HEARTFAID

D21 – Functional specifications of data warehouse implementation and data preparation

Submission date: 14/09/2007
Due date of document: 01/08/2007

HEARTFAID

A KNOWLEDGE BASED PLATFORM OF SERVICES FOR SUPPORTING MEDICAL-CLINICAL MANAGEMENT OF THE HEART FAILURE WITHIN THE ELDERLY POPULATION

Project summary	
Project acronym:	HEARTFAID
Project identifier:	IST – 2005 – 027107
Duration of the Project:	01/02/2006 – 31/01/2009
Project Co-ordinator:	UNICAL University of Calabria (Italy)
Thematic Priority:	Information Society Technology
Instrument:	Specific Targeted Research or Innovation Project

Consortium

- UNICAL - Università della Calabria (Italy)
- UNICZ - Università degli studi Magna Graecia di Catanzaro (Italy)
- UNIMIB - Università degli studi di Milano Bicocca (Italy)
- JUMC - Jagiellonian University Medical College (Poland)
- VMWS - Virtual Medical World Solutions Ltd (United Kingdom)
- FORTHNET S. A.- Hellenic Telecommunications and Telematic Applications Company S. A. (Greece)
- SYNAP - Synapsis s.r.l. (Italy)
- CNR - Consiglio Nazionale delle Ricerche (Italy)
- FORTH - Foundation for Research and Technology Hellas (Greece)
- RBI - Rudjer Boskovic Institute (Croatia)
- AUXOL - Istituto Auxologico Italiano (Italy)

D21 – Functional specifications of data warehouse implementation and data preparation

Document summary	
Document title:	D21 – Functional specifications of data warehouse implementation and data preparation
Document classification:	Derivable D21
Dissemination level:	CO
Submission date:	14 September 2007
Due date:	01 August 2007
Authors:	Antonio Candelieri, Domenico Conforti, Rosita Guido, Vincenzo Lagani - UNICAL Dragan Gamberger, Tomislav Šmuc, Marin Prcela, Mislav Malenica, Rajko Horvat - RBI
Work package:	WP4 – Knowledge representation, discovery and management
Report version:	1.1

Short description
The deliverable presents details of the data warehousing implementation necessary for the data extraction from operational databases and data transformation into the form that can enter the knowledge discovery process.

Change record		
Version number	Changes	Release date
0.0	first draft of the table of contents	24/05/2007
0.1	first draft of the document	20/06/2007
0.2	second draft of the document	17/08/2007
1.0	First complete deliverable	31/08/2007
1.1	Final document version	10/09/2007

Table of contents

1. Executive Summary

Functional specifications for the data warehouse implementation and data preparation summarizes the work performed under first two tasks, T4.1 - Implementation of a suitable data warehouse for knowledge discovery and T4.2 - Data understanding and preparation, of the WP4 - Knowledge representation, discovery and management.

Data warehousing solution is relevant for integration between data and knowledge levels. Its main task is data preparation, storage and transformation in the form appropriate for the knowledge discovery process.

This deliverable reports the requirements put in front of the data warehousing solution, with respect to the main platform functionalities. From these requirements basic architecture of data warehouse components, storage, and component interactions are developed in the form of functional specifications.

Introduction provides the brief overview of the main purpose and functionalities of the data warehousing component as well as basic requirements related to final data preparation for knowledge discovery. It follows the detailed methodological presentation of the state-of-the-art in data warehousing and OLAP technology with special overview of current applications and research issues within the health care domain. The choice of the PENTAHO technological platform for the actual implementation of the data warehouse prototype is discussed at the end of this part.

After the presentation of all relevant technological issues, data warehouse system requirements are presented, starting from the general ones and proceeding to specific requirements and optional functionalities. Based on these requirements, the prototype component and the storage architectures of the data warehouse system are developed. Functionalities of the extraction transformation and loading (ETL) tools are given next. It follows the mapping of HF platform user types and data warehouse functionalities with basic technological functional specifications of the system.

Following sections are devoted to detailed presentation of the requirements for the process of final data preparation and transformation for knowledge discovery. Based on these requirements, functional specifications of the so called KD-ETL (knowledge discovery extraction, transformation and loading) component are exposed.

Special attention is devoted to the transformation of data collected as time sequences. Two templates for forming data tables for knowledge discovery tasks are specified and flattening processes for long and short sequences are defined. Finally, the Appendix includes the complete list of patient descriptors extracted from follow-up visits and used for the knowledge discovery.

2. List of abbreviations

TERM	DEFINITION
BI	Business Intelligence
BPV	Blood Pressure Variability
BRS	BaroReflex Sensitivity
CRF	Case Report Form
DB	Data Base
DBMS	Data Base Management System
DM	Data Mining
DSS	Decision Support System
DW	Data Warehouse
ETL	Extraction, Transformation, and Loading
GUI	Graphical User Interface
HCR	Health Care Record
HF	Heart Failure
HFP	HEARTFAID Platform
HRV	Heart Rate Variability
JDBC	Java Database Connectivity
KD	Knowledge Discovery
KDD	Knowledge Discovery in Databases
MDBMS	Multidimensional database management system
MOLAP	Multidimensional OLAP
NYHA	New York Heart Association
OLAP	On-Line Analytical Processing
OLTP	On-Line Transaction Processing

3. Introduction

The HEARTFAID platform objective is to aid healthcare stakeholders through a multi-level heterogeneous architecture, efficiently providing functionalities in handling data, information and knowledge, and supporting health care. Four general levels of HEARTFAID platform information system architecture are:

- Data level
- Middleware level
- Knowledge level
- Decision support level

HEARTFAID Data Warehouse solution is relevant for integration between data and knowledge levels. Its main task is data preparation and transformation in the form appropriate for the knowledge discovery process. Structuring data warehouse solution according to some general principles and HEARTFAID project specific requirements, as well as drafting main data preparation functionalities are the main tasks of this deliverable.

3.1. Data warehousing background

One of the most important assets of any organization today is its information. This information is almost always kept by an organization in two systems: operational storage systems of records and a data warehouse. Operational storage is where the data is recorded and used for on-line processing. Data warehouse (DW) is where the information based on collected and structured data is delivered to users for off-line data analysis tasks.

DW concept was developed as a result of a need for automated support in decision making, primarily in economy and finance branches. The main purpose was the support in business decision making, where the analysis of the large amounts of data is performed repeatedly and in a very specific manner, and where the timing for business decision is crucial. In that environment, DW has specific requirements:

- providing prompt access to high quality data and data understanding;
- using the data for extracting valuable information;
- flexible and efficient way of accessing the data for the analytics, managers and decision makers;
- providing the complete picture of the business activity;
- analysis and understanding of business fluctuations;
- tracing and predicting the market situation with prompt reaction to changes.

In medical practice, DW has yet to settle its role, since there are very few examples of practical implementations of DW in real medical environments [1]. Within the HEARTFAID platform, DW's primary task is data preparation for the KDD process.

3.2. Database and data warehouse

Within the informational system, the database takes the central position. The data in the database represents the current state of the system in a way that is appropriate for utilization.

The operational (transactional) database gathers data that is interconnected and stores it without unnecessary redundancy with the purpose of making it available in various appliances and in an optimal way. The data is stored in a way that is not influenced by the applications that are using it. The common ways of using, adding, changing, relocating and deleting data are established. The data in the database is structured following a data model, which is characterized by the following components:

- set of objects, that are basic elements of data model;
- set of operations, which can be executed upon the objects and are used to access, browse, and modify the data about the objects;
- set of integrity rules, which implicitly or explicitly define the set of consistent data states and/or state changes, and that are general in a sense that they are applicable to any database that uses the defined model.

Database management system (DBMS) is an application system that provides the basic functionalities of data models in a process of creation and usage of databases. It consists of integrated collection of application tools that provides:

- data description and manipulation;
- high level user interface to data, independent of data structure;
- efficient usage and understanding information stored in a database.

The transaction component of an information system is also often referenced as OLTP (On Line Transaction Processing) component. OLTP stands for a class of programs that facilitates and manages transaction-oriented applications. The main benefits of OLTP are simplicity and efficiency.

The differences in goals between a database and a data warehouse system directly affect the differences of their data structure, components, and processes. The users of an operational storage use mostly one record at a time, while users of a data warehouse use larger chunks of data to extract information and get answers to more general questions about processes in the organization. Extraction of summary information directly from the database (when possible) would require complex queries that are hard to comprehend and maintain. The operations that would occur in query executions would be complex with high execution cost. This problem is mainly the consequence of the time expenses for combining relational tables. Additionally, the frequency of such queries can be unacceptable for operational databases.

The basic differences between database systems and data warehouse systems are summarized in Table 3-1.

Table 3-1. Database and a data warehouse relationship summary.

	OLTP	**DW**
Data content	Current values, detailed data	Historical data, detailed and summarized data
Data volatility	Data volatile	Data persistent
Purpose	Support for every-day operations	Reporting and data analysis
Processing item	Transaction	Query
Users	Personnel	Analytics and managers
Availability	Very important	Less important
Data updating	Frequent	No direct updates
Operations	Reading/Writing	Reading
User interaction	Foreseen	Ad-hoc
Data accessing count	Dozens	Millions
Focus	Storing data	Retrieving information

3.3. Data warehouse characteristics

The target features of the data warehouse in informational systems are:

- thematic orientation;
- integration and consistency;
- data valid for a specific moment in time or in time period;
- persistency.

The centre of a data warehouse model refers to the real world entities that are of concern in data analysis and decision support tasks. The thematic orientation is regularly diverse from the operational approach. In the medical environment, operational database is patient centric, and deals with entering and retrieving single patient data for the health care professional. Data warehouse is related to extraction of general relations (knowledge) concerning many patients (interesting subpopulations) regarding quality of care, understanding causes or consequences of relevant medical events and states, or financial (management) issues.

Patient databases are institution specific; it is rather hard to find two institutions that have the same database structure and profile. Moreover, it is possible that within one institution data is held in more than one database and these databases are not mutually compliant. The problems in database diversity may be summarized to the following:

- different platforms (various operating systems);
- different types of databases and files;
- different types of data formats;
- inconsistent database primary and foreign keys;
- various measure units, measure time intervals, ...

Even though the institutions perform rather similar medical procedures and treat the patients with the same problems in similar manner, the data that is monitored and stored does not appear in the databases in the same format. Nevertheless, every data item in every database might hold valuable information for disease analysis and for discovering potentially valuable medical knowledge. The most important characteristic of the DW is that it holds the data integrated from various institution-specific databases (Figure 3-1).

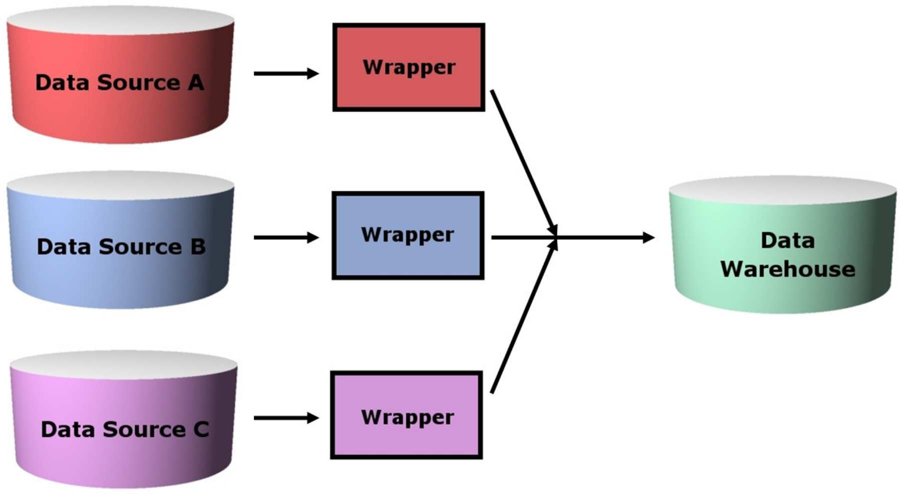


Figure 3-1. Data integration within the data warehouse.

The integration term also comprises the consistent names of data elements, measure units for data elements, coding structures, keys to data elements etc. The data should be stored in a unique and generally accepted manner.

The resulting DW holds comprehensive stream of captured data images represented by the sequence of data layers describing the state of the patients for a specific time intervals. Each layer represents image of data from a certain moment of time or from period of time. If the image was taken properly, there is no need for updating the data afterwards; corrections are made only in situations when oversights occur in data collecting process. It leads to the persistency of the DW data. Once the data is loaded into DW, it is not modified or deleted. Only new data is added in predefined time intervals.

3.4. Data preparation and transformation for knowledge discovery

Knowledge discovery has the purpose to detect potentially relevant relations among data. For that purpose, data must be transformed into the form of a table with *examples* described by their *attributes* (Figure 3-2). In medical applications rows of the table (examples) are usually patients while columns (attributes) are their properties consisting of signs, symptoms, findings, prescribed medication, and other information describing their current status and medical history.

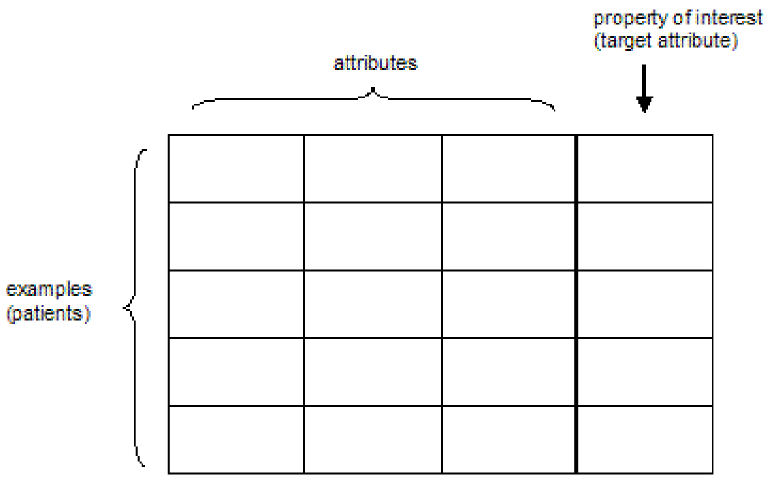


Figure 3-2. Knowledge discovery task prepared in the form of a table ready for the knowledge discovery process

In order to enable the transformation, the first step is data understanding with the intention to define goals of the knowledge discovery process. Each goal can be implemented by more than one concrete knowledge discovery task. For each task we have to define: patient subpopulation that will be used in the task, attributes (patient characteristics) that will be analysed, and the *property of interest* (i.e. worsening status, unexpected hospitalization, NYHA classification variation, decompensation). It means that for each task we need a separate data table.

Data understanding is strongly related to expert medical expectations from the data analysis process. Technical part of the work is to formulate tasks so that reasonable knowledge discovery can be performed. It must be ensured that:

a) there are enough useful attributes for the selected property of interest

b) there is enough examples of patients that have property of interest and enough control examples (patients that do not have the property of interest)

The data understanding process usually results in many different knowledge discovery tasks. Each of them, depending on the number of available examples, can be specified for various subpopulations (i.e. age group, NYHA class, sex, medication type). It is the task of the data warehousing to enable extraction of data in the form prepared for knowledge discovery for all defined tasks and various subpopulations. In an ideal situation, data warehousing can also help in the attribute selection. In this way, for the property of interest, for each task and subpopulation different attribute subsets can be specified with intention to enable detection of different relations.

If data cleansing is successfully done during data warehousing (as the part of the ETL process) then no special data exception handling is necessary as part of the data preparation step. In contrast to that, attribute selection, feature construction and selection, as well as noise detection, are always specific for the constructed knowledge discovery task.

Special attention in medical domains like HF must be devoted to the fact that most of the collected data is time related. At the first place these are continuously monitored data. Patient data recollected during multiple visits to specialized institutions (i.e. laboratory assessment, current therapy, changes in status) has also the meaning of time sequences. Knowledge discovery can not effectively use such time dependent information directly. It is an important task of the data preparation step to transform time sequences into useful attributes that can enter the standard knowledge discovery process [2].

3.5. Bibliography and references

[1] Pedersen, T.B., Jensen, C.S.(1998) *Research Issues in Clinical Data Warehousing*. Proc. Int. Conf. on Scientific and Statistical Database Management (SSDBM'98). IEEE Computer Society Press (2001) pp. 43-52.

[2] Augusto, J.C. (2005) *Temporal reasoning for decision support in medicine*. AI in Medicine 33(1):1-24.

4. State-of-the-art in data warehousing and OLAP applications

Data warehousing and OLAP analytic tools are the mainstream technology developed by most of the large IT companies. The reason for it lays in the fact that primary users of DW and OLAP technologies are business oriented companies. That is also the reason that most of the DW + OLAP platforms are in common IT language addressed as business intelligence (BI) software and applications. Business intelligence (BI) is a business management term which refers to applications and technologies used to gather, provide access to, and analyze data and information about business operations. In contemporary business intelligence systems data warehouse is the primary component.

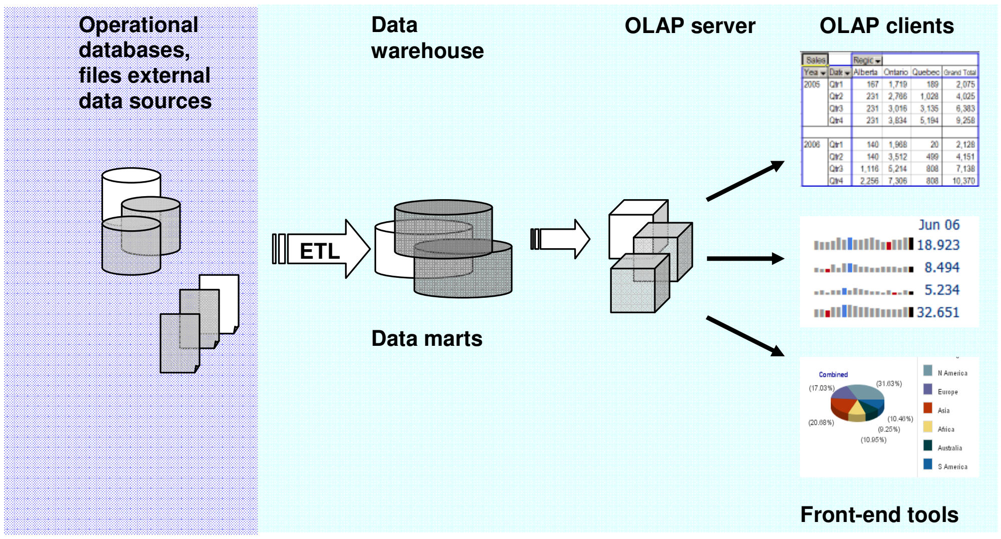
When a BI system is well-designed and properly integrated into organization's processes and decision-making process, it may be able to improve a company's performance. Having access to timely and accurate information is an important resource for organization, which can expedite decision-making and improve end user's experience. For the system to work effectively, organizations address the need to have a secure computer system which can specify different levels of user access to the data warehouse. As well, a system needs to have sufficient data capacity, a plan for how long the data will be stored (data retention). Analysts also need to set benchmark and performance targets for the system.

4.1. Data warehouse architecture

Data warehouses generally adopt a multi-tier architecture (Figure 4-1).

The first tier is a warehouse server, often implemented by relational DBMS (Data Base Management System). Data of interest must be extracted from operational legacy databases, and cleaned and transformed by ETL (Extraction, Transformation, Loading) tools before being loaded in the warehouse. This step aims to consolidate heterogeneous schema (structure heterogeneity, semantic heterogeneity) and to reduce data in order to make it conform to the data warehouse model (using different transformation, aggregation, and discretization functions). Data warehouse finally contains high quality, historical, and homogeneous data.

The second tier is a data mart. The data mart handles data sourced from the data warehouse, reduced for a selected subject. The main advantage of data marts is isolating data of interest for a smaller scope, thus permitting the focusing on optimization needs for this data and increase security control. However, this intermediate data mart is optional.



Sales		Regic			
Yea	Date	Alberta	Ontario	Quebec	Grand Total
2005	Qtr1	167	1,719	189	2,075
	Qtr2	231	2,766	1,028	4,025
	Qtr3	231	3,016	3,135	6,383
	Qtr4	231	3,834	5,194	9,258
2006	Qtr1	140	1,968	20	2,128
	Qtr2	140	3,512	499	4,151
	Qtr3	1,116	5,214	808	7,138
	Qtr4	2,256	7,306	808	10,370

Figure 4-1. OLAP tier architecture

The OLAP server composes the 3rd level. It calculates and optimizes the hypercube, i.e. the set of fact values for all the tuples of instances of dimensions. In order to optimize accesses against the data, query results are pre-calculated in the form of aggregates. OLAP operators permit materializing various views of the hypercube, allowing interactive queries and analysis of the data. Common OLAP operators include roll-up, drill-down, slice and dice, rotate.

The fourth level is an OLAP client, which provides a user interface with reporting tools, analysis tools and/or data mining tools. For a standard use, appropriate software solutions already exist. If the studied data and analysis processes are complex, then a specific interface must be designed.

4.2. Data warehouse methodological foundation

A data warehouse is commonly seen as a single site repository of information collected from multiple sources. The information in the data warehouse is organized around major subjects and is calculated in the form of aggregates. The data is modelled so as to allow pre-computation and/or fast access to various views of the data, providing interactive summarized data views.

4.2.1. Data models

According to Kimball [1], the data-modelling paradigm for a data warehouse must comply with requirements that are profoundly different from the data models in OLTP or operational environments. The data model of the data warehouse must be easy for the end-user to understand and write queries, and must maximize the efficiency of queries. Data warehouse models are called multidimensional models or hypercubes and have been formalized by several authors [1, 2]. They are designed to represent measurable facts or indicators and the various dimensions that characterize the facts.

In a data warehouse, data is adapted to efficiently suit the predefined data analysis tasks. The efficiency is manifested as a fluency and readiness of data that is processed and analyzed in multiple dimensions (time dimension, patient centric view, institution summaries etc.).

A conceptual model of a database is entirely independent of database management system (whether it is implemented as hierarchical model, network model, or relational model) and of operational system which endorses the database. Within a conceptual model, only model entities, attributes for those entities and relations among entities should be described; the technical details of real model implementation are irrelevant.

Database models

A classic example of conceptual model is ER model (entity-relationship model, Figure 4-2). The conceptual ER model is represented by ER diagram, for which a few different but very similar approaches exist. ER model represented using ER diagram has proved to be appropriate for conceptual database representation and is used very often in practice. The user friendly approach of the ER diagram depicts relations among the objects that are included in defining the database structure. The rules for transformation from ER model to any other standardized database model (hierarchical, network, or relational) are defined.

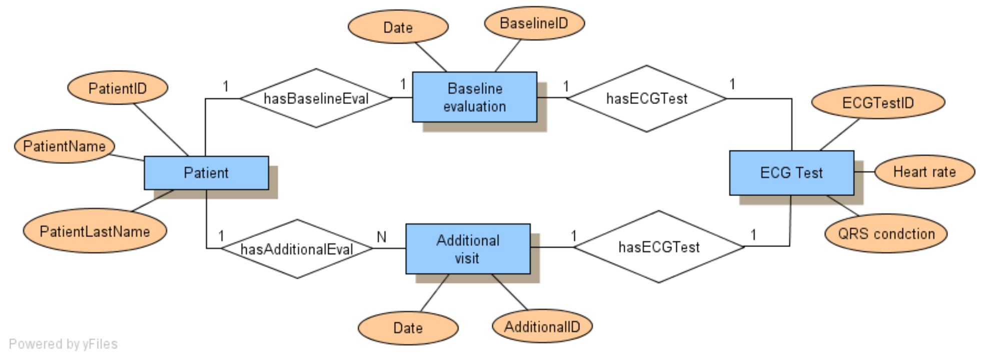


Figure 4-2. Entity-relationship data model.

