Chapter 1

Ontology-based Data Integration: A Case Study in Clinical Trials

Sandra Geisler[1], Christoph Quix[1], Anke Schmeink[2], David Kensche[1]
[1]*RWTH Aachen University, Informatik 5 (Information Systems)*
[2]*RWTH Aachen University, UMIC Research Centre*
{*geisler, quix, kensche*}*@dbis.rwth-aachen.de,*
schmeink@umic.rwth-aachen.de

In studies involving medical devices, standardization of data formats of medical devices is still an open problem. Although some standards exist, manufacturers often implement their own proprietary format or use competing standards. Furthermore, especially during the development of a medical device the data format may change. Therefore, flexible means to integrate different kinds of data sources and schemas are required.

To address the challenges in data management for clinical trials, we developed an ontology-based data integration architecture which is capable of creating executable ETL (Extract–Transform–Load) processes based on a user-defined mapping between ontologies. The system has been implemented and evaluated in the field of clinical trials with medical devices.

1.1 Introduction

Clinical trials are a common means in medical research to investigate new medications, medical devices, and other medical products. The data collected during clinical trials is very valuable to the organizations carrying out the trials to show the efficiency and harmlessness of their products or to (dis)prove a thesis. While the study register of the U.S. National Institute of Health contained approximately 23 500 clinical studies in 2006 [Bestehorn *et al.* (2006)], this number quadrupled (81 770) in November

2009. Especially the pharmaceutical industry and manufacturers of medical devices push the conduct of more and more studies. Therefore, rapid and efficient, but also accurate study planning, conduction and analysis of results are indispensable [Hanover and Julian (2004)]. The careful collection, handling and storage of data while obeying national and international regulations is a major task in clinical study management [Prokscha and Anisfeld (2006)].

Heterogeneous data sources are also a challenge in data management for clinical studies: device manufacturers often implement their own proprietary format, or several standards exist for the same kind of data [Fischer *et al.* (2003)]. Furthermore, the data format may change during the development of a medical device. Thus, flexible means to integrate data sources and their schemas are required. Many data integration architectures used in the medical domain facilitate an integrated schema and relational DBMSs as they provide efficient methods to store, query and analyze data [Prokscha and Anisfeld (2006)].

Despite the advantages of an integrated schema architecture, it also poses several problems. Such an architecture requires an immense upfront effort for design and implementation. Furthermore, the Extraction-Transformation-Loading (ETL) processes, including the mappings between the source schemas and the integrated schema, have to be created [Vassiliadis *et al.* (2001a)]. Hence, when a source schema has changed or new sources have to be added, mappings, ETL processes, and possibly the integrated schema have to be adapted as well. Especially in the medical domain, the definition of an integrated common schema is a major issue as there exists a plethora of synonyms, homonyms, and related terms [Gardner (2005)]. Today, a common means to deal with these lingual and domain-specific issues is the use of ontologies. In the medical field taxonomies and ontologies have already been used intensively and constitute an approved method to organize and structure knowledge [Hahn and Schulz (2004)]. However, purely knowledge-based systems lack the mature functionalities for storage, management, integration and analysis of huge amounts of data.

To overcome the described problems, we developed an ontology-based data integration system which is capable of creating executable ETL processes based on a mapping between a *core ontology* which contains a domains' vocabulary and separate *source ontologies.*

In this chapter, we describe the design and implementation of our system *OnTrIS* (Ontology-based Trial Information System), which supports the entire data integration process: the sources and the integrated schema are described by source ontologies and core ontology, respectively, and the integrated schema is created from a user-defined subset of the core ontology. Furthermore, mappings can be defined between core and source ontologies to enable the assembling of the integration process by ready-to-use data integration modules. Thus, we provide an end-to-end, ontology-based data integration solution which has not been shown in other approaches [Wache *et al.* (2001); Bellatreche *et al.* (2004); Skoutas and Simitsis (2006); Bergamaschi *et al.* (2009)]. The system has been evaluated in the field of clinical trials with medical devices trial conducted in the MyHeart subproject Heart Failure Management (HFM) [Philips Research Europe (2008)]. MyHeart aims at developing intelligent systems for the prevention and monitoring of cardiovascular diseases.

