are recorded. After clicking the button add a new box, in the next form a particular stable is chosen. After choosing the stable, a number of the box menu is shown still in an unrecorded stable. By the button send the form we check if registration of the box is well recorded.



Registrace boxu = registration of boxes

Stáj název = Name of the stable

Číslo boxu = Number of the box

Název boxu = Name of the box

Odeslat formulář = Send the form

pic. 5.5 Sample of the form for adding the box into a stable.

Registration of type_horse

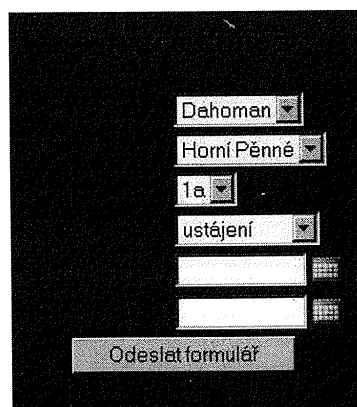
Record a type of horse menu which is in the stud farm.

Registration of type_manipulation

A simple way of recording the type of manipulation. The owner of the stud farm wanted to have a survey about a reason of putting the horse into a stud farm. Since it was not possible to find out the number of reasons, now any reason can be recorded in any time.

Registration of stabling

Now the horse is stabled. After clicking the button stabling we get into a well arranged form which serves for stabling the horse. There are various columns which are needed to be filled out at stabling. During choosing a stable, select a box in the stable menu will be shown one line below. But the boxes can be chosen just from those not occupied in the stable. It is done so to prevent from a problem of having two different horses in one box and in one stable. Another important thing is a date of entering. After its assignment and sending the form, horse's feeding days spent in the stable are counted until a date of leaving the stud farm which is written by edit menu.



Registrace koně = Registration of the horse

Jméno koně = Name of the horse

Stáj = Stable

Box = Box

Důvod vstupu = Reason of the entrance

Datum vstupu = Date of the entrance

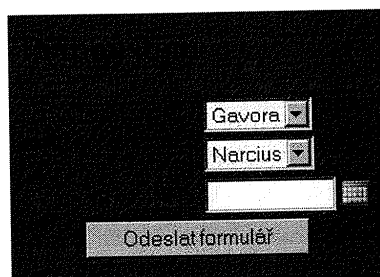
Datum výstupu = Date of the exit

Odeslat formulář = Send the form

pic. 5.5 Sample for stabling the horse.

Registration of serving

Registration of serving serves for a well arranged registration of mares used for serving. After clicking the button add a new pair a well arranged form is shown. Here is choose a mare menu and also assign a stallion menu. The date of a potential serving is determined and the form is sent. Other items are recorded. The user inserts the data into them later.



Připouštění = Serving

Klisna = Mare

Přidělen = Assigned

Datum připouštění = Date of the serving

Odeslat formulář = Send the form

pic. 5.6 Illustration of the form of serving.

5.3 Surveys

Here the user can look over all information about the stud farm and all items which are recorded. All possibilities of survey were consulted with the assigner since he knew which information he needed to be kept in particular links.

Surveys horses all

Sum shows all registered and just stabling horses. In clerkish tables too we can watch sum e.g. health card horses. After clicking on health cards display digestedly in new windows. Health card is editable for pertinent completion health information.

Výpis koní = Surveys horses all

Majitel = Owner

Jméno koně = Name of the horse

Kategorie = Category

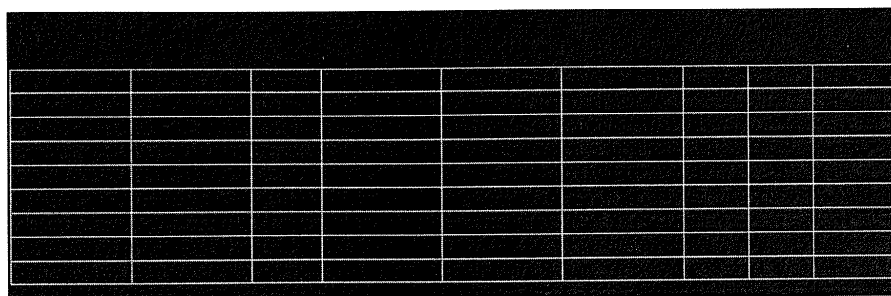
Číslo koně = Numer of the horse

Číslo průkazu = Card No.