The main advantage of ER model, along with its user friendliness, is that it is easily converted to the relational data model, which is commonly used as a model in great majority of DBMS and in practical implementation of databases.

In a relational data model (Figure 4-3) the basic structural element is *relation*. The relation is best presented in a form of a table where (caution – term “relation” is used both for a basic structural element, which is in this case a table, and for relations among entities):

- columns represent attributes;
- rows represent entities description;
- values in a column are defined by attribute domain;

- every table has a primary key – a set of attributes (one or more) that have unique value combination for each table row (entity), and can be used to identify every row.

The relational model has the same structure and syntax for representing both entities and relations among them. It has the following characteristics:

- simplicity;
- mathematical formalism that ensures uniqueness and clarity in defining data structures and operations on data;
- generality;
- independency of implementation details and browsing method.

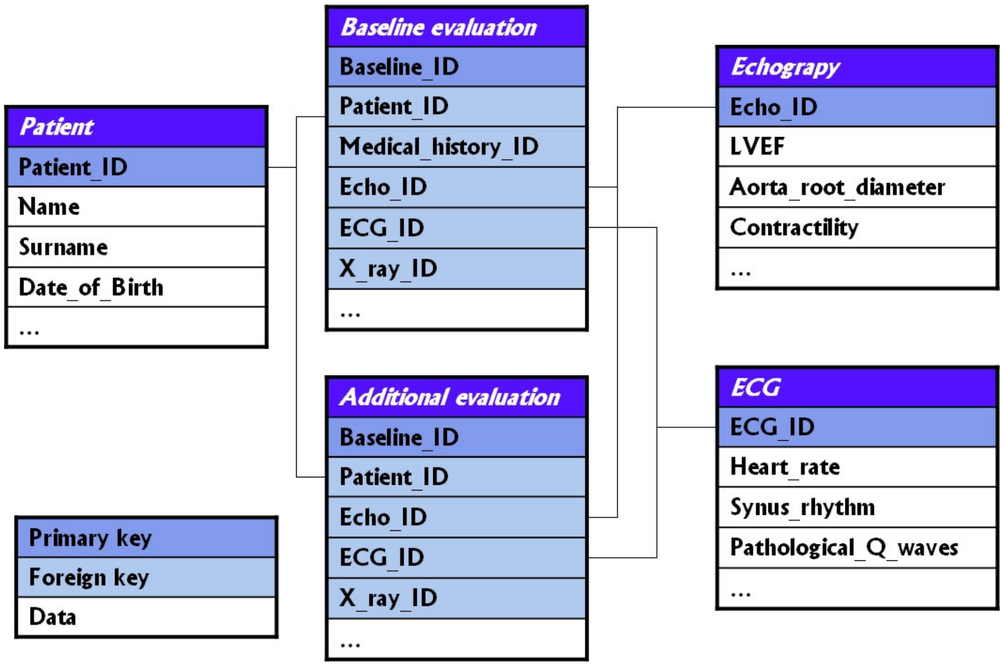


Figure 4-3. Relational data model

Relational model is appropriate for designing systems for transactional processing and less appropriate for data warehouse modelling, especially when concerning the execution of more complex data queries.

Data warehouse models

Data warehouse, as a central storage for data presentation and analysis, should be built upon a conceptual model that is clearly denoting the central entity for analysis. It enables that certain parameters are completely independent while others are functionally and/or transitively dependant. This is appropriate for summating and aggregation. Those features are supported by *multidimensional conceptual model* (dimensional conceptual model) and *dimensional fact model* (dimensional model).

Dimensional fact model is denoting the practical realization of multidimensional conceptual model inside DBMS. In that sense, the multidimensional conceptual model is often referenced as a dimensional conceptual model. Kimball [1] is describing the dimensional modelling as a technique of conceptual and implementation modelling in which standardized intuitive concept enables effective data access.

Multidimensional view on data achieved using such models is structured by placing the data element on the intersection of the dimensions that define it. Such concept is fairly intuitive and comprehendible to the users of the data warehouse.

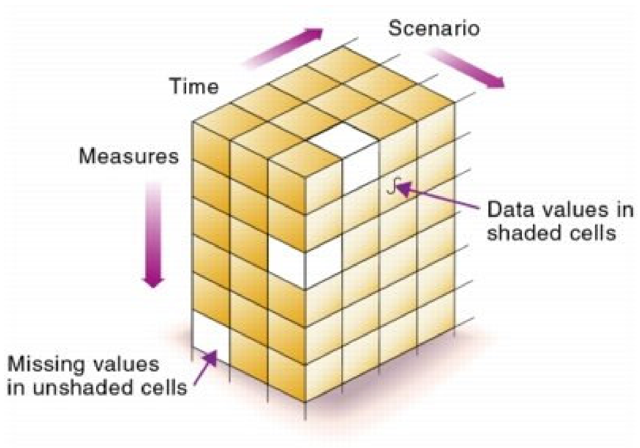


Figure 4-4. Multidimensional data model.

Organizing and storing the data in multidimensional model (Figure 4-4) provides the following features:

- data vividness;
- simple user interfaces to the data warehouse;
- efficient data queries.

The number of dimensions is commonly more than three; the term regularly used is “n-dimensional cube” or “hypercube”. The multiple dimensions are hard to visualize, which brings out the need for using various OLAP tools.

The dimensional fact model is a model in which a data warehouse is represented by a set of *fact schemes.* The basic components of a fact scheme are:

- fact, a centre of interest in a process of decision making (e.g. patients, hospitalizations, laboratory tests),
- measures, numerical attributes used for calculations, and
- dimensions, attributes with finite set of states, determining the lowest data detail granularity.

The schema for a dimensional model has a central fact table and multiple dimension tables. A dimensional model may produce a *star schema* (Figure 4-5) or a *snowflake schema* (Figure 4-6).

In a star schema, all dimension tables are connected directly to the fact table. A snowflake schema allows for one or more dimensional tables not to be connected directly to the fact table, but through other dimension tables.

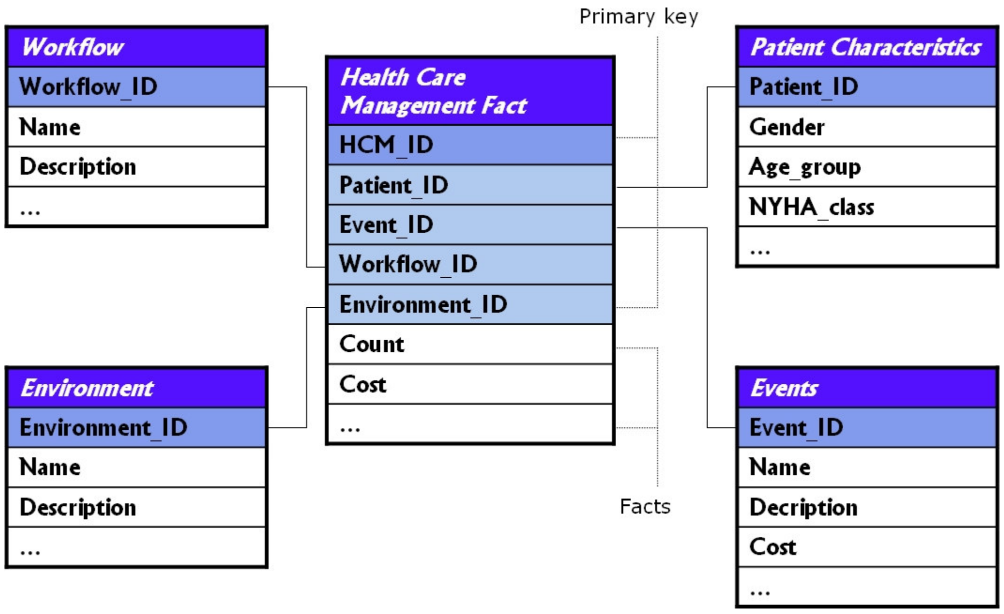


Figure 4-5. Star schema.

Both star and snowflake schemas are dimensional models; the difference is in their physical implementations. Snowflake schemas support ease of dimension maintenance because they are more normalized. Star schemas are easier for direct user access and often support simpler and more efficient queries. The decision to model a dimension as a star or snowflake depends on the nature of the dimension itself, such as how frequently it changes and which of its elements change, and often involves evaluating tradeoffs between ease of use and ease of maintenance. It is often the easiest to maintain a complex dimension by snow-flaking the dimension.

Due to the dynamic nature of facts, the fact scheme nearly always includes the time dimension, whose granularity may vary (minute, hour, day, week, month...).

Finally, when multiple fact tables are required, the "galaxy schema", or "fact constellation" model allows the design of collection of stars.

A standard data warehouse query more often requests summary (aggregated data) than the data at the basic level of granularity. The star schema has a single fact table which stores the data at its single level of granularity, and holds no summary data. The basic level of granularity does not provide efficient data extraction, which draws the need of storing the data with multiple levels of granularity inside the data warehouse. The aggregated facts are commonly stored in a separate fact tables.

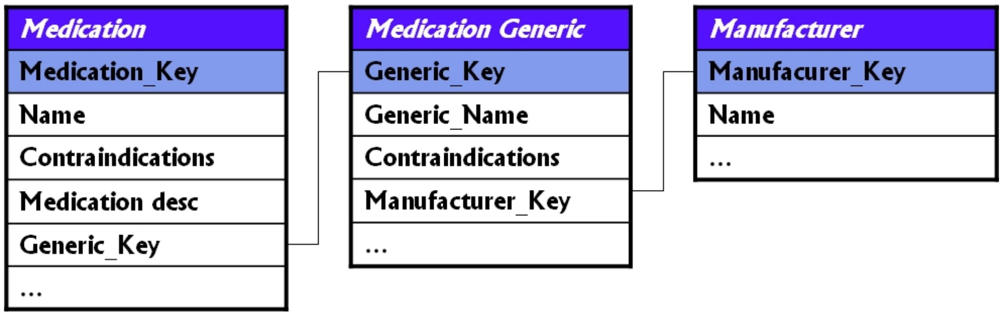


Figure 4-6. Flake schema.

Aggregation handles the need of repetitive calculations and fastens the query execution. However, it increases the required disk space and complicates the data warehouse management. Also, the data loading time is prolonged, since the aggregations are calculated at the loading time.

When modelling the summary tables, the most important is to determine which hierarchical level is the most frequently queried. Later, it is easy to check for which hierarchical elements of the model the response time is longest.

On the implementation level of conceptual fact model there are three generally recognized solutions [3]:

- Relational OLAP (ROLAP),
- Multidimensional OLAP (MOLAP) and
- Hybrid OLAP (HOLAP).

In the ROLAP implementation model, all data is stored into a relational database. Execution of queries from OLAP tools generates SQL queries based on the meta-data. Therefore, OLAP tools are directly accessing the data in the data warehouse. The OLAP tools are expected to perform the analyses which are to be executed promptly and with the same possibilities and efficiency regardless of the chosen dimension. It must be applied a specific design that will mimic the data structure of the multidimensional model in order to enable efficient and simple execution of complex multidimensional queries. These requirements are best met with the star schema dimensional fact model.

MOLAP model stores the data inside the database that is not of the relational type, but that is specifically optimized for storing the data in a storage that supports multidimensional analysis (Multidimensional Database - MDDB). The data is stored in multidimensional fields that enable the conceptual view in a form of a cube. The location of the cell containing the required data is calculated at the query execution time.

HOLAP combines the two above described approaches. Since the relational database has larger capacity and multidimensional database has higher query execution speed, the detailed data is generally stored in a relational database (relational data warehouse), and the summary data that is frequently used in a queries is stored in a multidimensional database.

4.2.2. ETL

The processes of Extraction, Transformation and Loading (ETL) understands a series of complex tasks that refer to identifying the data sets in various data sources, data extraction, cleansing and transformation, quality assurance process, and storing the collected target data into the data storage.

The extracting covers the reading and understanding the source data, selection of the most relevant data for the analysis, selection of data with highest quality, and mapping the extracted data in order to prepare for further processing (Table 4-2). Problems that occur in extracting process are:

- unclear and undocumented data structures and entries;
- data inconsistencies;
- data redundancies.

In the process of analyzing data sources, the problem of inconsistencies among them causes a series of problems. The presence of multiple data sources gives possibility of data entries that are contradictory and also the possibility of data redundancies. The data that refers to the same entity is often diverse in form and content. Each data source has been built on its own purpose and requirements, without taking care of other sources, which leads to uncoordinated database structures.

When the data is extracted, it is being transformed to the appropriate format to enable the loading into the data warehouse. The data is adjusted to the data model that is given in target data warehouse storage. The data should also be checked for its quality and integrated into the storage.

The data extracted from data sources and transformed to fit the warehouse requirements should have following characteristics:

- validity – the extracted data is accurate;
- relevance – the extracted data is relevant for the given problem and analysis;
- consistency – multiple redundant data sources are not contradictory;
- completeness – the extracted data should contain enough information to support the required decision making.

Since the data in the data warehouse is a sequence of data images, the care is taken for the data extracted from different data sources to refer to the same moment in time.

The data cleansing is a process of removing the data inconsistencies and anomalies that could influence the knowledge discovery process. It has a great importance in data warehouse system, since the quality of the KD results is highly dependant on the quality of cleansing process. The data cleansing process includes:

- unification of terminology,
- detection of outliers, and
- handling missing data items.

Table 4-2. Typical ETL operations/mechanisms

ETL Mechanism	Description
Loader	Loads data into the target of an ETL process (in a fact or dimension of a DW)
Filter	Filters and discards non-desired data, and verifies data quality by means of constraints.
Conversion	Changes data type and format or derives new data (derived attributes) from existing data.
Aggregation	Aggregates data (SUM, AVG, MAX/MIN, COUNT, etc.) based on some criteria.
Join	Joins two data sources related to each other with some attributes.
Merge	Integrates two or more data sources with compatible attributes
Wrapper	Transforms a native data source into a record based data source
Incorrect	Reroutes incorrect and discarded data to a different target for the later checking and corrections; it is used in conjunction with *Filter, Loader and Wrapper*

The data cleansing resolves the mistakes that were made during the data entry and enables the consistency within the data itself, consistency with the data in the same data source, consistency with the data from other data sources and consistency with the data that is already stored in the data warehouse storage.

The techniques of data cleansing, transformation and integration are applied iteratively. The logic of the processes is described and specified in order to automate the process to the maximum extent possible, since this meta-data greatly eases the repetition of the extracting process.

In the loading process, the emphasis is on the data quality. In most cases the quality of data is reduced due to the bad quality of data in data sources.

A quick data quality check can be made by inspecting the target data counts. The expected amount of extracted data in most cases denotes that the quality of loaded data is high and fairly reliable. The quality of data inside the warehouse can be checked on regular basis by verifying the random data sets from the storage.

4.3. Process of data warehouse design

The process of designing data warehousing solution for the particular organization or some complex activity is composed of a number of steps. Generic process of the data warehouse design is depicted in Figure 4-7.

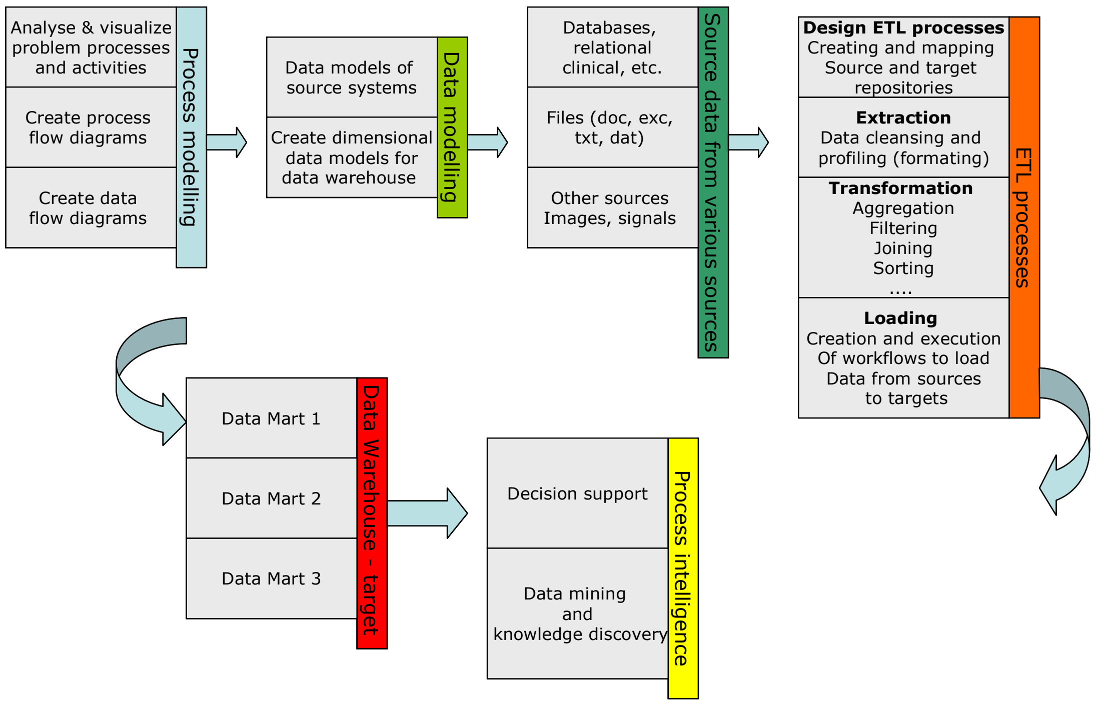

Analyse & visualize problem processes and activities
Create process flow diagrams
Create data flow diagrams
Process modelling
Data models of source systems
Create dimensional data models for data warehouse
Data modelling
Databases, relational clinical, etc.
Files (doc, exc, txt, dat)
Other sources Images, signals
Source data from various sources
Design ETL processes
Creating and mapping Source and target repositories
Extraction
Data cleansing and profiling (formating)
Transformation
Aggregation
Filtering
Joining
Sorting
....
Loading
Creation and execution Of workflows to load Data from sources to targets
ETL processes
Data Mart 1
Data Mart 2
Data Mart 3
Data Warehouse - target
Decision support
Data mining and knowledge discovery
Process intelligence

Figure 4-7. The sequence of processes in data warehouse design.

It starts from the analyses of the organizational/problem processes and activities from which diagrams of activities and related flows of data are constructed. It is actually the process of formalizing the user requirements. After that, data models for source databases as well as dimensional models for the data warehouse are defined. Identification of source and target data models is the necessary step for designing the overall process of data extraction, transformation and loading. Last steps of data warehouse design are oriented towards end user functionalities: design of thematic data marts/data cubes for applications in decision support processes (designing reports, OLAP front-end tools) or for further use in data mining and knowledge discovery.

The process of data warehouse design is not of a linear, straightforward character. Similarly to design of any software, it is an iterative process, as depicted in Figure 4-8. The primary task is always the analysis of the requirements, data, data sources and technical architecture. The complete design process is very much dependent on the problems that are to be solved by the system. In the design part, the emphasis is on the data model design since it is the one that is the basis for the accomplishing the user requirements, and it strongly affects the ETL process and technical environment of the system. The actual implementation comes after resolving all design issues [3].

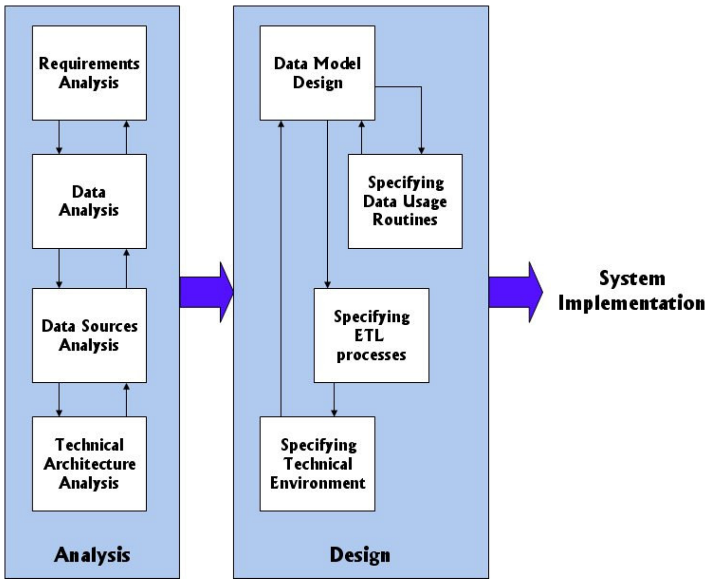


Figure 4-8. Iterative nature of the process of data warehouse design.

At the data warehouse modelling phase, there are two general approaches:

- Inmon [1] – in the data warehouse the normalized data model is dominant, and for the specific subjects (the most analyzed ones) smaller dimensional models are created. In this approach the normalized data model is in most cases constructed by the process of ER modelling (Entity-Relationship). The created dimensional models also contain the summary data.
- Kimball [2] – the complete data warehouse is constructed in one dimensional model. Separate dimensional models, implemented as star models, are connected by the use of shared dimension tables, which are balanced to fit the needs of many fact tables.

4.4. Modern data warehouse and business intelligence solutions

Nowadays there are numerous data warehouse and business intelligence solutions available, including:

- Business Objects

- Cognos
- GreenPlum
- IBM DB2 UDB8.2
- Information Builders
- JasperSoft open source business intelligence
- Microsoft SQL Server 2005
- Microstrategy
- Oracle 10G R2
- Pentaho Open Source Business Intelligence
- SAS

Nevertheless, it is difficult to find a single suite that covers all data storage and data analysis requirements. The goal is to reduce the number of packages that one needs for a concrete application. An illustration is given Figure 4-9. In this example, five suites with overlapping functionality can be consolidated into three separate standards: one for data warehousing; one for BI that uses ranging from operational reporting through to dashboards; and one for more sophisticated data mining.

The BI industry is a complex one, and industry experts differ on how exactly to draw the lines between different types of BI functionality, but there is a broad consensus about the following BI software categories:

- **Database management and data warehousing software**. These products provide the functionality for storing information efficiently for end-user access. Although these products are sometimes viewed as separate from the rest of the BI product choice, they are clearly a foundation of the overall BI system.
- **Data integration software**. These products are also known as extraction, transformation, and loading (ETL) software. They are designed to combine information from multiple sources to feed a database or data warehouse.
- **Operational or embedded reporting software**. These products provide the reports for the systems used to run the organization on a daily basis. They are often seamlessly embedded into applications. Examples of operational reports include bills of lading, invoices, and payroll slips.
- **Enterprise reporting software**. These products distribute reports efficiently to large groups of users inside or outside the organization. They include additional security, management, and distribution features that are lacking in simple operational reporting.