The remainder of the chapter is structured as follows: in the next section we briefly show the overall architecture of OnTrIS. Section 1.3 presents the core ontology of our approach and Section 1.4 explains the data integration approach. Section 1.5 details how we implemented the data integration processes. In Section 1.6, we present the evaluation results, which show the feasibility and usability of our approach. Section 1.7 discusses related work, before we conclude in the last section and point out future work.

1.2 System Architecture and Overview

Multidisciplinary teams in clinical trials include domain experts, such as medical professionals, physicians, chemists, or electrical engineers which are entrusted with the development of a medical product and responsible for the conduct of the trial. As these domain experts are usually not familiar with data management, we provide a comprehensible and fast approach to construct and deploy trial databases and corresponding data integration modules to import trial data from different sources. In this section, we will first describe the main constituents of OnTrIS and the basic steps to deploy a data management architecture for a clinical trial.

The architecture (see also Figure 1.1) contains first of all an extendable core ontology describing concepts and attributes of clinical trials called *Clinical Trial Data Management Ontology* (CTDMO). It is the basis for the clinical data management environment which will be created by OnTrIS. It is used to create new relational database schemas and data integration modules based on mappings between data source ontologies and concepts selected from the core ontology. To include trial specific concepts the user

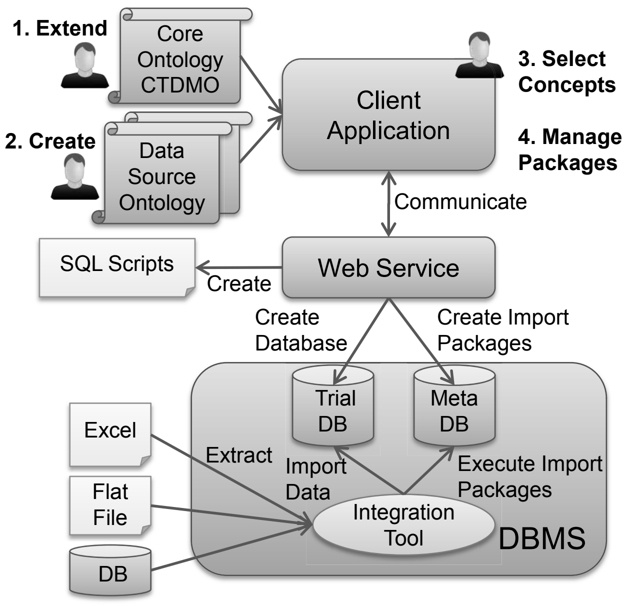


Fig. 1.1: The Workflow in the OnTrIS Architecture

can extend the CTDMO using an ontology modeling tool (cf. Section 1.3). The *client application* enables the user to select concepts from the CTDMO which she wants to use in her database, to create a relational database schema on a dedicated database server, and to create mappings and corresponding data integration modules to import data from different data sources. Furthermore, the user can create and manage ETL packages.

A *web service* fulfils all complex tasks, such as the processing of the ontologies, the creation of the database schemas (directly on the DBMS or as SQL scripts), of the meta representation for the created databases and for the data sources, and of the integration modules. A web service has been used to make the system more modular and secure.

Finally, a *data integration tool* is responsible for executing the data integration modules. It accesses the data sources and extracts the data according to the specifications in the data integration modules.

We point out, that the OnTrIS architecture is not restricted to the CTDMO or the field of clinical trials. Any other domain ontology obeying the modeling principles described in Section 1.3 can be used as a basis for the architecture as well.