Datum narození = Date of birth

Plemeno = Breed

Barva = Colour



pic. 5.7 Sample of a list of all stabled horses.

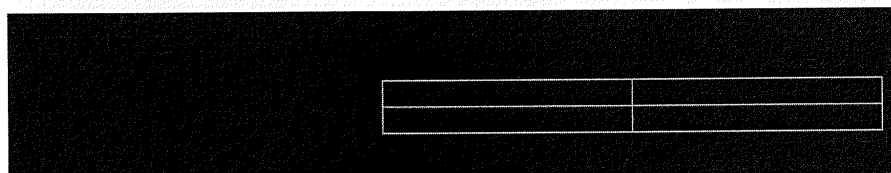
Survey of a mare gravidity

Mare gravidity item shows all mares which are before delivery and their first planned medical examination confirmed gravidity. In the table there is also a planned date of a presumable delivery which is calculated from a total duration of mare gravidity.

Klisny březí = Mares in-foal

Klisna = Mare

Datum plán porodu = Date of the plan of birth



pic. 5.8 Survey of mare gravidity.

Survey of owners

Then we can monitor all registered owners with their contact and e-mail address. There is the owners' address and the state.

Survey of free boxes

Free stalls provide the user with information about every box in a chosen stable. After selecting particular box the table with a number, a name of the stable and its occupation is shown. If a horse is stabled in the box, it is shown even with the date of stabling.

Výpis boxů = List of boxes

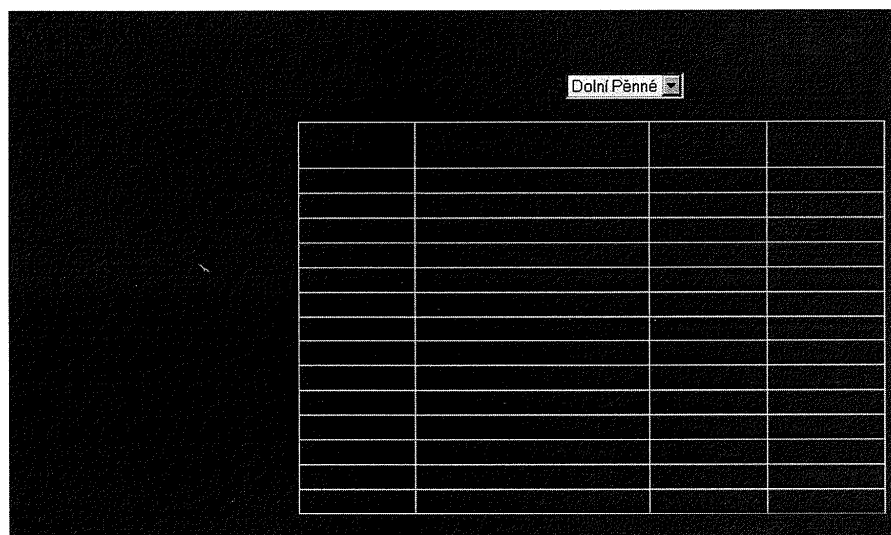
Obsazenost boxu = Occupation of the box

Zvol stáj = Choose stable

Datum vstupu = Date of the entrance

Box číslo = Numer of the box

Box název = Name of the box



pic. 5.9 Survey of free boxing in stable.

Survey division to

The last item in the menu survey is division to. For users is this item as important as the other ones. Here is the actual survey of types of horses which are actually stabled in the stud farm, arranged according to the type of a horse.