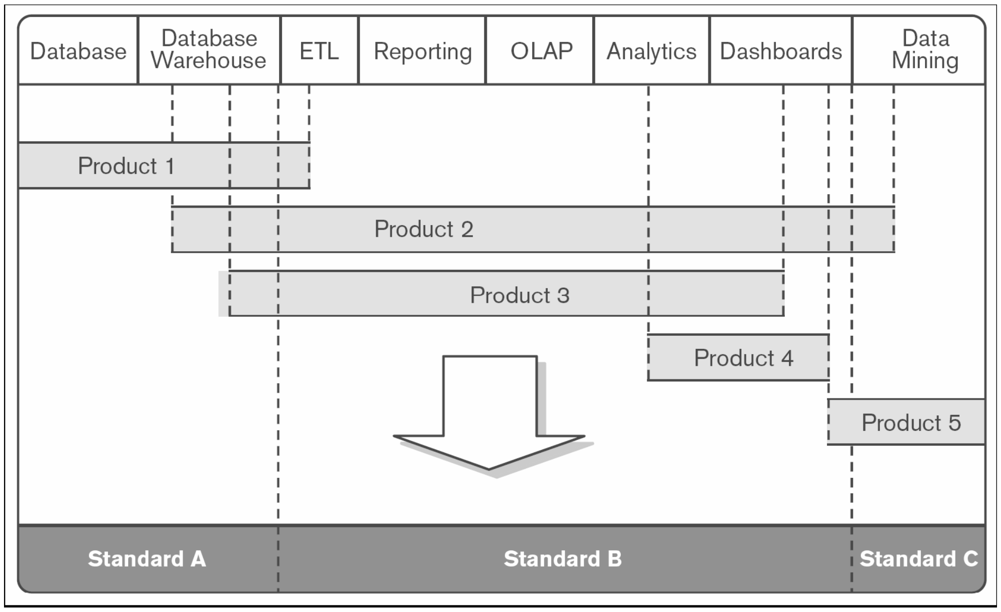


Figure 4-9. A set of unambiguous criteria should be used to reduce BI functional overlap [4].

- **Querying and analysis**. These products are designed to give business users and analysts autonomous and interactive information access. Users are not limited to the subset of information available in a particular report or cube, but can ask new questions and analyze any of the information stored in the database or data warehouse.
- **Analytic applications**. These products are designed for information analysis in a particular area, such as human resources or finance.
- **Dashboards and scorecards**. These products gather key corporate performance indicators and display them in a single interface. Typically associated with executive users, they may also support goal setting, collaboration, and standard business methodologies like Balanced Scorecard.
- **Data mining**. These products are used for sophisticated, automatic analysis of large amounts of data with lots of different variables. Tuning and interpreting these results requires specialized expertise in statistical modelling.

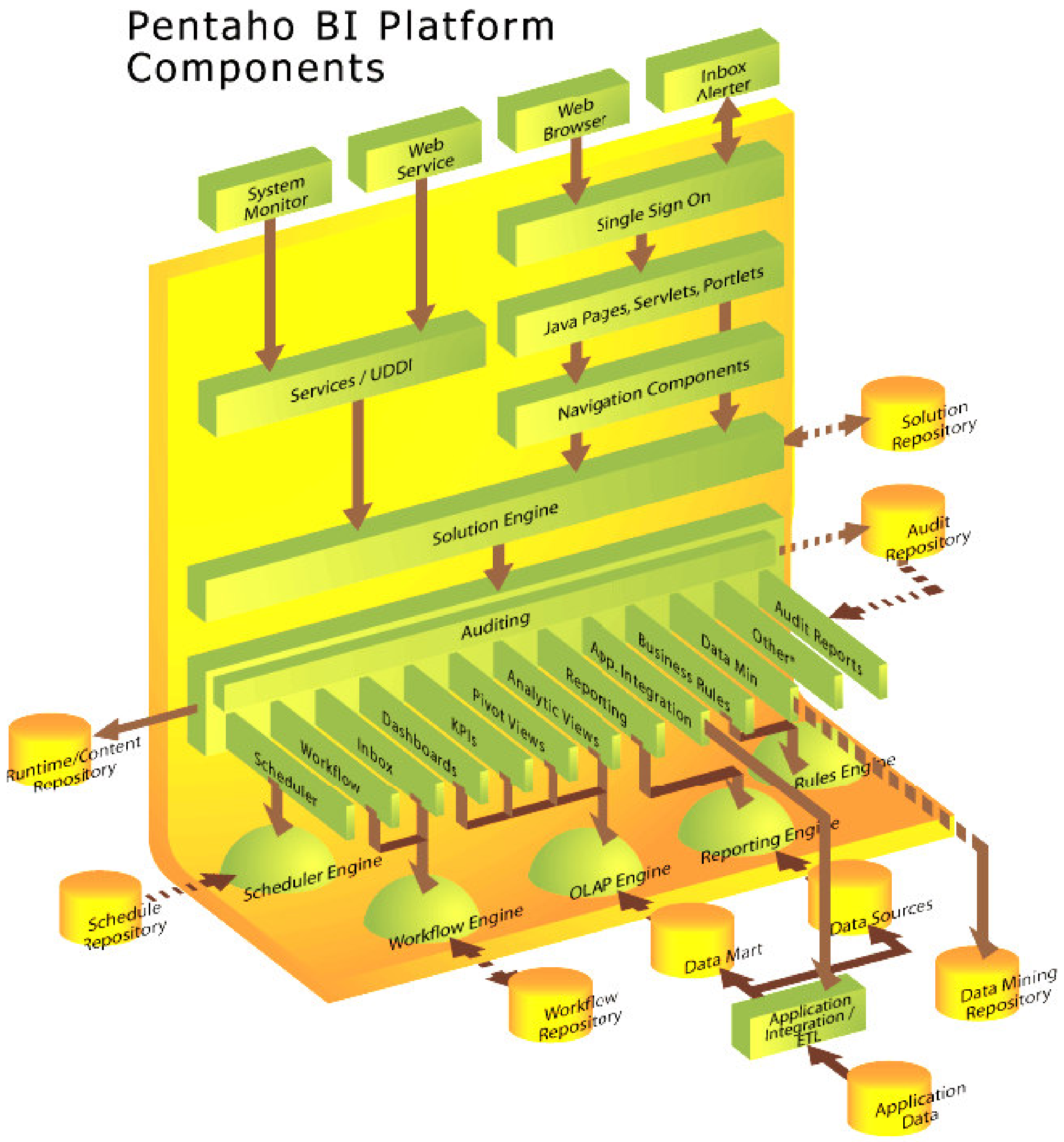


Figure 4-10. Pentaho server architecture diagram (from http:// kent.dl.sourceforge.net/sourceforge/pentaho/Pentaho_Technical_Whitepaper-1-6.pdf)

We have chosen PENTAHO Open source BI platform as a basis for prototyping the HF warehouse architecture, as well as for validation and testing of realistic use case scenarios [5]. Overall Pentaho platform server architecture is presented in Figure 4-10.

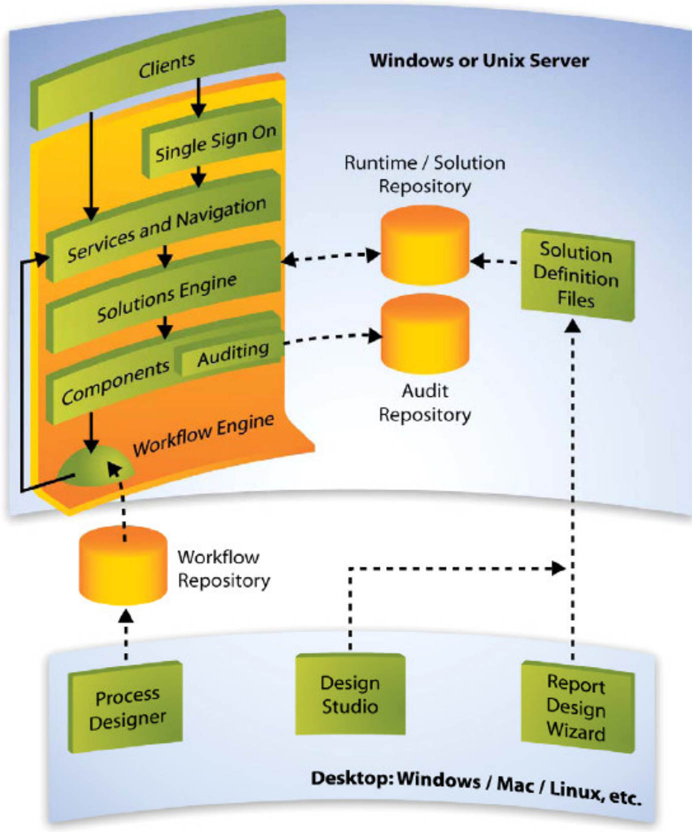


Figure 4-11. An architecture diagram for the Pentaho BI Workbench (from http:// kent.dl.sourceforge.net/sourceforge/pentaho/Pentaho_Technical_Whitepaper-1-6.pdf)

4.4.1. PENTAHO components

The Pentaho platform, the core architecture and foundation of the Pentaho Open BI Suite, are process-centric because the central controller is a workflow engine (Figure 4-11). The workflow engine is using process definitions for defining the business intelligence processes that execute within the platform. The processes can be easily customized and new processes can be added. The platform includes components and reports for analyzing a performance of these processes.

The design and administration workbench of Pentaho Design Studio is an Eclipse-based desktop workbench that provides:

- Easy to use design tools for reports, dashboards, analytic views
- Workflow process designer
- Business rules editors
- Data mining console for data preparation
- OLAP modeling tools

Pentaho technology can be embedded into standalone or server-based Java applications. Flexibility of the platform is a key advantage of the platform, thus only the Solution Engine and components package must be installed. Otherwise, only those components, engines, and repositories that are required for the specific application need to be configured. More specifically, these components are optional:

- Workflow engine, workflow repository and runtime repository
- Auditing and audit repository
- Application Integration / ETL for data extract, transformation and loading
- User interface components
- Solution repository and solution definition files

4.5. Data warehousing and OLAP in health care

Although DW&OLAP technologies are quite mature, their applications in health care are rather sparse. The basic needs addressed by introducing DW technology were more efficient overall care management, i.e. reduction of costs and increased quality of care. There are a number of applications in healthcare where DW and OLAP technologies might be of benefit [6, 7]:

Clinical

- Outcomes analysis
- Providing feedback to physicians on procedures and tests
- Decision support (e.g. single patient decision support)
- Physician performance analysis

Financial and organizational (health care management)

- Pairing clinical and financial records to determine cost effectiveness of care
- Staff planning
- Analysis of care delivery
- Information to support audits and external reports
- Resource utilization reporting
- Identifying quality indicators for best practices

Healthcare processes typically involve a series of patient visits and a series of outcomes. The modelling of outcomes associated with these types of healthcare processes is different from and not as well understood as the modelling of standard industry environments. For this reason, the typical multidimensional data warehouse designs that are frequently seen in other industries are often not a good match for data obtained from healthcare processes. Dimensional data warehouse modelling has been used in standard industry for years for decision support in such areas as transportation, production, sales, and marketing. Healthcare is well behind in the area of data warehouse management and decision support and needs to move forward in this direction.

In contrast to typical processes in industry, or retail business, which follow a linear pattern in which products, goods or customer orders move through a series of steps from raw material to finished product or from customer order to delivery, the process of patient care in healthcare is of an iterative, cyclic nature. Data is centred on patient treatment and is generated within the healthcare circle. The treatment is measured and the data is generated by different processes and different organizations around the circle. The healthcare process involves a series of follow-up visits and a series of outcome measurements and this makes outcome analysis extremely difficult. It is therefore necessary to develop approaches that are beyond the standard methods used for multidimensional data warehouse design.

In [6] authors present a successful DW integration project into an existing health care information environment. In [8] the authors implemented a slight variation of dimensional modelling technique to make a data warehouse more appropriate for healthcare outcome research. The authors present a case study of a multidimensional database design for a data warehouse of healthcare rehabilitation outcomes at the Centre for Rehabilitation Service at the University of Pittsburgh Medical Centre. The primary purpose of the data warehouse is to support various outcome analyses of outpatient rehabilitation therapies. This paper presents a multidimensional database design that can be used as a blueprint for the development of a data warehouse for healthcare decision support.

In [9] the authors describe the possibilities of using data warehousing and OLAP technologies in public health care in general and their experience gained during the implementation of a data warehouse of outpatient data at national level. In addition to numerous reports actually produced using the traditional standard tools (at PIHRS – Public Health Institute of the Republic of Slovenia), the authors want to give health care data analysts the possibility to perform interactive exploration, ad-hoc analysis and discover trends in a user-friendly way. The authors describe the steps performed when building the data warehouse of outpatient data and then the end-user applications for data exploration and analysis. The authors showed that data warehousing and OLAP technologies are suitable technologies for building decision support systems in the domain of public health care. Their future plan is to build a data warehouse of hospital visits data and apply data mining methods for advanced data analysis.

4.6. Towards knowledge warehousing in health care

Data warehousing, as an integral part of decision support systems (DSS) provides an infrastructure that enables organizations to extract and store data with intention to empower their decision makers (or knowledge workers) with information that allows them to make decisions based on solid factual knowledge. However, data represents only a part of an organization's knowledge. The majority of knowledge exists as knowledge in the form of experience and tacit knowledge of organization's employees. Recently, a concept of knowledge warehousing [10] as an extension to data warehousing, has been introduced. Knowledge warehouses should facilitate capturing and coding of the knowledge, its retrieval from other sources (data, texts), and sharing across the organization. Through these capabilities knowledge warehouse concept expands also the purpose of future DSS in the direction of knowledge improvement.

In contrast to the traditional data warehouse, knowledge warehousing introduces many new aspects. First of all, knowledge capture in contrast to data capture is much more difficult and requires introduction of new technologies as well as architectural changes in the computational platform. While sources of data in the organization are rather obvious, knowledge sources as well as their format are quite disparate. Knowledge of the organization lies in its formal or non-formal procedures (explicit form) and in the experience and inherent knowledge of the workers (tacit form). Traditional data warehousing encompasses data collected through different processes, but also reports or data mining models which are inferred from the data, representing explicit forms of the knowledge. In processes related to the automated decision support, knowledge is explicitly represented in the form of set of rules, decision tables or more sophisticated algorithms which are utilized through deployment of expert systems or intelligent agents.

This broad spectrum of organization's knowledge is difficult to realize using single technology platform, such as relational database, or data warehousing; it rather requires introduction of ontology management tools, knowledge bases and expert systems technology, as well as new modes of knowledge acquisition and presentation.

Ontology management tools, knowledge bases and expert systems technology are technologies used in different forms in a number of so called CGPs (clinical guidelines platforms) like EON, ASBRU, GASTON, which rely to a significant extent on some of the medical informatics standards like HL7, GLIF v3.0, UMLS.

These platforms typically concentrate on the core part of managing and using existing general knowledge of the domain, with primary goal to support decision making process. Problems of data collection, storage and use in knowledge improvement are not in the focus of these platforms.

HFP platform strives in a direction of integrating all this technologies in an efficient way: HFP data warehouse, integrated knowledge base and decision support system, tools that facilitate process of data collection, and elicitation of new knowledge show that the platform has most of the capabilities of a new paradigm: knowledge warehousing platform.

4.7. Bibliography and references

[1] Kimball, R., Ross, M. (2002) *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling* (Second Edition). John Wiley and Sons.

[2] Inmon, W.H. (1996) *Building the Data Warehouse* (second edition). John Wiley and Sons.

[3] Kimball, R. (1998) *The Data Warehouse Lifecycle Toolkit: Expert Methods for Designing, Developing, and Deploying Data Warehouses.* New York, Wiley.

[4] *Implementing Business Intelligence Standards and Competency Centers*, Bussines Objects, (white paper).

[5] *Pentaho – open source business intelligence.* URL: http://www.pentaho.com

[6] Ewen, E.F., Medsker, C., Dusterhoft, L.E., Levan-Shultz, K., Smith, J.L., Gottschall, M.A. (1998) *Data Warehousing in an Integrated Health System: Building the Business Case*. Proc. Int. Workshop on Data Warehousing and OLAP (DOLAP'98). ACM Press, New York (1998) pp. 47-53.

[7] Pedersen, K.V. (2004) *A Framework for a Clinical Reasoning Knowledge Warehouse*. Proc. of the IDEAS Workshop on Medical Information Systems: The Digital Hospital (IDEAS-DJ'04), (2004).

[8] Bambang, P., Scotch, M., Ahmad, S. (2005) *A Framework for Designing a Healthcare Outcome Data Warehouse*. Perspectives in Health Information Management 2005, 2:3.

[9] Hristovski, D., Rogac, M., Markota, M. (2001) *Using Data Warehousing and OLAP in Public Health Care*. Institute of Biomedical Informatics, Medical Faculty, University of Ljubljana. URL: www.amia.org/pubs/symposia/D200369.pdf

[10] Nemati, H.R., Stiger D.M., Lakshmi S.I., Herschel R.T. (2002) *Knowledge warehouse: an architectural integration of knowledge management, decision support, artificial intelligence and data warehousing*. Dec. Supp. Sys. 33, pp. 143-161.

5. Requirements for the HFP data warehouse

Every information system or software development project starts from the specification of requirements detailed through the interaction process with users. Most of the requirements for the HFP data warehouse could be inferred from previous deliverables of the project. We give here a hierarchical breakdown of requirements and functionalities to be satisfied by the HFP-DW component:

General requirements:

- HFP-DW should work in the direction of main HEARTFAID goals and objectives;
- HFP-DW architecture as well as its functional capabilities should enable more efficient enforcement of EBM process principles.

Core HFP-DW requirements

Provide input data formatted appropriately for solving following research problems:

- identification of early predictors of heart failure worsening (heart rate changes, respiratory rate, bioimpedenziometry, body temperature variation)
- identification and evaluation of most relevant prognostic factors (HRV, BPV, baroreflex sensitivity)
- building and evaluation of models for early detection of decompensation events, prognostic models, and models for optimal treatment/therapy

Optional HFP-DW functionality

Health care management decision support requirements

- Provide set of variables, or indicators that will serve the purpose of quantification of HFP benefits on the level of resources utilization, quality of care and overall cost. These can be monitored via OLAP tools, or pre-defined reports.

Single patient decision support requirements

- Provide OLAP analytics for the care team on single patient data through decision support interfaces (constant monitoring data-measures with respect to most important patient state dimensions).

Technological and security requirements

- Web-access, Java technology, open source, interoperability, security and data privacy requirements.

5.1. General HFP data warehouse requirements

HEARTFAID is a research and development project aiming at defining efficient and effective health care organization models for "optimal" management of care in the field of cardiovascular diseases, and HF in particular. HEARTFAID involves the design, development, and validation of an innovative knowledge-based service platform that is able to improve early diagnosis and lead to more effective management of heart diseases. HEARTFAID focuses on the following typical settings: health care facilities, home premises and mobile patient. Data collection of HEARTFAID patients will be performed in each of these contexts, according to specified protocols.

5.1.1. HEARTFAID goals and objectives

The long-term goal of the innovative HEARTFAID platform is the improvement of the processes of diagnosis, prognosis, and therapy provision, providing: (a) electronic health record for easy and ubiquitous access to heterogeneous patient data; (b) integrated services for healthcare professionals, including patient telemonitoring, signal and image processing, alert and alarm system; (c) clinical decision support in the HF domain, based on pattern recognition in historical data, knowledge discovery analysis, and inferences based on the accumulated patients' clinical data.

The core of the platform is the formalization of pre-existing clinical knowledge, generally described in clinical guidelines, but also available in internalized form of domain expert - cardiologist knowledge. This formalization or encoding the knowledge into a computable format represents the core of the decision support system, i.e. its starting knowledge base. Additional component of the platform is the capability to generate new knowledge or revise existing knowledge via knowledge discovery components of the platform.

Based on general goals of the HEARTFAID project we define the structure and functionalities of the data warehousing component within the knowledge warehouse. Conceptual development of traditional data warehouses starts with business process requirements specification. Identifying specific requirements for the health care environment includes specification of processes to be supported by the warehouse, stakeholders or users of the warehouse, and objectives or goals of the organization.

In summary, general goals of the HEARTFAID platform are:

- to support all health care players (general practitioners, cardiologists, but also patients) in the decision making and management of patients with heart failure in particular to improve quality of life for elderly patients;
- from the general management perspective to reduce the number of hospitalizations;

- provide knowledge workers and medical experts with integrated environment to access data and applications tailored for the discovery of new and revision of existing knowledge in prescribed forms.

These goals can also be understood as basic requirements for the realization of the platform. Specific reasoning inherent to health care imposes additional requirements upon general data warehousing functionality [1]. The medical reasoning process in patient treatment rests on general principles defined under so called evidence based medicine principles (EBM).

5.1.2. Evidence based medicine (EBM) principles

The definition of EBM taken from Dr. David Sackett [2] is: "EBM is the conscientious, explicit and judicious use of current best evidence in making decisions about the care of the individual patient. It means ***integrating individual clinical expertise*** with the ***best available external clinical evidence*** from systematic research." Main steps in clinical decision making based on EBM principles are following:

- The patient clinical problem or question
- Construction of a well built clinical question. Clinical questions can be:
    - diagnostics
    - prognostics
    - harm/etiology
    - treatment (non-pharmaceutical: quality of life, habits)
    - therapy (pharmaceutical: drugs, compliance to the therapy)
- Selection of resources and conducting a research
- Appraising the evidence
- New therapy (pharmaceutical: drugs)

With regard to these basic EBM principles, HFP data warehouse will intervene particularly in the *research environment,* offering data collected and structured in such a way to provide the basis for performing medical research related to improving diagnosis, prognosis and treatment of patients in follow-up. Data warehouse and OLAP tools will provide additional *intelligence* support within *medical environment*, providing standard OLAP functionalities over the single patient data within the HFP decision support system.

5.2. Specific requirements for the HFP-DW

General HEARTFAID project goals and EBM principles together define set of general requirements for the conceptual design of the HEARTFAID platform. Specific requirements are related to the medical problem that platform is supposed to address: heart failure syndrome in elderly population, particularly the constant follow-up and monitoring of patients.

In fact, besides other functionalities, HFP will provide a series of knowledge – based services (e.g. alert system, DSS services), designed in order to address afore mentioned medical problems. These services will be developed either via the integration of experts' knowledge in the platform or exploiting the potentialities of Knowledge Discovering in Databases (KDD) techniques.

As described in the deliverable D5, "Medical-clinical processes and requirements in HF domain and formulation of the decision making problems", there is a huge number of possible data mining tasks that could be addressed in the field of heart failure. Each single data mining task requires the analysis of a specific dataset, and each dataset has to be collected and, moreover, requires a certain number of pre-processing operations in order to be available for the application of data mining algorithms. It is important to understand that different datasets can be extracted by different sources, and that they need different sequences of data preparation operations.

Thus, in order to manage the necessity of dealing with several data mining task, HFP data warehouse have to serve as a flexible interface to heterogeneous data sources and it should enable several different data transformation operations.

Regarding data sources, it is obvious that the principal source of data for HFP data warehouse is the central repository of HEARTFAID platform, even if other data sources, i.e. historical data provided by clinical partners, have to be taken in account.

Concerning data transformation operations, Section 8 defines exhaustively which functionalities should be provided by HFP data warehouse. It is noticeable that the listed data transformation operations are not strictly belonging to data warehouse technologies, but lie between data warehouse and data mining fields.