1.3 The CTDM Ontology

Ontologies have been identified as an approved method to model a domain of discourse on a highly abstracted and semantic level [Gruber (1993)]. Ontologies have been proven suitable for database design, automatic database generation and data integration [Calvanese *et al.* (2002); Wache *et al.* (2001); Zamboulis *et al.* (2008)]. The applicability in data sharing environments has already been pointed out in [Gruber (1993)] as an ontology is a *"a logical description of shared data, allowing application programs and databases to interoperate without having to share data structures."* Therefore, the basis for the OnTrIS data integration is a semantic description of clinical trials in form of an ontology, the Clinical Trial Data Management Ontology (CTDMO, the core ontology).

Several alternative methodologies exist, which can be used to develop ontologies, such as the Enterprise Model Approach, METHONTOLOGY or TOVE (see [Pinto and Martins (2004)] for a comprehensive survey of methods). Many of these methodologies have common steps, such as the definition of the purpose, the conceptualization, and the implementation (or formalization). The work of Noy and McGuinness [Noy and McGuinness (2001)] is a well known approach to manual ontology development and has been used in this work. The steps proposed by Noy and McGuinnness have been supplemented in our work by steps for ontology evaluation [Gomez-Perez (2004)]. As we focus on the data integration and creation of ETL processes in this chapter, we refer the interested reader to [Geisler *et al.* (2007)] for more details on the development. The core principles of the CTDMO are described in the following.

Eight core concepts have been identified to structure the CTDMO. The core concepts are shown as dark ellipses in Figure 1.2 and are described below. The example concepts in the figure have been adapted from the extended Heart Failure Management (HFM) trial ontology, which has been created during the evaluation of the system. There are two types of relationships: *isA*-relationships and object properties. *isA*-relationships represent an inheritance relationship and are illustrated with a thin line. If three dots follow the *isA* keyword, the inheritance relationship is deeper than it is shown, i.e., concepts between the two concepts in the hierarchy are hidden. Object properties are drawn as thick lines in the figure.

The root concepts comprise the following:

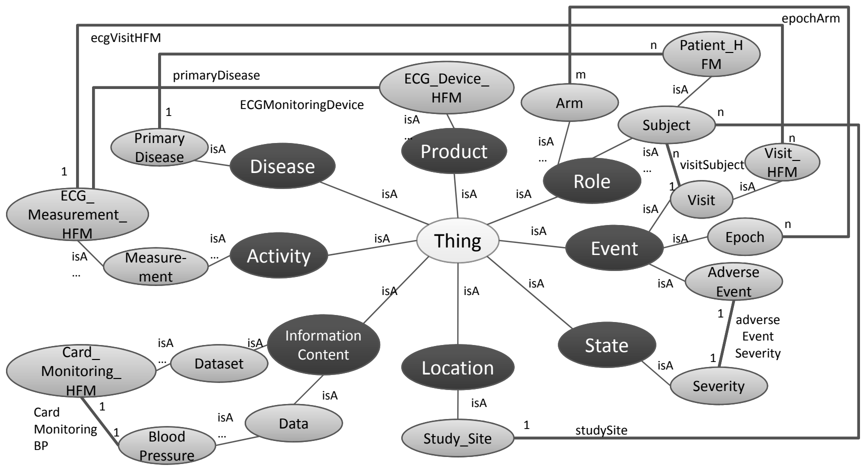


Fig. 1.2: Example Concepts of the CTDMO

Activity subsumes all concepts which express some action executed during a clinical trial. An example is the concept *HFM_ECG_Measurement*.

Product is supposed to contain all things which are used to conduct an activity, e.g., *Composite_Card_Device*. Medical devices are medical products which are used and tested in a study.

Role describes all persons and roles involved in a clinical study, e.g., a *Patient*.

Information_Content contains all concepts which may represent data, e.g., *HFM_Card_Monitoring*, representing a cardiovascular measurement.

Location organizes spatial extents related to the study. The concept *Study Site* is an example for a location.

Event is an occurrence which is happening during a study and usually having a start date. An example subconcept is a *Visit*, which subsumes all information of one visit session.

Disease is a parent concept for all disease concepts. *PrimaryDisease* is a subconcept example.

State is used to subsume concepts that describe a certain state of an object, e.g., *Severity* of an adverse effect.