5.4 Monitoring

In this item the user monitors the feeding days of horses which are actually stabled in the stud farm.

Monitoring of feeding days

Monitoring of feeding days actually stabled in the stud farm is one of the main items which the user wanted to monitor.

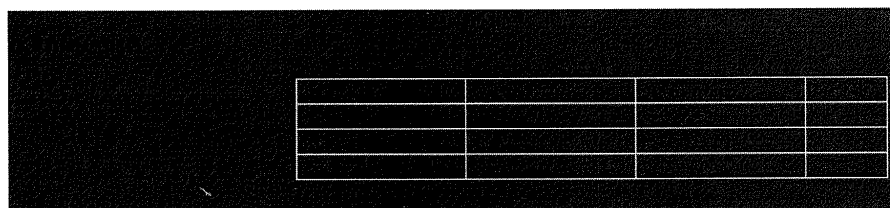
Krmné dny = Feeding days

Datum vstupu = Date of the entrance

Jméno koně = Name of the horse

Počet dnů = Number of days

Kategorie = Category



pic. 5.10 monitoring of feeding days of stabled horses.

5.5 History

The item history provides a historical survey of all stabled horses and all pregnant mares.

History of stabling

The item allows to monitor history of all stabled horses even in the past. In case the user does not need this entry in history, he can delete it.

Historie ustájení koně = History of stabling of the horse

Jméno koně = Name of the horse

Datum vstupu = Date of the entrance

Kategorie = Category

Datum výstupu = Date of the exit

Stáj = Stable

Počet dnů = Number of days

Důvod vstupu = Reason of the entrance

pic. 5.11 historical survey of all stabled horses in the past.

History of gravidity

Sum watch historical survey of all in former times brooked mare.

5.6 Testing

In terms of functioning, the module was treated and thereafter tested. The testing was provided at the assigner's. It was about different options in inserting proper data values.

A new horse accepted to the stud farm cannot get the same id_horse because the assignment of id_horse is protected by AUTO_INCREMENT function which automatically raises the value id_horse to upper line than a previous accepted horse.

Other testing course was a wrong entries insert into the form. Also these things were treated so the form will be filled out by correct data. For example, an owner's e-mail is not written in the table without @ sign.

In the final phase of the whole testing there was an exam of linking the primary and foreign keys connected between the tables. By assigning the fictional data, the inserted

data were shown in other connected tables too. After assigning the owners' data, his name was shown in select menu, in the form for adding a horse (pic. 5.3). Another test was to assign a date of horse's leaving the stud farm which was not shown in the survey of stabling all horses (pic. 5.7) any more and in serving menu its name was not shown, thereafter its stall was free. The horse was moved into a history of stabling horses (pic. 5.11). In the form of serving (pic. 5.6) for choosing a mare, there was not possible to see a stallion and vice versa.

Testing of the whole system showed us that the system is treated well and it is ready to be used.

5.7 Implementation of the project

For implementation of the whole project a suitable server will be selected to provide its services suitable for implementation of the database project. In server there must be a MySQL and PHP support with a possibility to link FTP by which the whole project will be recorded into the server. The database administration will be implemented by myAdmin with an assigned name and a password.

6 ECONOMIC EVALUATION OF THE THESIS

One of the last chapters I would dedicate to financial evaluation, actually to economical impacts associated with the implementation of this project in practice.

The highest investment for the stud farm Napajedl will be the database programme for registration of horses. Considering the hours and programming of the whole project I calculated the price of the whole programme to 20,000 Kč.

Considering a possibility of implementation of the project even in other stud farms I calculated the price of the whole project for the stud farm to 5,000 Kč. I will try to speak to other stud farms in surroundings and I will offer them an easy electronical registration of horses in their stud farm.