Moreover, it should be taken in to account that in contrast to traditional business intelligence environments, from which DW concept originates, data in health care setting is collected and stored in different manner. In fact, a traditional business process can be thought as a straightforward sequence of activities where at the end products are sold or built. Instead, a health care process can be viewed as a sequence of visits/contacts, without a predefined end, where at each visit patient data and outcomes are registered. Thus, health care data follow a sort of cycling process, and this kind of processes have not been extensively studied from the perspective of data warehouse.

For example, one of the medical problems faced inside HEARTFAID activities is the early prediction of acute decompensation, through the monitoring of the patients "at home" and "on moving". Data originated by telemonitoring will be analyzed with a KDD approach in order to extract an effective model able to predict decompensation probability. Thus, telemonitoring data will have to be stored inside HFP data warehouse, in order to be analyzed and prepared for DM and KD tasks. To date, very few cases of DW implementation for telemonitoring processes are known, and thus it is necessary to develop a new solution in order to store telemonitoring data inside HFP data warehouse, with the purpose to use it primarily for knowledge discovery.

Additionally, beside aforementioned KD tasks, OLAP technologies should be employed for the evaluation of HEARTFAID effectiveness and results. In particular, data related to costs and outcomes could be properly summarized through DW techniques into informative indices or set of performance indicators. Typical health care data needed for system performance assessment are the number and type of laboratory tests performed, days in hospital, number of follow-up visits, follow-up tests performed, drugs consumption. Moreover, some HFP specific data item should be included: real-time monitoring equipment costs, additional communication costs (generated by HFP).

Summarizing, the specific research requirements for the HFP data warehouse are the following:

1. Interface with heterogeneous data source, in particular perfect integration with HFP central repository.
2. Capability of storing different dataset, each one characterized by a proper schema in terms of concept, dimension and facts.
3. Several advanced data transformations functionalities, with the possibilities, where appropriate, to add new transformation functions in order to transform particular datasets.
4. Possibility to obtain the data in the best format in order to apply data mining algorithms.

Having in mind that DW + OLAP technologies are primarily used for decision support in business setting, we propose optional use of HFP-DW also in this context. Therefore, we split our requirements specification into two segments: (i) core requirements or knowledge discovery process requirements (ii) optional decision support functionalities.

5.3. Core requirement: Providing data for knowledge discovery

Data warehouse should facilitate knowledge discovery process in a way to provide the best architecture for easy extraction of parts of the data, with the aim to study problems most interesting from the HFP (EBM) perspective (diagnosis, prognosis, treatment). In particular, HFP will be focused on knowledge discovery related to constant patient follow-up and monitoring data and prediction/prevention of the decompensation events, as defined in D5. OLAP technologies provide decision support using data warehouses or data marts, through real-time exploratory data analysis and reporting tools. In business intelligence world, especially in mature and routine industries like retail business, all relevant data summaries are defined prior to designing data warehouse, because key variables and process performance indicators are well known in advance.

Within HEARTFAID project, more specifically in the patient treatment in follow-up or constant monitoring regime, which represents most demanding part of HFP, performance indicators describing patient state are defined in deliverable D5. However, their importance, the time scale, their mutual dependencies are the subject of research – i.e. subject of knowledge discovery. Therefore, DW and OLAP technology in our case will be primarily understood as the basis for knowledge discovery, and not as an ultimate resource supporting health care decisions. In the process of defining functional specifications for HFP-DW and OLAP we have relied on the capability of OLAP tools to seamlessly provide versatile/diverse output of data for knowledge discovery processes.

Data collected via HCRs, related to patient demographics, personal and family anamnesis, aetiology, follow-up testing and especially constant monitoring data are the basis for medical related knowledge discovery. Of particular interest are constant monitoring data in home and on-the-move environment, as this data is the basis for novel diagnostic and prognostic models that should serve as the basis for future performance of the decision support system and should be used within similar constant monitoring platforms for the prediction of worsening or improvement trends, avoidance of decompensation events via alarms and suggestions or cardiologist interventions. Since models for these situations actually do not exist, this is the main medical-research challenge of the HFP. Constant monitoring data are specified in D5 and consist of:

- Heart rate (Heart rate variability – HRV)
- Respiratory rate
- Bioimpedenziometry (variation of the water content in body)
- Body temperature
- Blood pressure (blood pressure variability - BPV)
- Baroreflex sensitivity (BRS)

Main research problems with respect to the relevant decision making problems for patients monitored at home, or on the move:

- identification of early predictors of heart failure worsening
- identify new early indicators of heart failure improvement
- detect early adverse effects of drugs treatment
- identify new prognostic factors

Research problems stated above require special transformation of measured data with respect to different time scales. It is important to stress that this transformations are performed on the data that is residing in the data warehouse, in contrast to standard transforms performed within ETL process, prior to loading the data into a data warehouse.

5.4. Optional HFP-DW functionalities

Core requirements to be satisfied by HFP data warehouse are related to provision of data for knowledge discovery tasks. However, it is possible to provide other users of HFP with additional (optional) functionalities relying on multidimensional modelling and OLAP tools which are inherent to data warehousing technology. It is proposed that these additional functionalities address two health care problems within HEARTFAID platform: patient follow-up and monitoring data and health care management.

5.4.1. Data marts for patient follow-up and constant monitoring

Part of the decision support system that DW+OLAP technology could address is single patient follow-up or monitoring at home. Using OLAP for single patient data care provider obtains multiple views of most relevant patient data with facts/measures such as heart rate variability, blood pressure variability related to most important dimensions: time, activity, part of the day, environment, etc.

This data put into data cubes for OLAP analysis can be used also as a starting data structures for more specific knowledge discovery process. OLAP functions can provide flexible data arrangement capabilities, especially interesting from the point of view of dynamic (time dimension) data. Using standard set of OLAP functions, user can through exploratory analysis construct data views that emphasize important trends and provide starting data tables for new knowledge discovery problems.

5.4.2. Data marts for health care management

In the HF platform, one of the possible scenarios has character of a traditional business setting: health care management. In order to prove effectiveness of the HFP, overall costs of the health care related to elderly heart failure patients, as well as outcomes have to be monitored and properly summarized into informative indices or set of performance indicators.

The information about effectiveness of the HFP with respect to the care process related to the treatment of elderly patients suffering from heart failure is difficult to assess or quantify without thorough collection and structuring the data from follow-up visits and monitoring at home. Data relevant for health care management should be in data marts/data cubes with facts constructed from diverse costs and outcomes, while dimensions should include varying time dimension, HFP environment, HFP workflow, patient NYHA class, sex and other patient characteristics.

The source of the HCM data resides in health care records (HCRs). Typical data items needed: number and type of laboratory tests performed, days in hospital, number of follow-up visits and follow-up tests performed, drugs consumption, outcomes. Some of HFP specific performance indicators may include: real-time monitoring of the overall patient treatment costs; additional communication costs (related to HFP), outcome assessment (for quality of care). All these data should be accompanied with related costs.

5.5. Technological requirements

The architectural requirements of HFP data warehouse are determined directly from HFP global architecture. In fact all HFP components are implemented in Java and, where applicable, they are open source oriented. Thus, in order to preserve the homogeneity of the whole platform HFP data warehouse should be also implemented in Java and possibly developed starting from a pre existent open source application. Along all the available solutions conformed to these requirements, Penthao BI platform seems to be the most complete and versatile one.

Regarding interoperability, a necessary requirement is the possibility to handle XML based communication, in order to create an interface with the middleware components of the platform.

Performance levels are not the principal issues in the realization of the HFP data warehouse. In fact current DW solutions are able to effectively manage data matrix with billions of data entries; thus, it seems very hard that the quantities of data collected by the HFP will exceed the limits of present technologies.

Finally, in order to guarantee a good privacy level all patients' records stored inside the HFP data warehouse will be anonymous. Security will be established by limiting accesses via password to a restricted number of users, and, if it will seem appropriate, by encrypting stored data.

5.6. User types and requirements

Having in mind HFP users, we have produced a following list of users with their roles and requirements related to the data warehousing components.

- HFP-DW administrator: Administrator will be responsible for maintenance of the ETL tools and processes, database management, and production/and eventual changes of data cubes for OLAP views and reporting, based on data marts.
- Medical staff: Medical specialists (i.e. cardiologist) should have the possibility to work on data preparation for knowledge discovery. They should also be able to use optional functionalities provided by the OLAP tools, related to patient monitoring data. This could enable them to explore patterns of measured data in relation to other patient dimensions and manipulate with data views, which could possibly lead to formulation of novel knowledge discovery problems. The user interfaces to the components should be simple and easy to use.
- Knowledge workers: Should be able to query HFP-DW via KD-ETL and produce datasets for knowledge discovery tasks. Also, they could use OLAP tools for exploratory data analyses and eventual preparation of new datasets for knowledge discovery using OLAP tools. They should be allowed to save new data locally.

- Health care management staff: (optional) Based on health care data marts, health care managers could use OLAP tools to produce tailored views to health care management data, or they can be provided with tailored health care management reports, produced by HFP-DW administrator.
- Public – Health care: (optional) HFP-DW could provide basic data summaries of HFP for public and provide free access to a subset of public health care management reports, based on health care data marts.

5.7. Bibliography and references

[1] Pedersen, K.V. (2004) A Framework for a Clinical Reasoning Knowledge Warehouse. Proc. of the IDEAS Workshop on Medical Information Systems: The Digital Hospital (IDEAS-DJ'04), (2004).

[2] Sackett, D. (1996) Evidence-based Medicine - What it is and what it isn't. http://www.cebm.net/ebm_is_isnt.asp.

6. Functional specification of data warehousing within HFP

Starting from the detailed specification of the requirements on the data warehouse, we construct a general architecture of the system. Figure 6-1 depicts basic architecture of the HFP data warehouse.

The first part of the DW architecture consists in data acquisition tools. As described in chapter 5, HFP data warehouse should receive data both form HFP central repository and external data sources. Thus ETL functionalities will comprehend:

1. Possibility to require and obtain data from HFP central repository via structured query (e.g. via XML messages)
2. Possibility to acquire data recorded in text files or in common relational data base systems (e.g. Oracle or Access DBs).

After the phase of data acquisition, HFP data warehouse should store the data. In the relational database and/or data mart as depicted in Figure 6-2. The export of the data for a specific data mining/knowledge discovery task will be performed by the KD-ETL component whose functions are discussed in Section 8.

Data stored in specifically designed data marts will be accessible by OLAP tools for visualization, manipulation and data exploration via graphical user interface (GUI). The GUI should have the following functionalities:

1. ask the OLAP engine in order to obtain the preferred data view;
2. apply data pre processing components, in order to properly transform the data;
3. allow the user to export the data in several output format (e.g. text file, spreadsheet, DB file).

Moreover, GUI should be based on web technology, in order to ensure a good integration level with the HFP user interface. The idea is to develop the HFP data warehouse GUI as a set of web based dashboards and then to integrate these dashboards as frames in the general HFP User Interface.

From this general overview, it should be evident that expected final users of the HFP data warehouse should be expert data analysts i.e. data mining experts or medical personnel interested in research topics. In particular, access to HFP data warehouse functionality should be allowed only after password checking. More traditional statistical functionalities, for non expert users, are planned to be provided directly by HFP user Interface.

Functional roles and capabilities of different components of HFP DW depicted in Figure 6-1, are given in following subsections, as well as ETL processing description, users and usage scenarios, and performance related information.

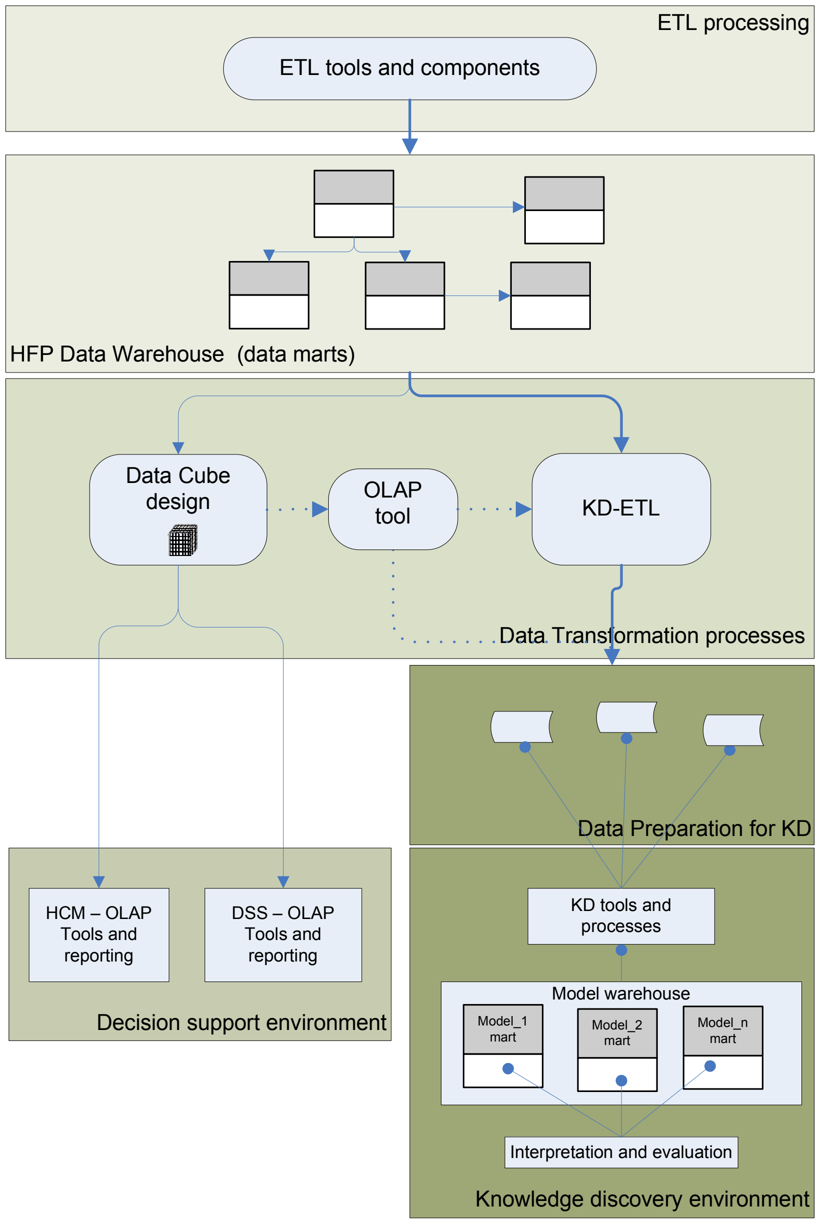

ETL processing
ETL tools and components
HFP Data Warehouse (data marts)
Data Cube design
OLAP tool
KD-ETL
Data Transformation processes
Data Preparation for KD
HCM – OLAP Tools and reporting
DSS – OLAP Tools and reporting
Decision support environment
KD tools and processes
Model warehouse
Model_1 mart
Model_2 mart
Model_n mart
Interpretation and evaluation
Knowledge discovery environment

Figure 6-1. Basic architecture of the HFP data warehousing and its components.

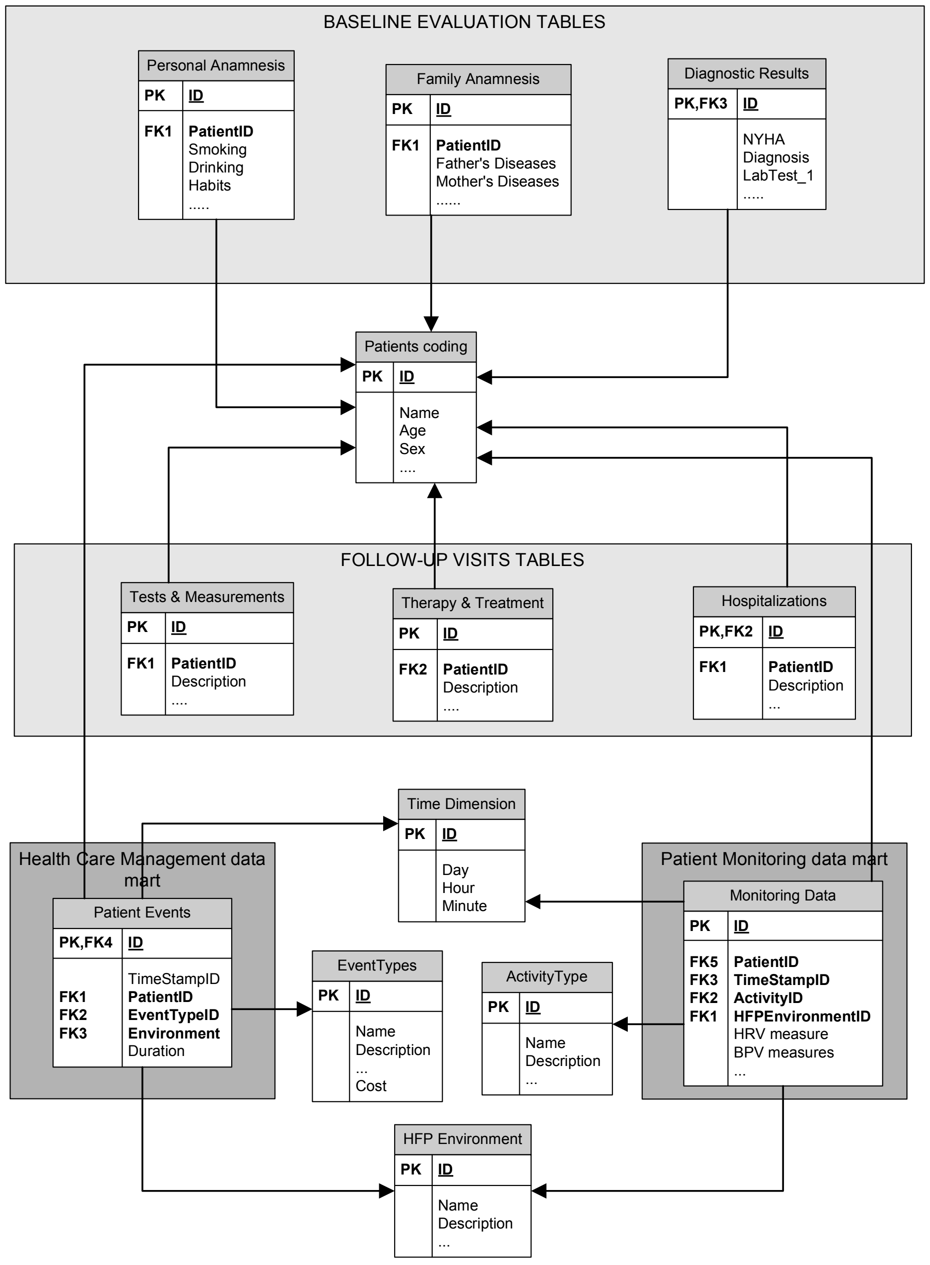


Figure 6-2. HFP data warehouse storage.

6.1. Data sources for HFP-DW

Data sources for HFP-DW are within diverse medical environments and are expected to be in diverse forms, which will be addressed herein as *clinical data repositories* (CDR). Main characteristic of CDR is that they are *patient centred*. In contrast, HFP-DW has to be *research centred*, and due to that reason, HFP-DW will contain a subset of data available in CDRs in a structure supporting research and eventually decision making. The format and content of the source data for the HFP-DW will generally follow *case report form (HFP-CRF)* specification. General structure of HFP CRF is given in the Table 6-1. Detailed specification of data in HFP-CRF is given in deliverable D9.

Table 6-1. Structure of the data in HFP-CRF.

Patient coding (ID, date entered, gender, date of birth)

Baseline evaluation	**Follow-up visit**	**Final evaluation**
Cardiovascular history	Changes in CHF status since the last visit	
Other medical history	New occurrence or change in cardiovascular status	New occurrence in cardiovascular status
Lifestyle information	Other changes in health status	Other changes in health status
Anamnesis		Anamnesis
Family history		
Current cardiovascular therapy	Changes in cardiovascular therapy	Current cardiovascular therapy
Physical examination	Physical examination	Physical examination
Laboratory assessment	Laboratory assessment	Laboratory assessment
Chest X-ray	Chest X-ray	Chest X-ray
12-lead electrocardiography	12-lead electrocardiography	12-lead electrocardiography
Echocardiography	Echocardiography	Echocardiography
24h Holter electrocardiography	24h Holter electrocardiography	24h Holter electrocardiography
Six-minute walking test	Six-minute walking test	Six-minute walking test
	Reason for additional visit	
Quality of life Questionnaire		Quality of life Questionnaire

Constant monitoring data (heart rate, blood pressure, weight, body temperature, respiratory rate, water content)

There are other source data that enter HFP-DW, besides those from HFP CRF. These include costs for different tests, examinations, hospitalization and visit costing information, necessary for health care management related data mart.

6.2. Data warehouse development tools

As given in previous section, source data for the HFP-DW reside within different CDRs. The interface to this source data from the point of view of HFP-DW is provided by HFP middleware. As described in D11, HFP middleware should guarantee easy, everywhere, anytime and safe access to data, and as such will represent a generic interface to CDR sources of the data for HFP-DW. Figure 6-3 depicts generic view to ETL processes of HFP-DW.

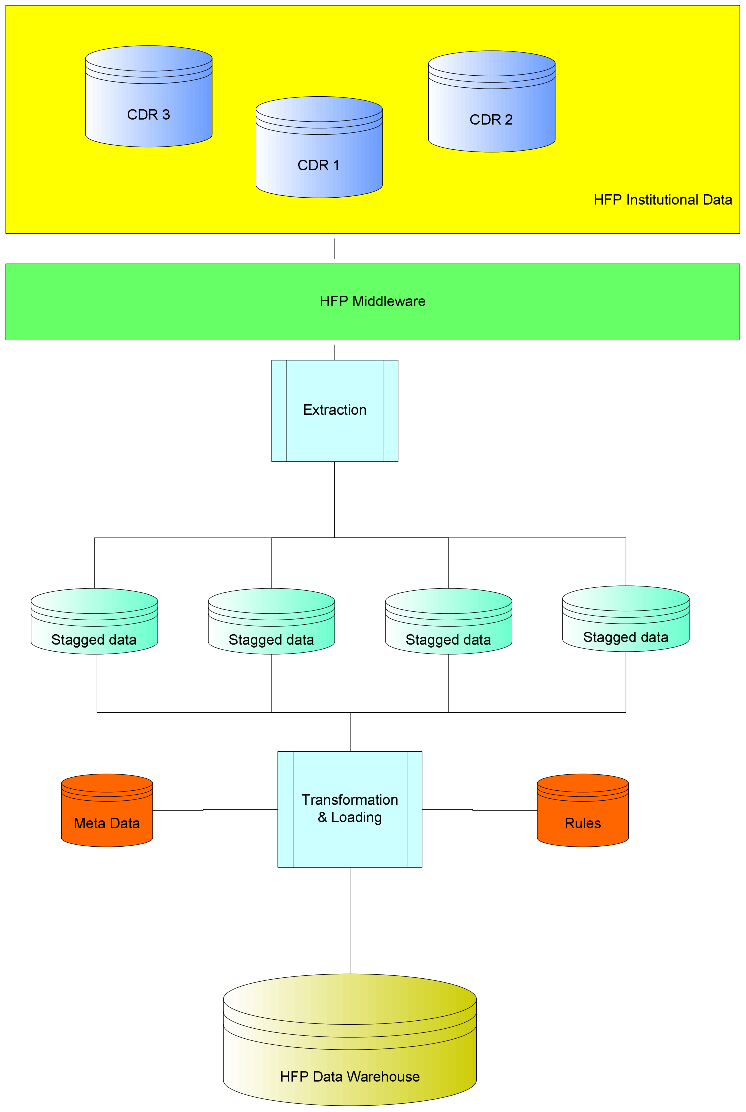


Figure 6-3. Overview of ETL processes within HFP-DW

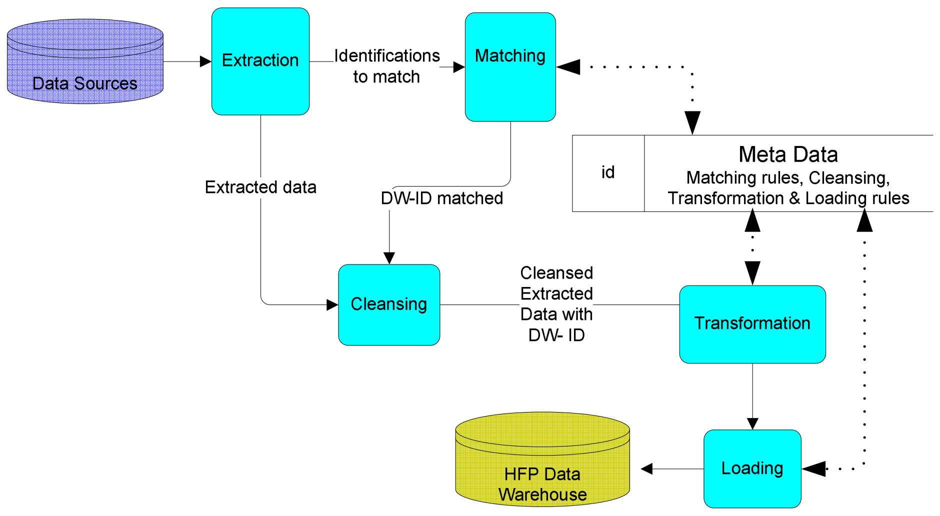


Figure 6-4. Detailed view to HFP-DW ETL process.