In the ontology development process we identified requirements for the ontology, where the most important among them demand the ontology

(1) to be extendable, enabling the addition of concepts and properties describing data of new trials;
(2) to have a concise and user-friendly structure to enable fast orientation;
(3) to have a modular design (concepts can be reused to describe other concepts), and
(4) to have a high quality regarding the consistency, documentation and comprehensibility.

To fulfil the aforementioned requirements, we defined modeling principles. Remember that the CTDMO is a core ontology which will be extended by users for each trial to include new concepts and properties. The extensions are stored in the core ontology itself as they are expected to be reused in other trials which leads to a growth of the ontology. First we introduce the notion of *hot spots*. Hot spots are concepts that have to be refined in the core ontology by the users as they are too general to use.

An example for a hot spot is the concept *Subject*, which represents the persons participating in a study. Obviously, the data to be collected for a person varies in each study; therefore, refinement of this concept is necessary. The idea of hot spots accounts for the requirement of extensibility.

Furthermore, object properties have been used to model relationships between concepts and their individuals. To enable the mapping to relational schemas, constraints for object properties in the core ontology have been restricted to be 1:1 (defined as functional and inverse functional), 1:N (defined as functional) and N:M relationships (defined with no restrictions).

For modularity of the ontology and reusability of ontology elements, some terms have been represented as object properties and explicit concepts, which could also have been modeled as datatype properties and simple domains. Especially concepts that represent data units, such as *Blood Pressure* or *Weight*, can be of use for multiple concepts such as *Patient* or multiple measurement concepts. Therefore, these data units are not modeled as datatype properties of the specific concepts but as concepts on their own using inverse functional object properties to reference them.

Starting from scratch building a new ontology is admittedly a comparable or even higher upfront effort compared to other data integration systems. But if a suitable core ontology can be reused, only some adaptations, such as the correct modeling of object properties and the actual extensions for the required purpose have to be done. Once the core ontology has been agreed upon it can be reused and refined and is therefore suitable for long-term application, which is beneficial for recurring problems such as fast setup of clinical trial data management environments.

1.4 Ontology-based Data Integration of Study Relevant Information

An important feature of OnTrIS is, that the mappings between the sources and the integrated system are defined between ontologies on a semantic level. The mappings defined between ontologies are then translated into executable ETL processes which are able to extract the data from the sources and to load them into an integrated database. Figure 1.3 gives an overview of our ontology-based integration approach.

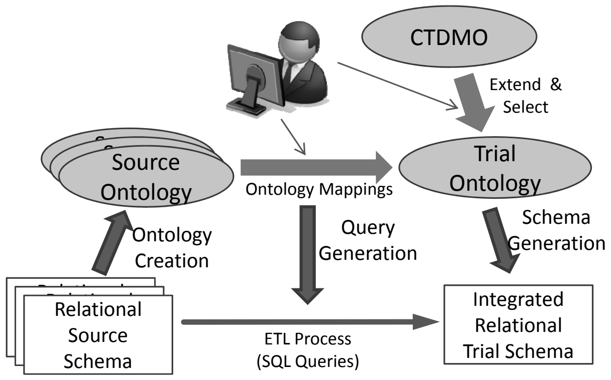
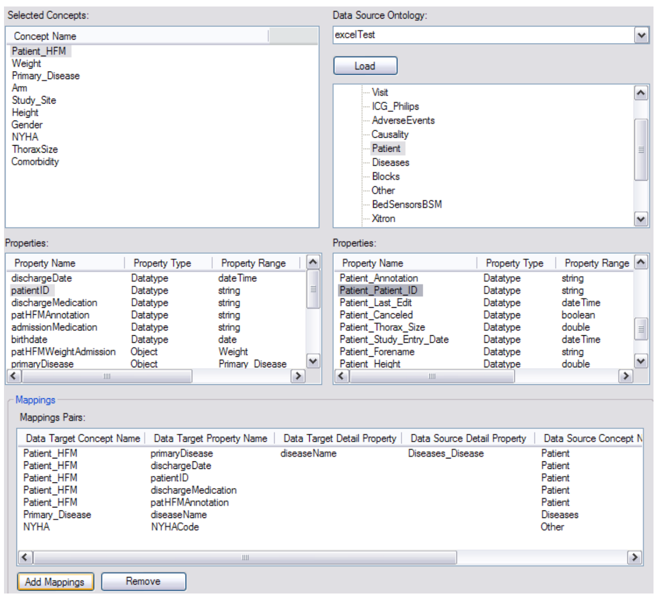