CONCLUSION

As everything has its beginning and its end, also my thesis had its beginning and now after several months of effort it draws to an end. In the introductory thematic chapters I described the brief characteristics of my aims and the use of freely available tools as PHP, MySQL, APACHE, the description of theoretic themes from a formation of the database to the UML language.

In the next parts of the thesis I focused on the formation and architecture of a database module in order to go together with the requirements of the end users. On the basis of their valuable knowledge I designed and created a complete database module. Then I tested it with the repeated assistance of the end users. After a successful general examination of the whole system connected with a detailed modification I sold it to the stud farm.

It is too soon to predict the accomplishment or failure of the whole project. This must be evaluated by the end user. Anyway I dare to claim that this thesis has accomplished its basic requirements demanded by the stud farm, namely its functional and graphic side.

An important part of it was the successful implementation of programming the PHP language and the database system MySQL which seem to be appropriate in realization of this or other similar projects.

Another essential step was the use of analysis by the UML tools which allowed faster and smooth transition from a diagram of the use to a construction of prototypes by the PHP scripts.

My work has not finished by the successful designing, formation and implementation of the database module in conditions of stud farm in Napajedla. The reverse is true. For me it is a commitment for the future. Even now I am preparing a training course for the end users, an extension of the existing database and we take account of installing on-line cameras into each stud. Inseparable part of the whole project will be a priority service and a data backup.

SUMMARY

The purpose of my bachelor project was the substitution and innovation of a current paper card file by a modern database system applied by methodology of software engineering where I used some methods of the UML language. For the successful implementation of electronic record I decided to make use some of servers which provide services for free with support of the MySQL and the PHP.

Since I convinced myself several times that even the apparent details can appear as an unexpected hitch and almost nothing is so easy, as I had thought before, even through this difficulty I consider this work as successful. It is not only because of my final bachelor's thesis but also because I gained valuable experience and knowledge during creating the project (I speak about valuable practical knowledge regarding programming, architecture of the database, look into the PHP language, or even an actual look into a daily running of the stud farm.

Thanks to the latest knowledge I succeeded to design and implement the compact database structure which, comparing to current paper records, will allow the following advantages:

- Saving of time in insert, record, backup and evaluation of the data;
- access to needed information via internet;
- modern graphic data processing.

BIBLIOGRAPHY:

- [1]ARLOW, J., NEUSTTADT, I.. *UML a unifikovaný proces vývoje aplikací*. Přel. B. Kiszka. 1. vyd. Brno : Computer Press, 2003. 390 s. ISBN 80-7226-947-X.
- [2]HERNANDEZ, Michael J.. *Návrh databází*. Přel. J. Bouda. 1. vyd. Praha: Grada publishing, 2006. 408 s. ISBN 80-247-0900-7.
- [3]KADLEC, V. *Agilní programování*. 1. vyd. Brno: Computer Press, 2004. 280s. ISBN 80-251-0342-0.
- [4]WILLIAMS, Hugh E., LANE, D.. *Programujeme webové aplikace pomocí PHP a MySQL*. Přel. D. Krásenský. 1. vyd. Praha: Computer Press, 2002. 532 s. ISBN 80-7226-760-4.
- [5]FORTA, B.. *Sams Teach Yourself SQL in 10 Minutes*. Third Edition. Sams Publishing, 2004. 256 s. ISBN 0-672-32567-5.
- [6]AMBLER, Scott W.. *The Elements of UML(TM) Style*. Cambridge University Press, 2002. 160 s. ISBN 0-521-52547-0.

LIST OF USED ABBREVIATIONS:

UML	Unified Modeling Language
SQL	Structured Query Language
MySQL	My Structured Query Language
RDBMS	Relational Database Management System
FTP	File Transport Protocol
NF	Normal Form
OCL	Object Constraint Language
PHP	Hypertext preprocesor
OOP	Object Oriented Programing
DFD	Data Flow Diagram
CSS	Cascading Style Sheets