More detailed description of the HFP-DW ETL process is provided in Figure 6-4 and the text that follows.

Data extracted from sources are used to produce two types of files: the identification records that have to be matched to the fields in the DW that go to the matching process (common to all source data extracted), and the extracted data that is provided directly to the cleansing process. Both records have an extraction sequence number.

- The common match process manages a database of identifiers used to assign an appropriate identifier to the identification provided by the extraction process. It combines the extraction sequence number and the identifier providing it to the cleansing process.
- The cleansing process cleans the extracted data and combines it with the appropriate HFP-DW identifier supplied by the matching process, using the extraction sequence number to recombine the data with the proper ID. The cleansing process produces the load data files without the original (source) identification data.
- Cleansed data obtain new HFP-id, and are transformed according to the predefined rules.
- Cleansed and transformed data are loaded into a proper table of the HFP-DW, with reference to DW meta-data.

The implementation of this process will rely on the use of Pentaho platform KETTLE environment.

6.2.1. KETTLE environment and tools

ETL software components are responsible for overall process of extraction, transformation and loading the data into the HFP-DW. As a production tool for the development and maintenance of all ETL related processes we plan to utilize KETTLE (acronym for "Kettle ETTL Environment") environment, composed of a set of ETL tools forming Pentaho Data Integration platform. KETTLE provides all functionalities needed for effective construction, maintenance and administration of ETL processes within HFP-DW. Components within KETTLE environment are: Spoon, Pan, Chef and Kitchen.

Spoon is a graphical user interface that allows design of transformations that can be run with the Kettle tool Pan. Pan is a data transformation engine that is capable of performing a multitude of functions such as reading, manipulating and writing data to and from various data sources. Figure 6-5 depicts one transformation process designed using spoon.

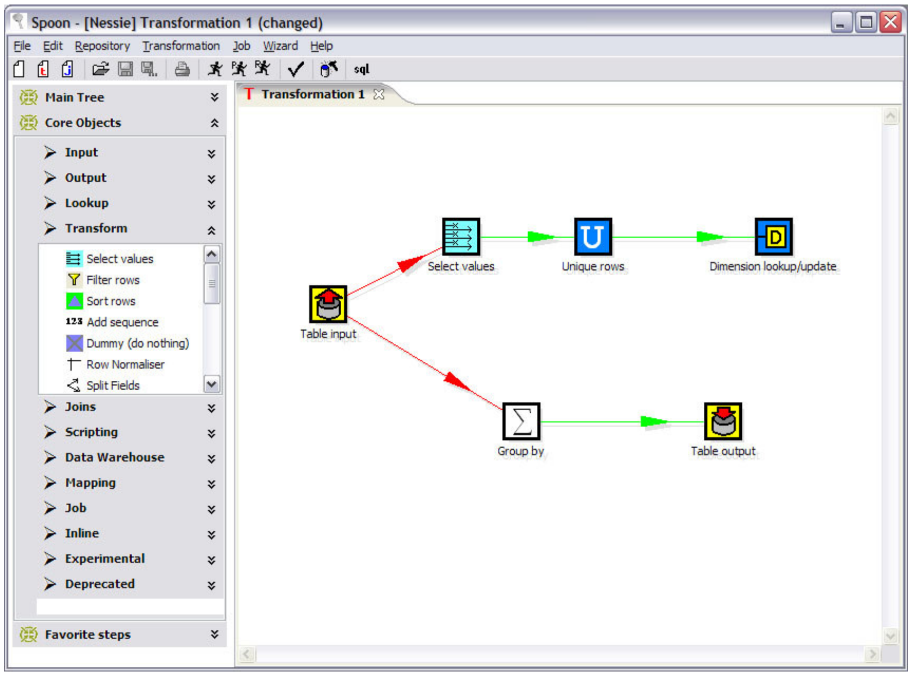


Figure 6-5. Transformation process designed using Spoon application.

Pan is a program that can execute transformations designed by Spoon in XML or in a database repository. Usually transformations are scheduled in batch mode to be run automatically at regular intervals.

Chef is a graphical user interface for the design of jobs that can be executed with the Kettle tool Kitchen.

Kitchen is a job execution engine that is capable of performing a multitude of functions such as: execute transformations, execute jobs, verify file existence, get files using FTP, SFTP, HTTP, etc.

6.2.2. Performance issues related to KETTLE environment

KETTLE provides connectivity to more than 15 DB systems (MySQL, Oracle, MS Access, MS SQL Server, DB2…). All the components need JRE V 1.4 or higher to execute. KETTLE environment can be installed on most of the available operating systems including Linux and MS Windows.

6.2.3. HFP-DW metadata

The metadata is a form of dictionary defining all the objects contained in the data sources and in the data warehouse. The metadata is also used to define extraction, cleansing, transformation and aggregation rules for files and data sources before they are loaded into the data warehouse. It also contains basic user profiles.

6.2.4. Data refreshment

The data warehouse will be refreshed on a periodic basis, but refresh rate will depend on the type of data involved. The care has to be taken in choosing the exact time for the refreshment, to minimize negative impacts on users since the data warehouse must be shut down during that time.

Table 6-2 shows the generic frequency of refresh for each data source. Adhering to this refresh schedule should ensure that different data subjects in the HFP-DW are synchronized, thus ensuring the basic data warehouse consistency. The source and target data specifications given in the Table 6-2 will be available in the meta data, as well as the last update dates for the data warehouse and data marts. It is important to stress that in the process of refreshing the data, all ETL processes will start from the existence of patient coding in HFP-DW, prior to extraction, transformation and loading of other data from the source databases. Therefore, patient coding data for the new patient in HFP are entered first to the HFP-DW, and only in subsequent refresh operations, source DB is inquired of the existence of other types of data.

Table 6-2. HFP-DW data sources and targets with basic refresh rates.

	Source	HFP-DW target (table / data mart)	Refresh rate	Comment
1	Patient enrolment	Patient code	Daily	First to enter HFP-DW
2	Baseline evaluation	Diagnostics, Anamnesis, Quality of Life HCM* data mart	Daily	Second to enter HFP-DW
3	Follow-up visit	Follow-up HCM data mart PM** data mart	Daily	Only after 1 & 2
4	Final evaluation	Follow-up HCM data mart PM data mart	Daily	Only after 1 & 2 & 3
5	Constant monitoring	Constant Monitoring PM data mart	Every (2-6) hour(s)	Only after 1 & 2

* HCM data mart – Health Care Management data mart
** PM data mart – Patient Monitoring data mart

6.3. HFP-DW user management and security functionalities

There are different users of the HFP data warehouse**.** Their roles and use case scenarios are the basis for their access management to different functionalities of the HFP data warehouse. Table 6-3 maps different users to the functionalities of the HFP data warehouse with associated access paths and security mechanisms.

Table 6-3. Mapping of HFP users to HFP-DW components and associated access paths and security mechanisms.

User type	HFP-DW functionality	Access path	Security mechanism
Administrators /programmers	ETL tools	WEB browser, SSH	Password protection
	DB management tools	WEB browser, SSH	Password protection
	KD-ETL	WEB browser, SSH	Password protection
	OLAP tools	WEB browser	Password protection
Knowledge workers	KD-ETL front-end, OLAP tools and views	WEB browser	Password protection
Care team (medical specialists)	KD-ETL front-end, OLAP tools and views, Patient Monitoring	WEB browser	Password protection
Patient (patient relatives)	Patient Monitoring dashboard	WEB browser	Password protection
Health care management	OLAP tools and views	WEB browser	Password protection
Public	Health care management reports	WEB browser	Free acces

6.4. Rules for ensuring data and systems security and integrity

The security in the information technology domain is commonly understood to mean the protection of an information system's assets such as the information and the information processing resources.

Specifically, security has been defined as the preservation of the confidentiality and integrity of the information systems and the data that they maintain as well as ensuring the accountability and availability of the services and data. Security measures also include detection, documentation (logging) and countering mechanisms against above threats.

Data warehousing system and data preparation utilities within HFP are not critical with respect to security issues, as are other HFP components, primarily because they are designed to support research segment of the healthcare cycle, rather than direct patient care.

As depicted in the Table 6-3, most of the user access to data warehouse components and data is protected via authorization mechanisms (password protection) and strong authentication and secure communication mechanisms (SSH) for critical layers such as server system administration.

It is also seen from the Table 6-3 that user access control is enforced by mapping users to secure applications roles. The data warehouse system ensures that it is only the trusted application (code package) implementing the role that determines the correct access conditions.

Data warehouse administrators should follow good security practices. For example, they should always:

- Revoke privileges from public accounts
- Change any default passwords after installation
- Set the proper permissions. If permissions are incorrectly set in the operating system, there may be a security loophole

6.4.1. Data integrity

Data integrity/data quality is defined as the state of completeness, consistency, timeliness, accuracy and validity that makes data appropriate for a given use. Alternatively stated, data integrity refers to whether the data is reliable, accurate, complete and relevant for its intended purpose.

Data warehouse integrity mechanisms include system integrity and database (relational database) properties such as entity integrity, referential integrity, processing integrity and adherence to the rules. System integrity should ensure that the data inserted into the data warehouse are the same as those retrieved by other components. Also, data must not be altered or deleted by an unauthorized user. Data warehouse administration must assure that data adheres to the rules. Inserting or updating mechanisms have to adhere to predefined rules for each column in each entity, and any attempt to violate these rules must fail. Integrity constraints and database triggers can be used to manage a database's data integrity rules.

Referential integrity must ensure that a rule defined on a column or set of columns in one table, match the values in a related table (the referenced value). Referential integrity rules should dictate what types of data manipulation are allowed on referenced values and how these actions affect dependent values.

Finally, data should be copied (backup) as a safeguard against unexpected data loss and application errors. In case of loosing the original data, it is possible to reconstruct it by using a backup.

6.4.2. Data privacy

Within DW data is completely anonymous and not linked to any identifiers leading to personal identification data. In the process of ETL personal data that first enter DW is identified by unique pseudo-ID supplied by the source without personal identification data (e.g. name, address). This pseudo-ID is then replaced by the newly generated unique DW-ID. Source pseudo-ID and DW-ID are mapped into a protected MetaData file which makes re-identification practically impossible.

Re-identification based on data matching process solely from data in data warehouse is not possible, without having access to source data.

6.5. Testing and validation strategy

Before the implementation of the HFP-DW components within the production HF platform, all the components should pass the process of validation and testing. Generic steps for the testing include: unit, functional and acceptance tests. *Unit testing* is performed by the programmers, and test cases are also designed by programmers. In principle, test cases must thoroughly test every line of the programmed code of the unit. Extraction units should therefore use samples from real (production) sources of data, and cleansing and transformation units must use the corresponding output of the extraction processes for their testing.

The main purpose of the *functional testing* is to validate that all the components in the environment are producing results as described in functional specifications. Functional testing also addresses inter-functional and overall performance issues. These tests are performed by the developers of functional specifications. Test cases for functional testing should include some use-case scenarios that are encountered in the production environment of the HF platform. This series of tests are performed within the development environment, using a small subset of data from production-level source databases.

Acceptance testing is generally done by the users of the HFP-DW (knowledge workers, medical staff, and patients). A formal series of tests must be constructed to test practically all the processes of HFP-DW components in the production use. Tests are performed within prototype production HF platform environment.

7. Data understanding, transformation, and preparation for knowledge discovery

Data understanding, transformation, and preparation are steps between data warehousing and knowledge discovery algorithms. They are a set of very different actions with the goal to transform the available DW data into the form that will enable effective execution of knowledge discovery algorithms. The successfulness of data preparation determines the quality and usefulness of the results obtained by the knowledge discovery process. It means that every action that can improve the final quality of the induction process can be included and is welcome as the part of the data preparation process. It also means that data preparation is iterative process in the way that results of previous knowledge discovery results may influence the procedures that will be included in the following data preparation processes.

By different data preparation we can influence not only the quality but also the type of the result that will be obtained from the knowledge discovery process. From different attribute subsets we can get different views about the relations present in the data set. In the descriptive induction tasks it is common to analyze the data from very different views and this practically means that we intentionally repeat the knowledge discovery tasks with differently selected induction goals (target attribute) and differently selected input attributes. The data preparation step has the function to prepare these different attribute subsets. But decision about what has to be selected can be done only by humans. This requires deep understanding of the knowledge discovery goals and understanding of properties and meaning of available data that can be used to achieve the goals.

The recognized high level goals are transformed to practical knowledge discovery tasks by selecting different target and input attributes. This is the first data preparation part. Then a series of data preparation steps is executed with the goal to insure good quality of the knowledge discovery process that will follow. These steps include feature selection, feature construction, and noise (outlier) detection and elimination.

7.1. Data understanding

Data understanding includes all human activities related to the definition of the goals of the knowledge discovery process. It is completely related to the domain (problem) that is analyzed and because of that it practically can not be formalized. But it must be noted that domains and data understanding is rarely a single action. We typically start with available expert knowledge and expectations about the results of the knowledge discovery process. But later, results obtained by first knowledge discovery iterations may stimulate reasoning about additional assumptions and goals. The result is that often the data understanding is an iterative process by itself, continuously going on during the complete knowledge discovery process.

Data understanding is strongly related to the process of human evaluation of the knowledge discovery results. Here we do not think on prediction quality results that can be mechanically measured but on human understanding of the meaning of the obtained results. The most relevant results are actually differences between human domain and data understanding before the knowledge discovery process and information received as the result of this process. If these differences can be accepted as significant and relevant then they can stimulate integration of novel knowledge into the HF knowledge base, stimulate changes in data collection process, and/or finally lead to changes in patient treatment. The results of the data understanding process should be appropriately and realistically defined knowledge discovery tasks in order to achieve these goals. And this can be obtained only in strong cooperation of domain and technical experts.

7.2. The classification task

The main part of KDD tasks depicted in D5 can be considered as classification problems, also known as supervised learning tasks.

Other kinds of KDD tasks exist, e.g. clustering or regression problem, but our attention will be mainly focused on supervised learning, because this kind of approach seems to be more appropriate in order to face the statements listed in the D5.
In a classification task, the data file is given as collection of observations (also called examples, instances or cases), used to design the classifier. Each example is a tuple represented by the input vector x, that can be interpreted as the realization of a random vector drawn from an underlying unknown distribution. The elements of the tuple are called attributes (or variables) and each tuple belongs only to a given class.

At the first place the user should determine the class attribute. It is the target of the knowledge discovery process and object for which we search description by other (input) attributes. It must be an attribute that has a small number of discrete values. If the target attribute is described with numerical values, the user must transform it in an attribute with, i.e., only two values corresponding to high and low numerical values (the boundary value should be determined by the user).

Some knowledge discovery algorithms can work only if the target attribute has exactly only two values (the so called binary classification). The multi-class problems can be solved, applying these knowledge discovery algorithms, by their transformation into the series of two-class problems.

Input attributes are all other attributes that can be used in the knowledge discovery process to describe differences that exist between target attribute classes. The knowledge discovery process is actually the search for optimal (logical) functions that can enable discrimination between classes of the target attribute.

Nevertheless, it is a good practice to build two-class problems directly in the data preparation step. In this way we control the complexity of the knowledge discovery process, the results of different algorithms are directly comparable, and the target class which we try to model can be explicitly specified.

Machine learning algorithms construct classifiers and models by combining features. For decision trees and random forest approaches, for instance, the features are input attributes, while for rule learning approaches features are simple attribute based logical expressions (i.e., (attribute A >1) AND (attribute B <0)). Although machine learning algorithms are very efficient in logical combinations, typically they can implement neither numerical operations among attribute values nor non-elementary logical operations, like sum modulo two among a series of binary attributes. Practically the only way to enable introduction of such relations into results of knowledge discovery process is in the data preparation phase when there is a possibility to introduce additional attributes into the data file.

It is well known that the error-free classification is not usually possible, thus the main goal is to find the classifier that generalizes well on unseen data. Among the several factors that can degrade the performance of the obtained classifier there are:

1. a large number of features;
2. presence of outliers;
3. complexity of the classification rule;
4. inappropriate classifier.

It is known that the ratio of the number of examples to the number of features should be fairly large for developing adequate machine learning system. It has been empirically shown in several studies that redundant or non-discriminatory features reduce the ability of classifiers to learn decision boundaries between data of different classes. Thus, there are different techniques, used to solve this problem, as described in what follows. Identification of relevant features is extremely important for classification tasks (since improve the accuracy and reduce the computational costs), as well as for understanding the relative significance of features [1]. Several experimental studies should fully understand the nature of the data before applying a method of feature selection or extraction. Adequate tests on feature redundancy in data, data normality, noise in data, and overall classification complexity of the task should be performed before attempting to select features and perform classification. For this purpose, in the research literature the attribute selection can be distinguished from the feature selection and feature extraction.

7.3. Attribute selection

Attribute selection is the direct consequence of the data understanding process. It refers to the problem of selecting input variables that are most predictive of a given outcome. The variable selection problems are found in all machine learning tasks, supervised or unsupervised. In the recent years, attribute selection has become the focus of a lot of research in several areas of application for which data sets with tens or hundreds of thousands of variables are available. The objective of variable selection is two-fold: improving the prediction performance of the predictors and providing a better understanding of the underlying concept that generated the data [2]. The final result is typically either a model in the language of input attributes or relative relevance of input attributes in the discrimination of classes. In any case, by selecting input attributes we specify the space in which we search for the solutions. Although we are always looking for good models, sometimes we intentionally omit attributes for which we in advance know that they have very strong correlation with the target attribute classes. The motivation is to allow that less strong connections can be detected. It means that although it is possible to select all other available attributes as input attributes, as we do in so called classification induction when we are interested only in good prediction results, in descriptive induction we typically repeat induction process for various subsets of attributes with the intention to detect logical connections between the target attribute and the selected input attributes.

7.4. Attribute construction from time sequences

Time sequence is a collection of observations made sequentially in time. Such data need some transformation before entering the knowledge discovery process. The reasons are that time sequences can be of variable length, what is unacceptable for standard prepositional knowledge discovery algorithms, and that extraction of properties describing sequences as a whole is necessary. Single measurements bring valuable information, but much more information is contained in the picture that all these measurements create together.

In order to describe a sequence of values as a whole, we use means of descriptive statistics. Potentially we can generate many attributes describing sequences. Some of them describe central tendency like mean, median and mode, some of them describe statistical variability like range and variance. We can measure concentration through entropy and Gini coefficient. Shape of the graph can be described by moments, skewness and kurtosis. Linear regression can define the trend. Relevant information can be obtained also by transformations like discrete Fourier transform and wavelets.

Not all information extracted from a sequence of measurement is relevant for some specific problem. For example, how measurements changed over last hours is not relevant when doing a long term prognosis. But if we want to predict acute states than only the last few measurements are really relevant. Problem is that these last few measurements will stay concealed if we look in a time-frame of one year.

In principle we must make difference between short and long data sequences. Both of them can be numerical and categorical. Automatically collected data is typically in a form of long numerical sequences while manually collected data is in a form of short sequences.

In long data sequence, general trends and short "events" consisting of a few successive characteristic values are important. Long sequences may contain erroneous, missing, or inconsistent data but this is not so important because no decision can be based on a single measurement. In contrast to that, in short sequences every data point is important and even more important are the differences between two successive points. Additionally, global and local trends may be relevant. Important "events" are characterized by significant difference between data points and local trends.

Transformation of numerical sequences is typically based on statistical properties (i.e. mean, range). The number of such descriptors can be large due to the fact that numerical sequences can be pre-process in many different ways in order to eliminate offset and linear trends as well as to allow amplitude scaling [3]. For categorical sequences relevant are descriptors like number of points, mode, relative frequency of mode, and the number of points in which values change.

For all medical sequences, the notion of time is important. Constructed features should describe the beginning of the sequence (*starting window*), its end (*ending window*), and differences between the complete sequence and its parts (windows). It follows from the fact that for detection of relevant relations it may be interesting to know the original status of the patient, its final status but even more relevant can be differences between these two descriptions [4]. In this setting it is important to define the length of the window. Automatic tools can not help in this respect. The suggestion is to use medical expert knowledge for the definition of optimal windows for each task. The only alternative is to define a few very different windows, repeat transformations for all of them, and to generate a large number of attributes that enter the knowledge discovery process. In that setting we expect that the results of the KD will demonstrate usefulness of some of the windows, and suggest their use in future experiments.

7.4.1. Events in time sequences

Special attention in time series analysis should be devoted to the "events". As mentioned, both long and short sequences can have interesting events included. The list of potentially relevant events for every sequence can be very long. Currently there is no automatic approach that can enable enumeration of all relevant events. The only practical solution is a short list of expert suggested events that are potentially interesting for the analysis in the concrete domain.

The events are significant for the description of the sequences. For example, it can be detected that NYHA class 3 patients have much more short periods of high heart frequency than NYHA 2 patients. In that example "short period of high heart frequency" is a relevant event. But events are even more significant for the definition of the properties of interest (i.e. patients changing from NYHA 2 to NYHA 3 or introduction of ARB medication). When events are used as property of interest for the knowledge discovery task, then we are actually interested not in the whole patient record but only its part that is either immediately before of the event (i.e. in the case of changing NYHA class), immediately after the event or we are interested in differences between the two states (i.e. in the case of changes in therapy). The consequence is that we do not use complete patient data as input but only data before the event and/or after it. It means that we can use the same patient in a few different examples (Figure 7-1). The possible combinations are:

a) the same patient has more events and each of them is one positive case

b) the patient has no events and different time intervals are used as different control cases

c) the same patient is used both as positive case(s) when the event occurs and as control case(s) in period of time rather far from the events.

The positive consequence of this fact is that we can have more examples than patients. But it also means rather complicated task for example extraction from the data warehouse.

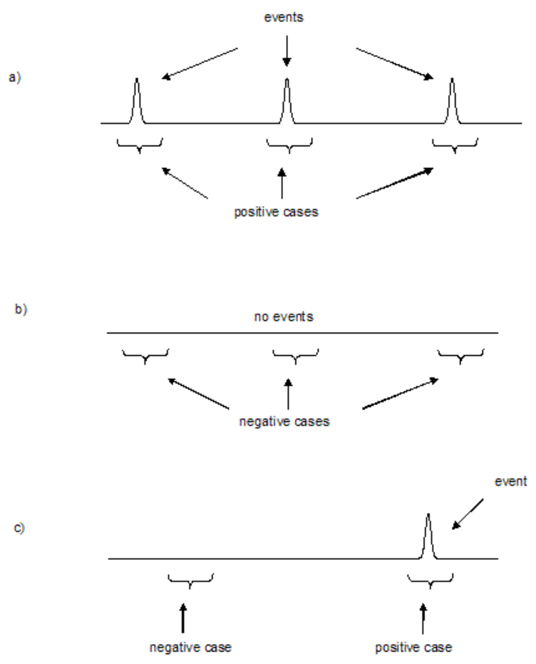


Figure 7-1. When events are used as a target property of interest for knowledge discovery, then each patient can produce a few different cases

It is important to notice that "events" can be also simple but relevant patient characteristics (i.e. current patient status is NYHA 2). In this way event driven transformation is the most often used approach for HF platform data. It's implementation is described in detail in Section 8.3.

7.5. Feature construction

The addition of relevant and potentially useful attributes into the original data file format is the process that in the machine learning field is known as *feature construction*. This term includes also construction of complex and relevant logical expressions inside the induction algorithms that is not applicable in the data preparation phase. When performing analysis of complex data one of the major problems stems from the number of variables involved (Figure 7-2). Analysis with a large number of variables generally requires a large amount of memory and computation power and a classification algorithm overfits the training sample and thus generalizes poorly to new instances. Feature extraction is a general term for methods of constructing combinations of the variables to get around these problems while still describing the data with sufficient accuracy. The feature extraction has been traditionally viewed as a process to use different weighting schemes to combine, linearly as in PCA and LDA or non-linearly as in neural networks, the features to produce a reduced number of new features, ideally uncorrelated.

Effective feature construction can significantly increase expressiveness of induced models and, because of that it is very relevant for the whole knowledge discovery process. In general feature construction is a process that is hard to systemize. It means there is no approach that can systematically detect all potentially relevant combination, especially when attributes have numerical values. In practice, the most efficient approach is a combination of some machine learning algorithm, like *rule induction algorithm*, that generates potentially relevant hypotheses, and human knowledge that transforms the hypotheses into attribute form with some realistic meaning.

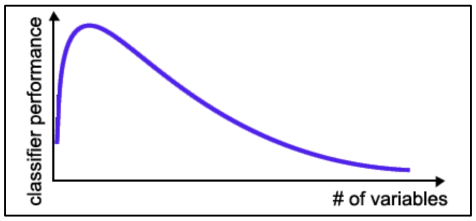


Figure 7-2. Classification performance behaviour depending by the numbers of variables

Best results are achieved when an expert constructs a set of application-dependent features. Nevertheless, if no such expert knowledge is available general dimensionality reduction techniques may help. These include:

- Principal components analysis
- Semidefinite embedding
- Multifactor dimensionality reduction
- Nonlinear dimensionality reduction
- Isomap
- Kernel PCA
- Latent semantic analysis
- Partial least squares

7.6. Feature selection

The feature selection refers to the selection of an optimum subset of features, derived from the input variables, to reduce the complexity of the knowledge discovery task [5,6]. The complexity of the knowledge discovery task is determined by the number of examples (typically rows in the data file) and the number of attributes (columns in the data file). For good knowledge discovery results the number of examples should be large[1] and the expected quality and especially the reliability of the induced relations grow with the number of examples. Expressiveness of the induced models grows with the number of attributes. Because of that, the suggestion is to include as much as possible of different attributes that describe the examples. The final result is that the data file can be rather large. As induction algorithms should analyze also logical combinations among attributes, the complexity of the knowledge discovery algorithms can be very large. Every machine learning algorithms includes mechanisms that prevent the complexity explosion. Nevertheless, it is regarded as a good practice to detect and eliminate attributes that are practically useless for the knowledge discovery process during the data preparation phase. There are two good reasons for that, especially in the situations when feature construction can add significant number of additional attributes. The first is to reduce time and space complexity for the execution of the knowledge discovery algorithms. The second and more relevant is that for some algorithms the quality and reliability of induced models can be reduced in the presence of irrelevant attributes. The process of the selection of the set of relevant attributes that will actually enter the inductive learning process is known as *feature selection.*

The reason that the term feature selection is used instead of attribute selection is that the term feature includes input attributes but also all, possibly very complex and possibly very large number of logical combinations of input attributes. Feature selection has the goal to select the most relevant subset in this huge space. At the basic level of data preprocessing, features can be only represented as attributes and feature selection actually means detection of a relevant subset of input attributes. More complex feature selection is typically part of machine learning algorithms.

Thus, attribute selection is distinct from feature selection if the predictor (classifier, regression machine, etc.) operates in a feature space that is not the input space. The use interchangeable of the terms variable and feature it is often observed.

From the theoretical point of view, the model selection problem of feature selection is notoriously hard and even harder is the simultaneous selection of the features and the learning machine.

[1] In medical application the number of examples is typically in the range of 100 to 500 and this is realistic also for the HF platform. Experiments can be done also with less than 100 examples but then complexity of the solutions is limited and reliability of the results is low.

The standard approach for the detection of relevant features is by measuring correlation with the target classes. More complex methods are based on application of some machine learning algorithms (i.e., Random Forest [7]) that after identification of most useful attributes for the distinction of the classes in the target attribute, declares these attributes as most relevant. Logically based relevance uses the concepts of absolute and relative irrelevance of simple logical expressions (i.e., (attribute A >0.5)) and declares as relevant all attributes that can be the base for at least one relevant feature [8].

Feature selection schemes are often divided in two categories:

1. Filter methods, which use only the general characteristics of the training data without reference to the learning algorithmfinally used
2. Wrapper methods, that make use of the performance of the chosen classifier to select features.

Within the pattern recognition literature the most popular schemes for feature selection are based on classification complexity estimation and validation set classification. In [9] taxonomy of feature selection algorithms has been provided.

Feature selection also helps to acquire better understanding about the data by telling which are the important features and how they are related with each other. By removing most irrelevant and redundant features from the data, feature selection helps to improve the performance of learning models by:

- alleviating the effect of the curse of dimensionality;
- enhancing generalization capability;
- speeding up learning process;
- improving model interpretability.

7.7. Data cleansing

Data cleansing is very relevant data preprocessing step. Although every machine learning algorithm can tolerate imperfect input data, explicit elimination of errors and outliers even before the start of these algorithms can improve the quality of the final result. If the problem can not be corrected then the problematic value is substituted by an unknown attribute value. If the problem is detected in the target class attribute then setting it to an unknown attribute (class) value practically eliminates the complete example form the induction process.

It is a good practice to discuss every detected data problem with the domain experts before its correction or elimination. The discussion can also help in improving the quality of the future data collection process. Detection of problematic values can be done by human inspection or by some software approach.

Expert outlier detection is aimed at identification of those attribute values that are either very different from other values from the same attribute (exceptions) or that are known as unrealistic due to the available medical knowledge. Especially interesting is detection of logical inconsistencies among different attribute values for the same example. Human outlier detection is applicable only for relative small data files and it is its main limitation. Automatic tools can not completely substitute human work because of the involved expert domain knowledge, but they can effectively help in detecting exceptions at the level of a single attribute.

Much more practically relevant automatic tools for data cleansing are based on iterative application of machine learning procedures. The idea is to apply some noise tolerant machine learning algorithm and analyze the classification behaviour of the resulting model. For those examples that are not correctly classified by the resulting model it can be assumed that they contain some noise in relevant attributes, either in the class attribute or in some input attribute that is very relevant for the resulting model. The characteristic of the approach is that it can not identify the concrete problematic attribute value but only identify the complete example as potential outlier. It means that if we accept the suggestion we can only eliminate the complete example and data cleansing is performed by example elimination. Nevertheless, the approach is very attractive because it can ensure data consistency that is very relevant for successful knowledge discovery without any domain knowledge.

For the automatic data cleansing tasks any machine learning algorithm with good classification and noise tolerance characteristics can be used. Because of its systematic and iterative approach, application of the ILLM noise detection algorithm is suggested for the HF platform [10].

7.8. Handling unknown attribute values

All real world applications of inductive learning systems are confronted with the problem of unknown attribute values in the training set. The simplest approach is to ignore examples with unknown attribute values. But this approach can be applied only when the number of unknown values is small and the number of available examples is very large. A commonly used approach is to replace missing values with a default value in the data preparation phase. For example, CN2 [11] replaces unknown values of discrete attributes by most commonly occurring value (the mode value), and unknown values for continuous attributes by the average value (the mean value). It is also possible to substitute the unknown values by random values or to try to estimate their values from known values of other attributes in the same example. These and similar techniques have drawbacks, especially when the number of unknown attribute values is high. Extensive experiments presented in [12] demonstrate that none of the mentioned approaches is absolutely superior compared to the others.

An alternative approach suggested in [13] is to reduce the apparent gain from testing attribute A by the proportion of cases with unknown values of A. The rationale behind this approach is that testing attribute A will yield no information when it has unknown values, and when A has unknown values for all examples it is completely useless. The advantage of the approach is that it is theoretically sound. The problem is that its implementation is not a simple task. In decision tree induction we can easily appropriately reduce the information gain for the node testing attribute A that includes unknown values, but we do not have an effective method to partition examples with unknown values based on such a node. The consequence is that we can not expect optimal performance in nodes below the one that is based on an attribute that has many unknown values. In decision tree induction experiments reported in [10] the approach had similar prediction quality as those based on substituting unknown values by some known value in data preprocessing.

The approach based on reducing the gain from testing attributes with unknown values seems much more appropriate for covering rule induction approaches. The reason is that we do not have to partition the training set as in decision tree learning, but only to appropriately redefine the used heuristic evaluation measure so that the resulting value will get reduced by the proportion of unknown values. Although the principle is simple, there is a practical problem that, in contrast to decision tree induction, rules typically represent simultaneous decisions based on more than one attribute value and this makes the necessary computations potentially complex.

7.9. Bibliography

[1] Guyon, I., Elisseeff, A. (2003) *An Introduction to Variable and Feature Selection*. Journal of Machine Learning Research 3: 1157-1182.

[2] Hall, M.A., Holmes, G. (2003) *Benchmarking Attribute Selection Techniques for Discrete Class Data Mining*. IEEE Transactions on Knowledge and Data Engineering 15(3).

[3] Shahar, Y. (1997) *A framework for knowledge-based temporal abstraction*. Artificial Intelligence 90:79-133.

[4] Belazzi, R.,Larizza,C., Magni, P., Belazzi, R. (2005) *Temporal data mining for the quality assessment of hemodialysis services*. AI in Medicine 34:25-39.

[5] Dash, M., Liu, H. (1997) *Feature selection for classification*. Intelligent Data Analysis 1(3).

[6] Kira, K., Rendell, L.A. (1992) *A practical approach to feature selection*. Proceedings of the International Conference on Machine Learning, D. Sleeman and P. Edwards, editors, pp. 249-256.

[7] Breiman, L. (2001) *Random forests.* Machine Learning Journal, 45:5–32.

[8] Lavrac, N., Gamberger, D., Turney, P. (1998) *A relevancy filter for constructive induction.* IEEE Intelligent Systems & Their Applications, 13:50-56.

[9] Jain, A.K., Zongker, D. (1997) *Feature-selection: evaluation, application, and small sample performance.* IEEE Trans. Pattern Anal. Machine Intell. 19 (152).

[10] Gamberger, D., Lavrac, N., and Dzeroski, S. (2000) *Noise detection and elimination in data preprocessing: experiments in medical domains.* Applied Artificial Intelligence, 14(2):205-223.

[11] Clark, P., Niblett, T, (1989) *The CN2 induction algorithm*. Machine Learning, 3(4):261-283.

[12] Bruha, I., Franek F. (1996) *Comparison of various routines for unknown attribute value processing: The covering paradigm.* International Journal of Pattern Recognition and Artificial Intelligence, 10(8): 939-955.

[13] Quinlan, J.R. (1989) *Unknown Attribute Values in Induction.* Proc. of the Sixth International Workshop on Machine Learning, ML-1989, pp. 164-168.

8. Functional specification of data preparation for knowledge discovery

Data preparation begins with definition of the goals of the knowledge discovery (KD) process. These goals are implemented by different tasks and for each of them the data preparation process constructs the appropriate data table. The following steps are physical data extraction from the data warehouse and data loading into tables prepared for the KD process. In the HF platform, special attention is devoted to the problem of the transformation of data available in a form of time sequences. The presentation in this section ends with a description of the steps implemented for the preparation of the available HF related retrospective data sets.

8.1. Data and domain understanding

Regarding D5, there have been identified the following relevant knowledge discovery goals:

I. Find the single biomedical parameter or the combination of parameters (among signs, symptoms, and test results) that reliably define different degrees of HF severity and that may increase the diagnostic capability for HF. It should be noted that NYHA classification takes into account only the symptoms of heart failure (dyspnoea, fatigue, arrhythmia), and no signs or test results. In this way it is only a subjective classification. The degree of heart failure severity should be defined not only by regarding NYHA class, but also by signs, selected tests (in particular echocardiography) and laboratory tests results.

II. Find the single biomedical parameter or the combination of parameters (among signs, symptoms, and test results) that identify the best that the subject is in stable, improving, rapidly worsening, or slowly worsening condition. Find predicting parameters and their possible cut-off and reference values or ranges for each of these main diagnostic classes of HF, thus suggesting new possible HF severity classification.

III. Test if the quality of results for both I) and II) can increase in the case of repeated home telemonitoring of relevant symptoms and parameters like blood pressure, heart rate, respiratory frequency, percentage of body water, weight, and body temperature.

IV. Among continuously monitoring data, identify the potential precipitating and exacerbating factors of decompensated CHF.

V. Evaluate consequences of an applied therapy.

The list is not complete, but according to D5 it presents a set of currently challenging tasks. Tasks related to continuously monitoring data which in general cannot be collected without the HF platform, seem to be especially interesting.

The process of data understanding determines the tasks for the data warehousing. Data warehousing must take extra care of the way in which target attributes are prepared and what is the definition of the examples. They should enable that the results of the knowledge discovery are such that they represent answers to the recognized goals I-V.

Goal I

The goal I consists of two sub-goals. The first is to test if NYHA class can be reliably predicted by single biomedical parameter or the combination of parameters. The second is to test if there is a better way to describe patient status, because NYHA classification is not an objective classification. Both of these goals can be achieved by analysing patient data collected during their visits to specialized institutions or hospitals. Every complete data record is an example for this knowledge discovery task. It means that the same patient can provide more examples, depending on how many times his complete medical status has been collected.

For the first sub-goal the target attribute is the NYHA class at this visit. There are four different NYHA classes, and because of that the problem is transformed to a series of two-class problems. Although this can be done in a few different ways, the suggested approach is to make three two class problems: class I versus classes II-IV, classes I and II versus classes III and IV, and classes I-III versus class IV. These three data sets enable analysis of relevant decision points among NYHA classes.

For the second sub-goal, the suggested approach is application of unsupervised clustering. In this setting the information about the NYHA class is treated as a normal input attribute. In order to be able to use standard classification knowledge discovery algorithms for clustering purposes, the binary classification task is formed in a way that all available (real) examples are positive examples and an additional set of negative examples is obtained by random sampling (shuffling) of real attribute values. In this way we have a target attribute that is positive for real examples and negative for artificially generated examples. The relevant subpopulations detected for the positive class can be interpreted as relevant patient clusters. After recognizing them, their usefulness should be tested for prognostic purposes in the same way as for NYHA classes.

Goal II

Besides information necessary for goal I, we need some information about patient status after the data collection stage for goal II. Ideally, it can be NYHA class determined by medical personnel at the time of the next visit to the specialized institution, or it can be information about hospitalization that followed after the data collection stage. If the hospitalization occurred soon, we can classify the case as rapidly worsening; if we have increased NYHA class but without hospitalization we can classify it as slowly worsening, while stable conditions are when there are no hospitalizations and no changes in the NYHA class. As in goal I, in goal II we can have more examples generated from the data of the same patient, but in contrast to goal I we also need some information collected at the following visit or hospitalization. Therefore, the number of examples for goal II is smaller than for goal I.

Goal III

This goal is an extension of goals I and II with information about repeated measurement of parameters like blood pressure, heart rate, respiratory frequency, percentage of body water, weight, and body temperature. The target attributes are constructed in the same way as in goals I and II, but input attributes should include information about at least one continuously measured parameter. It is not necessary that all of the parameters are monitored.

For the success of the knowledge discovery process, decisive is transformation of the data sequences to the attribute form that can be used by knowledge discovery algorithms. The transformation is described in detail in Section 8.2.2.

Goal IV

Decompensation is a major problem of HF management. Its early detection can be done only in the case of repeated home telemonitoring of relevant parameters which, in combination with previously collected patient data, can trigger the alarm. It means that availability of parameters which are measured repeatedly is the necessary condition both for decompensation detection and for the knowledge discovery process that needs to detect relevant cut-off points. As in goal III, it is not necessary for all parameters to be monitored.

The main problem of the goal IV is a definition of positive and negative examples. Ideally, the patients that have been admitted to the hospital because of the decompensation represent positive examples. More specifically, the data collected *before* the hospitalization represents positive cases. The only problem is that there will be a small number of such examples, i.e. patients that have been continuously monitored for some time and then the documented decompensation has occurred. It is much easier to collect negative cases (continuously monitored patients without decompensation), but the problem is ensuring that the decompensation *did not* in fact occur in some relevant period after the last data that enters the knowledge discovery process. Such negative proof is always problematic, and because of that the only solution is to be aware that negative examples can be false negative and to assume that such cases will be rare. In order to allow to knowledge discovery algorithms to compensate false negatives, it is suggested that number of collected negative examples is large, i.e. five or ten times larger that the number of positive examples.

An alternative approach is using significant changes in some continuously monitored parameters as markers for positive cases. For examples, the increase of 15% in the heart rate (always measured in the resting conditions) in respect to the mean value measured in the previous period that is at least one month long, may be an indication of the worsening conditions. Such patient can be accepted as the positive case and the data collected in the period *before* the detected increase is acceptable as a positive example if in the period before the onset there were no similar situations (all fluctuations have been below 10%). As negative examples, acceptable are the periods with low data fluctuations (below 10%) that are *followed* by at least equally long period (1 month) of low fluctuations. The knowledge discovery task is prediction of significant changes. The advantage of the approach is that there can be significantly more positive and negative cases. The problem is that we do not have the guarantee that significant changes actually represent decompensation. Additionally, the unreliability of the repeated measurements may be the problem. While this type of noise can be acceptable for input attributes, in the described setting the noise influences the target attribute. In a drastic situation, the result can be that a large number of false positive cases completely disables effective knowledge discovery.

Goal V

The intention is to test effectiveness of the applied therapy. The differences between patients that are receiving some therapy and the ones that are not receiving it, as well as consequences of starting or stopping some treatment, can be tested.

8.2. Specification of the data transformation process

The practical realization of the tasks presented in the previous section is realized in three levels. At the highest level there are templates for the data table construction which define the type of the information included in the resulting data tables. There are two very general templates. The idea is to implement these complex data models and to enable realization of the large variety of resulting data table types by simple omission of some template parts. At the medium level there are procedures for data transformation while on the lowest level there is user selection of actually included attributes.

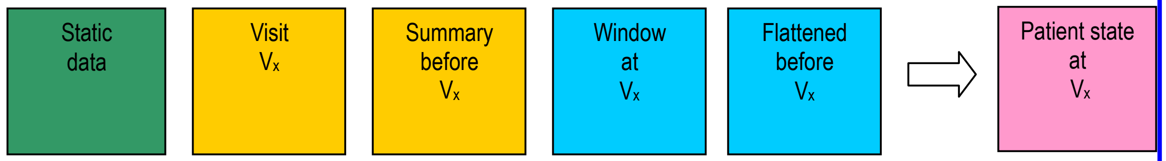
8.2.1. Data table templates

Figures 8.1-8.2 define two basic templates for data table building. They implement the event driven approach for data preparation described in Section 7.3 and are sufficient for the realization of all tasks specified in Section 8.1.

The first template is applicable for all tasks in which we are interested for understanding and predicting patient status or prognosis. It includes only the patient data before the specified event. The second template is specially intended for the analysis of changes of the therapy and it includes data both before and after the event.

Both templates are general and can be reduced in many different ways based on the availability of data (continuous monitoring present or not) and on the exploration goals (i.e. reduced set of attributes, exclusion of information about previous follow-up visits, exclusion of information before the event).

Template 1:



Purpose:

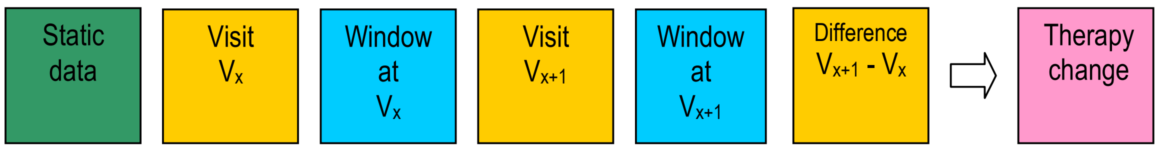
- Short and long term prognosis and patient status evaluation based on data from the follow-up period and continuous monitoring (yellow – data form follow-up visits, blue – continuously monitored data).

Goals:

- Goals I-IV from Section 8.1

Figure 8-1. Dataset template for KD tasks type 1.

Template 2:



Purpose:

- Evaluation of medication (treatment) changes based on data from the follow-up period and continuous monitoring.

Goals:

- Goal V from Section 8.1

Figure 8-2. Dataset template for KD tasks type 2.

Legend

🟩 Static data	*Static patient data* The block includes data like demographic data, cardiovascular history, personal and family anamnesis.
🟨 Visit Vx	*Baseline, additional visit, of final visit data* The block includes physical examination data, laboratory assessment, echocardiography data, etc. at visit V_x (the last available visit before the event).
🟨 Summary before Vx	*Summary of visits Baseline->V_x* Flattened visit data for the complete period and differences between V_x and baseline evaluation.
🟦 Window at Vx	*Continuously monitored data* Flattened continuously monitored data in the pre-specified window corresponding to the visit V_x.
🟦 Flattened Vx	*Flattened continuously monitored data* Flattened data for the complete period Baseline-V_x, flattened window at baseline, and differences between baseline, V_x window, and the complete period.
🟪 Patient state Vx	*Property of interest* Patient state at V_x, or patient state at V_{x+1} , or difference in states V_{x+1} - V_x, or patient outcome anytime after V_x.
🟪 Therapy change Vx	*Property of interest* Changes of therapy at additional visit V_x.

Properties of interest for goals specified in Section 8.1 are summarized in Table 8-1. The table includes a short list for each goal representing the starting research point. The specified properties of interest correspond to data tables parts "Patient state V_x" and "Therapy change V_x" defined in data table templates.

Table 8-1. Specification of properties of interests for different KD goals.

GOALS	Property of interest
goal I and III	- NYHA 1 in V_x - NYHA 1 or 2 in V_x - NYHA 3 or 4 in V_x - NYHA 4 in V_x - unsupervised detection of relevant subpopulations
goal II and III	- increased NYHA class between visits V_x and V_{x+1} - decreased NYHA class between visits V_x and V_{x+1} - required hospitalization between visits V_x and V_{x+1} - fatigue change + or dyspnea change + at additional visit V_x
goal IV	- sudden changes in continuously monitored heart rate (to be defined) - significant increase in continuously monitored % of body water (to be defined) - significant increase in continuously monitored respiratory rate (to be defined)
goal V	- introduced beta blocker - introduced diuretics - discontinued diuretics

Each of the specified KD tasks are performed for the complete available patient domain. But also, interesting subtasks are if the domain is a subpopulation of:

- male patients
- female patients
- NYHA 1 or NYHA 2 patients
- NYHA 3 or NYHA 4 patients
- collection center MIL (or KRK, or CAT)
- current therapy (i.e. antiarrhytmics or anticoagulants)

8.2.2. Flattening of long numerical sequences

Flattening is the process of transformation of sequence data into a fixed set of attributes. For long numerical sequences, it is always done for some fixed time interval dT called window. The process can be repeated for different windows and for the complete sequence. Finally, results for different windows, including also differences between corresponding computed values, can be integrated into one row of the resulting KD table.

In HFP, we use following nine aggregated attributes for numerical data:

- number of data points
- arithmetic mean
- median
- minimum
- maximum
- maximum - minimum
- interquartile distance
- standard deviation
- trend (slope of linear regression)

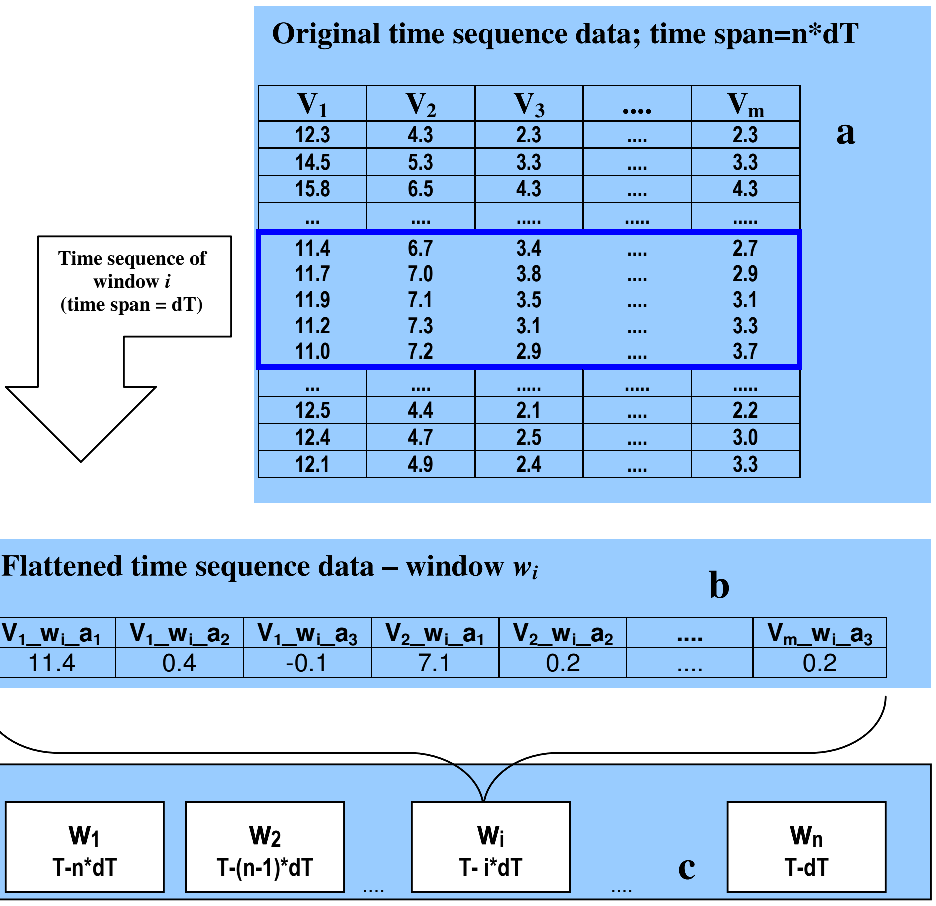
These aggregated attributes are produced for each window, including the complete sequence.

Flattening transform is illustrated in Figure 8-3 for the set of variables Vj, j=1,…,m; set of time windows wi , i=1,..,n (each spanning pre-defined time interval dT); and set of aggregates ak,k=1,2,3 (e.g. a1=mean, a2=standard deviation, a3=trend).

When the building block "*Flattened before V_x*" (defined in Section 8.2.1) is used in the KD table templates, it means that we are doing:

- flattening for the complete available sequence between baseline evaluation and visit V_x (9 attributes)
- flattening for the first window in the sequence (9 attributes)
- flattening for the last window in the sequence (9 attributes)
- compute difference between the last and the first window for all aggregated attributes (number of points, mean, … , trend) (9 attributes)
- compute difference between the whole sequence and the first window for all aggregated attributes (9 attributes)
- compute difference between the whole sequence and the last window for all aggregated attributes (9 attributes)

In this way we get in total 6 times 9 aggregated attributes from one long numerical sequence. Exceptionally, aggregated data about window V_x (9 attributes) can be omitted if they are already included by the box "Window at V_x".


Original time sequence data; time span=n*dT
a
Time sequence of window *i* (time span = dT)
Flattened time sequence data – window w_i
b
c
W1 T-n*dT
W2 T-(n-1)*dT
....
Wi T- i*dT
....
Wn T-dT

V_1	V_2	V_3	....	V_m
12.3	4.3	2.3	....	2.3
14.5	5.3	3.3	....	3.3
15.8	6.5	4.3	....	4.3
...	....	.....	.....	.....
11.4	6.7	3.4	....	2.7
11.7	7.0	3.8	....	2.9
11.9	7.1	3.5	....	3.1
11.2	7.3	3.1	....	3.3
11.0	7.2	2.9	....	3.7
...	....	.....	.....	.....
12.5	4.4	2.1	....	2.2
12.4	4.7	2.5	....	3.0
12.1	4.9	2.4	....	3.3

$V_1\_w_i\_a_1$	$V_1\_w_i\_a_2$	$V_1\_w_i\_a_3$	$V_2\_w_i\_a_1$	$V_2\_w_i\_a_2$	....	$V_m\_w_i\_a_3$
11.4	0.4	-0.1	7.1	0.2	....	0.2

Figure 8-3. Transformation of the time series numerical data into the flattened time series representation suitable for knowledge discovery. Process starts with a query for the time series data over a certain period of time (a); (b) one window of flattened time series data represents a part of one row of data in the final KD table (c).

Some of the aggregated attributes can be obtained using standard OLAP functionalities (roll-up and roll-down, slicing, pivoting). However, there are transforms that are not available within standard OLAP functionalities. These include, for example, median and slope computation as well as annotation of special events. They are computed by a stand-alone software application (KDD-ETL) described in Section 8.3.

8.2.3. Flattening of short data sequences

Short sequences can be numerical or categorical.

Short numerical sequences are flattened into 9 aggregated attributes in a way identical to long sequences. The difference is that for short sequences flattening is performed on the whole sequence only and not for windows at the sequence beginning and end. It means that for building block "*Summary before V_x*" for every short numerical sequence we do:

- flattening for the complete available sequence from the baseline evaluation to the visit V_x (9 aggregated attributes)
- compute the difference between V_x value and baseline evaluation value (1 attribute)
- compute the difference between mean for the sequence and V_x value (1 attribute)
- compute the difference between mean for the sequence and base line value (1 attribute)

In this way flattening results in 12 aggregated attributes for every short numerical sequence.

For short categorical values, the following aggregated attributes are generated:

- number of data points
- mode (most frequent value)
- percentage of mode (percentage of specified values that are equal mode)
- number of points in which the value changes

When building block "*Summary before V_x*" has to be prepared for every short categorical sequence (including also Cb descriptors with only two categories) we do:

- flattening for the complete available sequence from the baseline evaluation to the visit V_x (4 aggregated attributes)
- compute the difference between V_x and baseline evaluation value (set 1 if equal otherwise 0) (1 attribute)
- compute the difference between V_x and mode for the sequence (set 1 if equal otherwise 0) (1 attribute)
- compute the difference between baseline evaluation value and mode for the sequence (set 1 if equal otherwise 0) (1 attribute)

In this way we get 7 aggregated attributes for every categorical sequence.

8.2.4. Other relevant transformations

Some patient data requires special attention during the transformation to the KD tables. In Appendix A is the complete list of descriptors with comments about their transformation. For each attribute the type is specified: N - numeric, C - categorical, Cb - binary categorical. The type determines the procedures that are used for flattening of data sequences.

Table 8-2. Functionalities of the KD-ETL component

Functionality	Description
User interface	KD-ETL has a WEB-browser based user interface; the user interface will have simple structure reflecting main KD-ETL functionalities: input specification, data selection and data transform options, output and logging options.
Input	
Query from HFP-DW	KD-ETL has predefined queries for typical KD tasks; using metadata, user will also have the possibility to create own queries
Output of OLAP tool	KD-ETL has the capability of loading data files produced by OLAP tool Mondrian
Output datasets of KD-ETL	KD-ETL is able to reload already prepared files
Data transformation	
Flattening time sequence data	For the time sequence types of data, KD-ETL performs flattening transforms with some user options (i.e. window size and total period)
Define attribute set	User can select attributes that will be transformed and actually used in the output table.
Logging	
Acquired data log	Detail log of data retrieved form HFP-DW, data marts, OLAP or provenance data if data are loaded from KD-ETL files produced earlier
User choices in KD-ETL	All user actions during the KD-ETL session
Data transforms	Attributes and transforms produced on the data during the KD-ETL session
Output	
Formatting (selectable)	Output formats: arff, csv, tab-delimited data
Saving locally	Data is saved on the HFP-DW server
Making file(s) available to the user	Data is made available to the user who generated data via WEB linked file.

8.3. Data preparation for KD process: KD-ETL

As presented in Figure 6.1, data stored in the HFP-DW has to be transformed into the data tables appropriate for different KD problems. These transformations are provided via two generic paths: (i) direct query and transform of data from the data warehouse; (ii) by designing data cubes and using OLAP tools for data exploration, forming and then exporting basic data tables for further transformations. In both scenarios decisive is the role of the software tool called KD-ETL (Extraction, Transformation, and Loading for KD) which will be implemented within the HEARTFAID data warehouse environment with the main function to produce input data tables for different knowledge discovery tasks. Table 8-2 gives detailed specification of KD-ETL functions.

8.4. Retrospective data preparation

Data preparation process for retrospective data is illustrated with two very different domains. It is assumed that at least one additional retrospective domain with manually collected data simulating continuous patient monitoring will be analyzed.

The first domain presented in this section is the dataset collected at UNICZ (University of Catanzaro) for the investigation of *correlations among gene polymorphisms and cardiographic parameters*. This is a small dataset with the data collected in the form of four tables. In each of them there is patient data specifying measured gene polymorphisms as well as clinical, echocardiographic, follow-up, and blood serum analysis data. The second retrospective dataset is coming from the Italian Association of Cardiology (ANMCO). It is a large dataset representing *prognosis evaluation* for more than 18000 HF patients collected in Italian hospitals in the period of 1995.-2005. The dataset describes patients at the first visit and specify patient outcomes in the period of a few years of follow up.

8.4.1. Gene polymorphisms dataset

Dataset contains information about HF patients describing their gene polymorphism data combined with clinical information (anamnestic data, blood pressure data, and some ECG measurements), echocardiographic data (like left ventricular diameter, interventricular septum, and shortening fraction), and blood serum analysis (like cholesterol, fibrinogen, and blood cell counts). The data is distributed in four tables depending on measured polymorphisms. Tables contain between 71 and 176 patients. The same patient can be in more than one table.

Data warehousing

The fact that some patients occur in more then one table suggested data integration as the main data warehousing step. By integrating the data we create possibilities for investigating relations among different genes and simultaneous analysis of all gene polymorphisms versus echocardiographic data.

Data integration has been done through a simple database transformation. The advantage of the approach is that during this transformation it was possible to check data consistency in different tables, and an error has been successfully detected and corrected. Additionally, in a few cases we were able to eliminate unknown values when we had known values in other tables.

In this dataset we have no time sequences and therefore data warehousing steps end with the export from the data table to the text format. Finally, only the heading necessary for knowledge discovery algorithms had to be added.

Data and domain understanding

The available data includes information about following genes: IRS-1, IRS-2, PPR-gamma, KIR, UCP-2, TRB3, IL10-1092, IL10-592, ACE (Angiotensin-Converting-Enzyme), ADD (alpha-adducin), and ALD (aldosterone-synthase). For some of them, rather small number of measurements is available (i.e. for IRS-2 only 30).

The most relevant task pointed out by medical doctors is the detection of relations between polymorphism data and echocardiographic data. The later include 13 different measurements. It must be noted that among these data, the information about body surface has been added, although formally it is not a part of the echocardiographic information.

The dataset includes information about blood serum analysis, so it can also be used for the detection of correlations between gene information and blood data, as well as between echocardiographic information and blood data.

It can be noted that besides very restricted information about some gene polymorphisms, also some echocardiographic and blood data contains a lot of unknown values as well. This must be taken into account both during data pre-processing and during the selection of potential target functions.

Attribute selection

In this domain the task is detection of correlations between two sets of attributes. For such task no knowledge discovery tool is directly applicable. The transformation into a form that can be solved is possible only by appropriate target attribute selection. Actually, the task is transformed into a series of classification problems. In the first part we select each gene as a target attribute (resulting in 11 classification tasks) and use echocardiographic data as input attributes used for building classifiers. When such information is available, the polymorphism that is known as correlated with HF is selected as a target class. In the second part we select each echocardiographic attribute as a target attributes (resulting in 16 classification tasks) and use gene information as the input attributes. In this case, selection of the target class is not so simple because echocardiographic attributes are continuous. We have used median to separate them into two regions and typically the higher is then used as the definition of a target class. Clinical and blood serum analysis data have been eliminated from this part of the analysis in order to restrict the search to only the correlations between gene and echocardiographic data.

Feature selection and construction

Feature selection is not relevant because this dataset contains a small number of attributes. In contrast to that, feature construction seems potentially very relevant. This is also noticed by medical doctors independently. They have already included some artificially constructed features into the original dataset. The two of them are differences among attribute values (relative wall thickness-calculated for interventricular septum and relative wall thickness-calculated for posterior wall), two are left ventricular mass index and its normalization by the body surface, one is E/A ratio, and two are obtained by exponentiation of diastolic interventricular septum and diastolic posterior wall, respectively. The former are potentially very useful in contrast to the last two obtained by exponentiation that are not useful for most knowledge discovery algorithms as stand-alone attributes.

Preliminary data analysis results demonstrate usefulness and relevance of the body surface as the normalization factor. This is in accordance with medical expectations and justifies its inclusion in the dataset. Introduction of normalization by body surface seems reasonable for some other echocardiographic values as well. During data preparation there have been constructed additional attributes presenting values of echocardiographic data divided by body surface for 17 attributes in total. During the knowledge discovery process, if the relevance of any of these additional attributes would be demonstrated, there is a possibility to additionally introduce more complex relations with respect to the body surface or between the original attributes.

Data cleansing

Originally, unknown values in data tables have been left as empty fields or some clinically unrealistic value (like zero for body surface and end systolic volume) has been used for that purpose. Unknown values have been systematically substituted by '?'.

For some attributes, there have been noticed an unexpected values. The values have been substituted by unknown values. Examples are (Table 8-3):

Table 8-3. Examples of unexpected attribute values.

patient ID	attribute	value
1240	A	5.6
487	aorta	6.63
906	systolic posterior wall	3.18

It has also been detected that patient with ID 197 has a few very uncommon values. It is assumed that in this case the patient is an exception and he has been eliminated as a complete instance.

Noise detection process has to be performed separately for each classification task and this will be performed as a part of the knowledge discovery process.

8.4.2. Prognosis evaluation dataset

The prognosis evaluation dataset has been prepared for the HEARDFAID project by ANMCO (Associazione Nazinale Medici Cardiologi Ospedalieri) as requested by prof. Perticone and dr. Sciaqua from University of Catanzaro. It represents the information extracted from a huge database on HF patients collected by Italian Network of hospitals in the period of 1995.-2005. The dataset includes data about more than 18000 patients. Each patient is described by 45 attributes collected at the first visit (like blood pressure, ejection fraction, and cardiothoracic ratio) and events (outcomes) related to the patient in the period of a few years of follow up. For each event there is a date that enables, in the combination with the date of the first visit, calculation of the time difference from the first visit to the event. The included events are: patient’s death, hospitalization for HF causes, any hospitalization, HF worsening, arrhythmic event, ischemic event, stroke, syncope, embolus, cardiac surgery, and variation of the NYHA class.

Data warehousing

The dataset requested no special data warehousing procedures. Moreover, in order to start with data preparation steps, no format changes have been necessary.

Data and domain understanding

ANMCO dataset registers the initial state of patients at the moment of enrolment in ANMCO observational study and their known outcomes. For decision support services, the dataset could be useful in order to define predictive rules and models for HF prognosis. The limitation is that for each patient only one visit is registered; thus it is not possible to extract prognostic models based on multiple visits information.

In order to construct useful prognostic models, different data mining tasks can be defined. Those tasks can be based on a single event or on a combination of events (global approach). Considering each outcome separately, it is possible to build a series of dataset in which the class attribute reports “yes” or “not”, depending whether the specified event (outcome) happened or not. For example, we have a dataset for the outcome “DEATH”, in which, besides the input data describing patient status, there is a class attribute reporting if the patient is dead or not. The time interval between the first visit and the event can vary from a few days to several years depending from the specific patient. This information is crucial for prognostic purposes.

In order to be able to treat the prognosis as a classification task, it is necessary to define one or more fixed time periods. For HF it seems reasonable to select the following time periods: three months, six months, and one year. In order to detect the danger of an approaching arrhythmic event the following question is relevant: "What are the properties of patients that had an arrhythmic event during three months after the first visit?" For this task, positive cases are patients with arrhythmic events that happened in less than three months after the first visit. Negative cases are not all other patients, but only those that had no arrhythmic events and the follow-up period was longer than three months. Long term danger of an arrhythmic event can be recognized by a similar classification problem in which time period of one year is used instead of three months. Practically it means that for each of 11 different events we can form a knowledge discovery task for three different periods, resulting in 33 different knowledge discovery problems. It can be expected that only a few of them will result with interesting and relevant results. However, it is not possible in advance to define a smaller subset of tasks. Additionally, based on medical knowledge, it can be assumed that the results will significantly depend on the patient's NYHA class at the first visit. It is also relevant that prognostic models are related to the NYHA class. Because the numbers of patients with NYHA class I and IV are relative small, we decided to repeat the experiments only for two subpopulations: for patients either in classes I or II and for patients either in classes III or IV.

Additionally, for each event it is also relevant to describe patients not in danger of that event. In order to achieve this goal, the knowledge discovery task is formed in a way that positive cases are those patients that have been monitored for more than two years and have not experienced the event (outcome). The negative cases are patients that have experienced the event within two years. The experiments have to be repeated for both NYHA subpopulations. Similarly, we are also interested in a description of patients for which the event can be expected soon in contrast to those for which the event can be expected later. The positive patients are patients who experienced the event in the first three months after the first visit while negative instances are patients that experienced the event after two years or more.

Finally, it must be noted that the variation of the NYHA class is a special event which enables detection of both improvements and worsening in the patient status. It means that separate models should be constructed for both cases for patients in NYHA classes II and III at the first visit.

Attribute selection

Selection of the target attribute and definition of the target class follows directly from the definition of different data mining tasks. It must also be noted that for each task we use only a part of the data base (example selection).

Input data attribute selection is straightforward. All attributes related to the patient identification and events (outcomes) are eliminated, keeping only the information related to the first visit. Additionally, we also eliminated 5 attributes describing medication that is present at the first visit. The reason for doing so is that HF medical treatment has changed over the last years significantly and because we do not want to use information about medication that involves a lot of medical history information into prognostic models whose intention is to be based on exact and measurable facts.

Feature selection and construction

Feature selection is not relevant because this dataset contains a small number of attributes. In contrast to that, feature construction is necessary.

At first, patient age has been introduced instead of the date of birth. Age has been calculated as the difference between the date of birth and the date of the first visit (in years). Additionally, for each event we had to substitute the date of the event with the difference between the first visit and the event in days. This information is essential for the determination of the respective target class examples.

Finally, it is known that ejection fraction (EF) is usually calculated by echocardiography devised by using left ventricular diameters. The most used formulas for EF calculation are:

Cubic formula: $$\frac{LVTDD^3 - LVTSD^3}{LVTDD^3}$$

Squared formula: $$\frac{LVTDD^2 - LVTSD^2}{LVTDD^2}$$

It seems that some EF values reported in the dataset are calculated with the first formula, and some other ones with the second one; moreover some EF values seem to be calculated in some other manner. Thus we decided to add to the dataset the EF values calculated with standard formulas. Subsequent experiments will demonstrate which EF definition is more informative for prognostic purposes.

Data cleansing

Many attributes in the original dataset show non-realistic values; in particular, many values are not compatible with human physiology, and are probably typing errors. In order to solve this problem, we have defined upper and lower bounds for each attribute. Then we have substituted the out of range values with unknown values ("?"). The following table presents the used ranges for data cleansing (Table 8-4):

Table 8-4. Functionalities of the KD-ETL component

Attribute	Range	Notes
Age	18 – ∞	We retain only adults
Weight	40 – 140	
Height	140 – 200	
BMI	14.2 – 45.3	
PAS	80 – 220	
PAD	49 – 120	
Heart Rate	37 – 140	
EF (Ejection Fraction)	8 – 55	Usually, HF patients have a very low EF. Thus, we have deleted patients with high EF values
LVTDD (Left Ventricular Diastolic Diameter)	40 – 90	
LVTSD (Left Ventricular Systolic Diameter)	19 – 84	
RATIOCARD	0.38 – 0.8	
Sodium	125 – 155	
Potassium	2.53 – 6.4	
Haemoglobin	6.9 – 20	
Creatinine	0.34 – 3.97	
Glycaemia	40 – 381	
Uricaemia	1.7 – 14	
CHOLETOT (Total Cholesterol)	75 – 347	
CHOLEHDL (HDL Cholesterol)	15 – 98	

Additionally, based on age and ejection fraction we decided to eliminate patients less than 18 years old and also those patients with ejection fraction bigger than 55. Finally we detected and eliminated seven patients where it has been detected that some events had happened before the first visit. The final dataset has 17662 examples.

9. Optional health care management and patient monitoring data marts

While KD-ETL is the main tool for preparing the data from HFP-DW for knowledge discovery tasks, the use of OLAP technology upon the data stored in HFP-DW, provides means for constructing views to the data that might be of use for medical staff or care teams as well as for health care managers in their decision making processes. In a limited way OLAP can also be used as data preparation tool for subsequent data mining or knowledge discovery. Within HEARTFAID platform, both of these functionalities will be optional, and their use will depend on the user capabilities. However, both medical staff and health care managers will have predefined views to either patient monitoring data or basic health care management data. Generic multidimensional model for patient monitoring is given in Figure 9-1. Main fact table contains constantly monitored data, while main dimensions are: Time, HFP Environment, Patient_State, Patient characteristics (demography, anamnestic features).

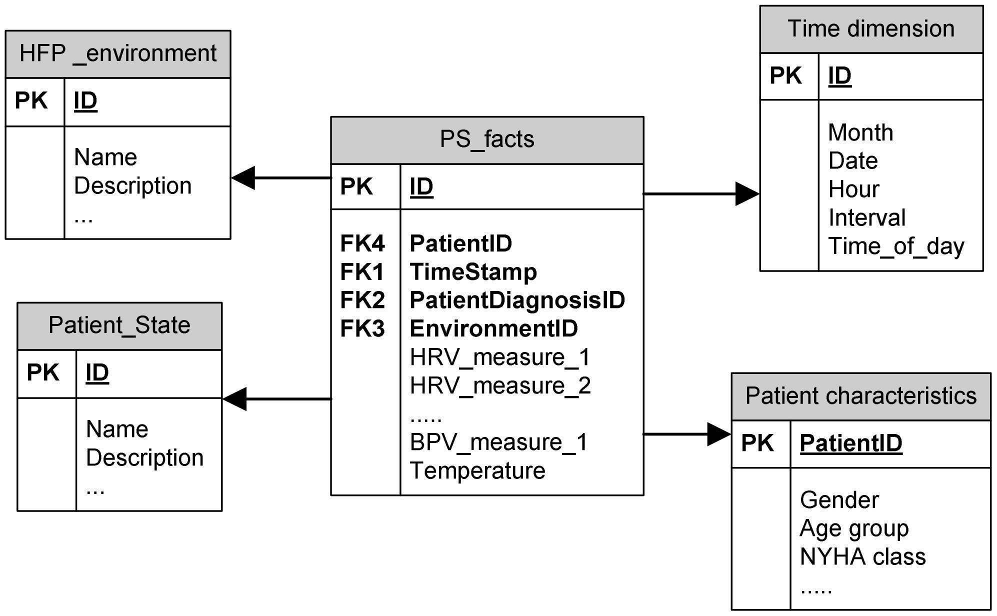


Figure 9-1. Generic template for HFP Patient monitoring (PM) data mart.

Figure 9-2. depicts a generic multidimensional model for the health care management within HEARTFAID platform.

Generic multidimensional models defined in Figures 9-1. and 9-2. are constructed from the data marts within HFP-DW, using Cube Designer of the Pentaho platform. Constructed data cubes are subsequently viewed using Mondrian OLAP tool, which provides diverse capabilities typical for OLAP viewing and real-time data aggregation. Details of functionalities provided by Cube Designer and Mondrian are given in the following sections.

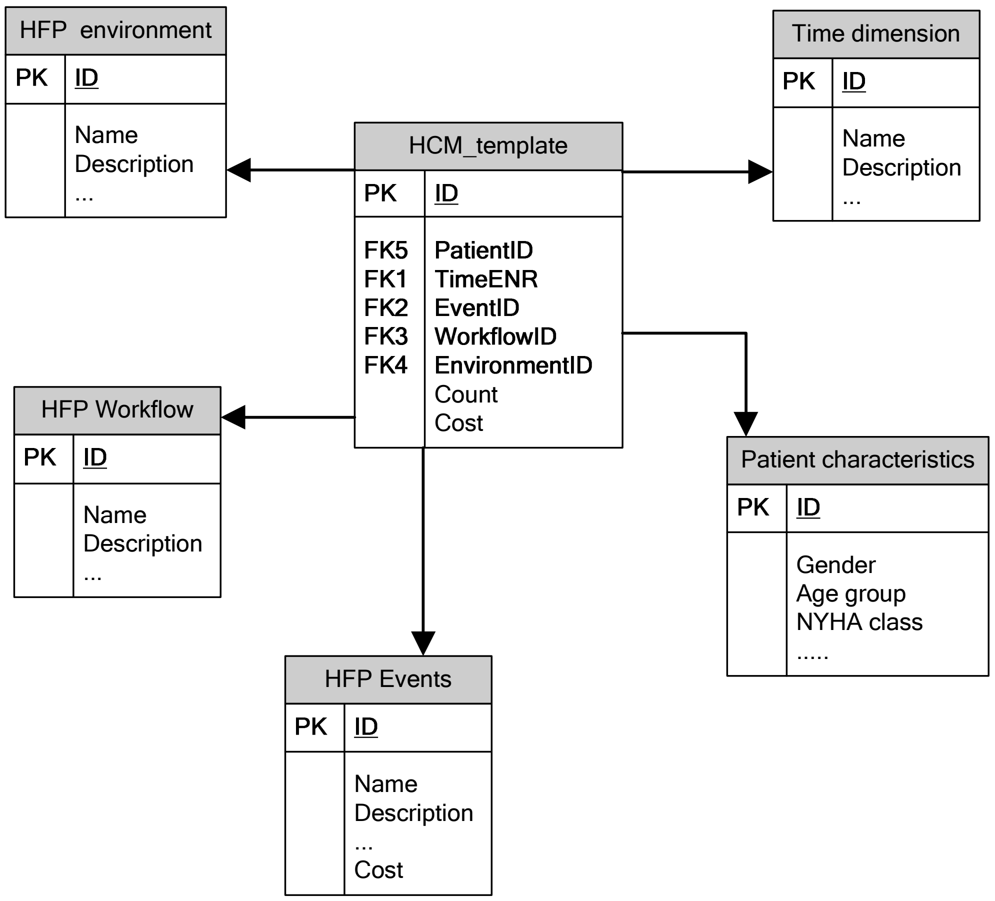


Figure 9-2. Generic template for HFP health care management (HCM) data mart.

9.1. Pentaho Cube Designer: data cube design development tool

Cube designer [1] is a Java application for designing data cube schemas for subsequent use by Mondrian OLAP tool. Cube designer supports three cube schema types: single table schema (both measures and dimensions are from a single table), star schema (single fact table and multiple dimension tables), snow flake schema (single fact table and hierarchical dimension tables). Cube Designer supports any JDBC compliant database, and JDBC drivers for MySQL, Oracle, Microsoft SQL server databases are included in the distribution. The process of the cube design is very simple, and when the hypercube is designed, its structure is exported (in XML format) for the direct, subsequent use by Mondrian OLAP tool.

9.2. Pentaho Mondrian OLAP tool

Mondrian [1] is an OLAP engine written in Java. It executes queries written in the MDX language, reading data from a relational database (RDBMS), and presents the results in a multidimensional format via a Java API. Mondrian OLAP tool will be used for exploratory data analysis within HEARTFAID platform portal, more specifically for:

- Interactive analysis of large or small volumes of information;
- "Dimensional" exploration of data, for example analyzing patient data by patient environment, time of day and/or by time period;
- Parsing of Multi-Dimensional eXpression (MDX) language into Structured Query Language (SQL) to retrieve answers to dimensional queries;
- High-speed queries through the use of aggregate tables in the RDBMS;
- Exporting user defined queries/views into EXCELL tables for knowledge discovery

Pre-defined data cubes and Mondrian views for patient monitoring and health care management can be produced with intention to give instant view to key patient data or key performance indicators of the HEARTFAID platform care processes. Additionally, Mondrian tool can be used by medical staff, knowledge workers or health care managers for exploratory data analysis and obtaining tailored reports or data summaries for specific purposes. Exporting data to e.g. EXCEL tables provides an opportunity to prepare data for new, not foreseen data mining or knowledge discovery problems, which are not provided by KD-ETL tool query and transformation functionalities.

9.3. Bibliography

[1] *Pentaho Open Source business intelligence*, web-site, URL: http://mondrian.pentaho.org/

10. Conclusions

The deliverable presents the results of the tasks T4.1 and T4.2 of the Work Package WP4. It is prepared by UNICAL and RBI and it specifies the data warehousing and data preparation processes for knowledge discovery tasks. The deliverable begins with general discussion of the data warehousing and data preparation procedures specifically for medical applications followed by detailed description of these tasks in the HF platform.

The ultimate goal of the work, together with task T4.3, is implementation of most appropriate knowledge discovery process for data collected by the platform. The platform integrates data collected in the specialized institutions during the follow-up period and the continuously monitored data collected in the home or on-the-move environment. This integration enables data analyses process that is potentially much more powerful then when such data are separately collected. In this respect expectations from the knowledge discovery process are very high. They are explicitly formulated in detailed specification of knowledge discovery goals.

The unique property is that both data collected in follow-up visits and continuously monitored data are time sequences which require special attention and procedures in the knowledge discovery process. The task of the data warehousing is to ensure high quality data extraction, unification, and integration of the collected patient data. Optional, but potentially very relevant are its parts for health care management and patient monitoring decision support functionality.

In the data preparation phase we have in detail elaborated the process of flattening of short and long time sequences. Additionally, templates for data sets construction that include flattened attributes have been developed and tested.

In both fields very modern techniques have been adapted for implementation into real platform environment. Their integration presents potentially relevant scientific advancement. The quality of the final result now depends on successful extraction of real platform data and selection of most appropriate machine learning algorithms for prepared knowledge discovery tasks.

Appendix A: Descriptor specification for KD process

The appendix presents the complete list of descriptors for the KD process. The list is based on the case report form and the descriptors represent data collected during patient visits to specialized medical institutions. The descriptors are divided in the group of static data presented in Part A1 and time sequence data presented in Part A2.

The descriptors in the first group are attributes of the static patient data template building block which is defined in Section 8.2.1. For every descriptor in Part A1 one attribute is generated in the static patient data block. It consists of 14 numerical and 30 categorical attributes.

The later group of descriptors is used as attributes in building block visit Vx . Every descriptor in Part A2 generates one attribute in the block visit Vx. In total this building block consists of 101 numerical and 71 categorical attributes. Additionally, data represent time sequences that can be flattened. Flattening process is described in Section 8.2.3 and it results in 7 and 12 attributes for every categorical and numerical descriptor, respectively. In total 1709 attributes constitute the box called summary before Vx data.

Descriptors are presented in groups corresponding to the data groups defined in the case report form. Each descriptor consists of:

- descriptor name (in italics)
- descriptor type (in bold) where N denotes numeric, C categorical, and Cb categorical binary type
- comments about used measure for N type, categories for C type, and the way the attribute is computed.

The descriptor type determines the way the corresponding attribute will be handled by the knowledge discovery algorithms and the way the flattening of the corresponding time sequence is performed.

A1 Static patient data

Demographic data

age **(N)**- computed as the difference between date of birth and the date of visit V_x (in years)

sex **(Cb)** - male, female

centre **(C)** - MIL1, MIL2, KRK, CAT

Cardiovascular history

previous_cardiovascular_disease **(Cb)** - yes, no

congestive_heart_failure **(Cb)** - yes, no *congestive_heart_failure_duration* **(N)** - computed in months (if no then unknown value)

coronary_heart_disease **(Cb)** - yes,no computed from the list

coronary_heart_disease_specific **(C)** - if yes specify (i.e. stable_angina_past, stable_angina_present, myocardial infarction, ..) if none specified then no

serious_ventricular_arrhythmias **(Cb)** - yes, no

clinical_significant_valvular_heart_disease **(Cb)** - yes, no

clinical_significant_valvular_heart_disease_specific **(C)** -if yes specify (i.e. mitral_stenosis, mitral_regurgitation, ..) if none specified then no

congenital_heart_disease **(Cb)** - yes, no

history_of_rheumatic_fever **(Cb)** - yes, no

history_of_rheumatic_fever **(Cb)** - yes, no

hypertension **(Cb)** - yes, no

peripheral_vascular_disease **(Cb)** - yes, no

dyslipidemia **(Cb)** - yes, no

Other medical history

endocrine_disorders **(Cb)** - yes, no

endocrine_disorders_specify **(C)** - if yes specify (diabetes, or hypothyroidism, or hyperthyroidism) if none specified then no

renal_failure **(Cb)** - yes, no

respiratory_disorders **(Cb)** - yes, no

respiratory_disorders_specify **(C)** - if yes specify (chronic, or bronchial, or ..) if none specified then no

anemia **(Cb)** - yes, no

others_related_to_heart_failure **(Cb)** - yes, no

others_related_to_heart_failure_specify **(C)** - if yes specify (connective_tissue_diseases, or ….) if none specified then no

others_not_related_to_heart_failure **(Cb)** - yes, no

*others_not_related_to_heart_failure*_specify **(C)** - if yes specify (gastrointestinal, or hepatic, or) if none specified then no

Life style information

smoking **(Cb)** - yes, no

alcohol_use **(Cb)** - yes, no

*physical activity***(C)** - sedentary, or moderate, regular

Family history

primary_cardiomyopathy **(Cb)** - yes, no

Anamnesis

fatigue_distance **(N)** - in meters

orthopnoe_in_the_ past **(Cb)** - yes, no

Quality of life questionnaire

minnesota_total_score **(N)**

SF-3_score_physical_functioning **(N)**

SF-3_score_role_physical **(N)**

SF-3_score_bodily_pain **(N)**

SF-3_score_general_health **(N)**

SF-3_score_physical_component_summary **(N)**

SF-3_score_vitality **(N)**

SF-3_score_social_functioning **(N)**

SF-3_score_role_emotional **(N)**

SF-3_score_mental_health **(N)**

SF-3_score_mental_component_summary **(N)**

A2 Time sequences from follow-up visits

Anamnesis

fatigue **(Cb)** - yes, no

dyspnoe **(Cb)** - yes, no (equal dyspnea_in_the_past for baseline evaluation)

nocturnal_dyspnoe **(Cb)** - yes, no (equal nocturnal_dyspnea_in_the_past for baseline evaluation)

peripheral_oedema **(Cb)** - yes, no

Changes in cardiovascular status (all unknown for baseline evaluation)

fatigue_change **(C)** - no, up, or down

dyspnoea_change **(C)** - no, up, or down

nocturnal dyspnoea_change **(C)** - no, up, or down

blood pressure_change **(C)** - no, up, or down

required hospitalization for CHF **(Cb)** - yes, no

Cardiovascular status

coronary_heart_disease **(Cb)** - yes,no computed from the list

coronary_heart_disease_specific **(C)** - if yes specify (i.e myocardial inferction, unstable ungina, ..) if none specified then no

rhythm_or_conduction_disturbances **(Cb)** - yes, no (unknown for baseline evaluation)

pacemaker_implantation **(Cb)** - yes, no

ICD_implantation **(Cb)** - yes, no

cerebrovascular_events **(Cb)** - yes, no

cerebrovascular_events_specify **(C)** - if yes specify (stroke or TIA) if none specified then no

history_of_deep_vein_thrombosis **(Cb)** - yes, no

pulmonary_embolism **(Cb)** - yes, no (unknown for baseline evaluation)

other_cardiovascular_event **(Cb)** - yes, no (unknown for baseline evaluation)

required_hospitalization_other_than CHF **(Cb)** - yes, no (unknown for baseline evaluation)

Other changes in health status (all unknown for baseline evaluation)

renal_failure_worsening **(Cb)** - yes, no

recent_creatinine_level **(N)**

pulmonary_function_worsening **(Cb)** - yes, no

anemia_worsening **(Cb)** - yes, no

hepatic_function_worsening **(Cb)** - yes, no

smoking_cessation **(Cb)** - yes, no

other_relevant_changes **(Cb)** - yes, no

Currant cardiovascular therapy

beta_blocker **(Cb)** - yes, no

beta_blocker_drug **(C)** - if yes specify else no

beta_blocker_dosage **(N)** - if yes specify in *mg* else 0

ACEI **(Cb)** - yes, no

ACEI_drug **(C)** - if yes specify else no

ACEI_dosage **(N)** - if yes specify in *mg* else 0

ARB **(Cb)** - yes, no

ARB_drug **(C)** - if yes specify else no

ARB_dosage **(N)** - if yes specify in *mg* else 0

diuretic **(Cb)** - yes, no

diuretic_drug **(C)** - if yes specify else no

diuretic_dosage **(N)** - if yes specify in *mg* else 0

potassium_sparing_diuretic **(Cb)** - yes, no

potassium_sparing_diuretic_drug **(C)** - if yes specify else no

potassium_sparing_diuretic_dosage **(N)** - if yes specify in *mg* else 0

antiplatelets **(Cb)** - yes, no

antiplatelets_drug **(C)** - if yes specify else no

antiplatelets_dosage **(N)** - if yes specify in *mg* else 0

anticoagulants **(Cb)** - yes, no

anticoagulants_drug **(C)** - if yes specify else no

anticoagulants_dosage **(N)** - if yes specify in *mg* else 0

antiarrhythmics **(Cb)** - yes, no

antiarrhythmics_drug **(C)** - if yes specify else no

antiarrhythmics_dosage **(N)** - if yes specify in *mg* else 0

digoxin **(Cb)** - yes, no

digoxin_drug **(C)** - if yes specify else no

digoxin_dosage **(N)** - if yes specify in *mg* else 0

nitrate **(Cb)** - yes, no

nitrate_drug **(C)** - if yes specify else no

nitrate_dosage **(N)** - if yes specify in *mg* else 0

Physical examination

weight **(N)** - in *kg*

BMI **(N)** - computed from weight and height

SBP **(N)** - from two measurements select the one more near to 120

DBP **(N)** - from two measurements select the one more near to 80

HR **(N)** - from two measurements select the lower value

respiratory_rate **(N)** - from two measurements select the lower value

heart_sound_third **(Cb)** - yes, no

heart_sound_fourth **(Cb)** - yes, no

heart_murmurs **(Cb)** - yes, no

pulmonary_rales **(Cb)** - yes, no

liver_enlargement **(Cb)** - yes, no

peripheral_oedema **(Cb)** - yes, no

peripheral_oedema_specific **(C)**

jugular_vein_congestion **(Cb)** - yes, no

hepatojugular_reflux **(Cb)** - yes, no

body temperature **(N)** - in Celsius

NYHA_class **(N)** - 1-4

Laboratory assessment

Hb **(N)** - in *g/l*

Ht **(N)** - in %

red_blood_count **(N)**

white_blood_count **(N)**

platelet_count **(N)**

sodium **(N)** - in *mmol/*

potassium **(N)** - in *mmol/*

glucose **(N)** - in *mmol/*

creatinine **(N)** - in *mmol/*

urea **(N)** - in *mmol/*

AST **(N)**

ALT **(N)**

total_cholesterol **(N)** - in *mmol/*

LDL_cholesterol **(N)** - in *mmol/*

HDL_cholesterol **(N)** - in *mmol/*

tryglicerides **(N)** - in *mmol/*

TSH **(N)** - in *mmol/l*

creatinine_clearance **(N)** - in *ml/min*

glycated_Hb **(N)** - in %

BNP **(N)** - in pmol/l

blood_samples_for_DNA_RNA_obtained **(Cb)** - yes, no

Chest X-ray

cardio_thoracic_ratio **(N)** - in %

features_of_pulmonary_congestion/oedema **(Cb)** - yes, no

12-lead electrocardiography

heart_rhythm_sinus **(Cb)** - yes, no

heart_rhythm_specify **(C)** - if no specify else yes

heart rate **(N)**

conduction_PQ **(N)** - in ms

conduction_QR **(N)** - in ms

conduction_QT **(N)** - in ms

pathological_Q_waves **(Cb)** - yes, no

ST_depression **(Cb)** - yes, no

ST_depression_max_value **(N)** - in *mm*

T_wave_changes **(Cb)** - yes, no

LVH_ECG **(N)** - in *mm*

Echocardiography

left_ventricle_end_diastolic_diameter **(N)** - in *mm*

left_ventricle_end_systolic_diameter **(N)** - in *mm*

left_ventricle_interventricular_septum_diastolic_thickness **(N)** - in *mm*

left_ventricle_posterior_wall_diastolic_thickness **(N)** - in *mm*

left_ventricle_end_diastolic_volume **(N)** - in *mm*

left_ventricle_end_systolic_volume **(N)** - in *mm*

left_ventricle_ejection_fraction_2D **(N)** - in *%*

left_ventricle_ejection_fraction_4D **(N)** - in *%*

right_ventricle_end_diastolic_diameter **(N)** - in *mm*

left_atrium_anteroposterior_diameter **(N)** - in *mm*

aorta_root_diameter **(N)** - in *mm*

aorta_ascending_aorta_diameter **(N)** - in *mm*

mitral_valve_Emax/Amax **(N)**

mitral_valve_ deceleration time **(N)** - in *ms*

mitral_valve_mitral_regurgitation **(N)**

pulmonary_artery_pressure **(N)** - in *mmHg*

contractility **(C)** - specify one of: normal, hypokinesis, akinesis, dyskinesis

24 h Holter electrocardiography

HR24_mean **(N)**

HR24_min **(N)**

HR24_max **(N)**

HR_day_mean **(N)**

HR_day_min **(N)**

HR_day_max **(N)**

HR_night_mean **(N)**

HR_night_min **(N)**

HR_night_max **(N)**

SVE **(N)**

SVTach **(N)**

atrial_fibrillation_flutter **(Cb)** - yes, no

VE **(N)**

VPairs **(N)**

VTach **(N)**

VF **(N)**

pauses_day **(N)**

pauses_night **(N)**

conduction_abnormalities **(Cb)** - yes, no

conduction_abnormalities_specify **(C)** - if yes specify else no

heart_rate_variability_SDNN **(N)**

heart_rate_variability_ADANN **(N)**

heart_rate_variability_rMSSD **(N)**

heart_rate_variability_pNN50 **(N)**

heart_rate_variability_total_power **(N)**

heart_rate_variability_HF **(N)**

heart_rate_variability_LF **(N)**

heart_rate_variability_VLF **(N)**

Six-minutes walking test

BP_baseline_sys **(N)** - in *mmHg*

BP_end_sys **(N)** - in *mmHg*

BP_difference_sys **(N)** - in *mmHg*

BP_baseline_dya **(N)** - in *mmHg*

BP_end_dya **(N)** - in *mmHg*

BP_difference_dya **(N)** - in *mmHg*

HR_baseline **(N)**

HR_end **(N)**

HR_difference **(N)**

SpO2_baseline **(N)** - in *%*

SpO2_end **(N)** - in *%*

SpO2_difference **(N)** - in *%*

walking_distance **(N)** - in *m*

Reason for additional visit (all unknown for baseline evaluation)

CHF_status_improved **(Cb)** - yes, no

CHF_status_worsened **(Cb)** - yes, no

CHF_status_reqires_hospitalization **(Cb)** - yes, no

CHF_status_change_in_therapy **(Cb)** - yes, no

CHF_status_requires_other_advice **(Cb)** - yes, no

time_to_next_visit **(N)** - computed as the difference between date of the scheduled visit and the date of visit V_x (in days)