Fig. 1.3: Overview of the Ontology-based Integration Approach

To be able to define mappings between ontologies, these ontologies must be available. The *trial ontology* is created by the user by (i) extending the CTDMO with trial specific concepts and (ii) selecting relevant concepts which are included in the integrated database schema, i.e., the trial ontology represents the integrated schema to be created. The selection is important as not all concepts of the extended ontology are required in the schema. As we follow a *multiple ontology approach* for data integration [Wache *et al.* (2001)], ontologies for each data source, including mappings to the trial ontology, also have to be defined. We use a semi-automatic approach (according to [Bellatreche *et al.* (2004)]) in which the system assists the user in creating the data source ontologies by translating the source schema into an ontology automatically. Furthermore, the definition of the mappings is supported by an easy-to-use graphical user interface (see Figure 1.4).



Concept Name
Patient_HFM
Weight
Primary_Disease
Arm
Study_Site
Height
Gender
NYHA
ThoraxSize
Comorbidity

- Visit
- ICG_Philips
- AdverseEvents
- Causality
- Patient
- Diseases
- Blocks
- Other
- BedSensorsBSM
- Xitron

Property Name	Property Type	Property Range
dischargeDate	Datatype	dateTime
patientID	Datatype	string
dischargeMedication	Datatype	string
patHFMAnnotation	Datatype	string
admissionMedication	Datatype	string
birthdate	Datatype	date
patHFMWeightAdmission	Object	Weight
primaryDisease	Object	Primary_Disease

Property Name	Property Type	Property Range
Patient_Annotation	Datatype	string
Patient_Patient_ID	Datatype	string
Patient_Last_Edit	Datatype	dateTime
Patient_Canceled	Datatype	boolean
Patient_Thorax_Size	Datatype	double
Patient_Study_Entry_Date	Datatype	dateTime
Patient_Forename	Datatype	string
Patient_Height	Datatype	double

Data Target Concept Name	Data Target Property Name	Data Target Detail Property	Data Source Detail Property	Data Source Concept N
Patient_HFM	primaryDisease	diseaseName	Diseases_Disease	Patient
Patient_HFM	dischargeDate			Patient
Patient_HFM	patientID			Patient
Patient_HFM	dischargeMedication			Patient
Patient_HFM	patHFMAnnotation			Patient
Primary_Disease	diseaseName			Diseases
NYHA	NYHACode			Other

Fig. 1.4: The Mapping Editor

In addition, we follow an in-advance (or materialized) integration approach, where the data is stored in an integrated database *in advance* to queries to this database. The in-advance integration facilitates fast query execution, especially with respect to data analysis and reporting. It also allows for auditing and metadata management, which are important features for clinical trial data management and which we implemented to adhere to legal regulations. Another important step in our approach is the translation between the schemas (of the sources and the integrated schema) and the corresponding ontologies. In OnTrIS, we support various types of data sources, including relational databases, flat files, and Excel sheets, which are common formats used in clinical trials. We assume, however, that the schemas of these sources can be represented in a relational way. The translation between ontologies and relational schemas has to be done in both directions, as indicated in Figure 1.3: the relational schemas of data sources are translated into source ontologies, and the trial ontology is translated into a relational schema for the integrated database. We will discuss the details of this translation in Section 1.4.1. Having defined the translation between ontologies and relational schemas, we are then able to specify how the ontology mappings should be translated into an executable query for the ETL process. The details of this query generation process will be discussed in Section Section 1.4.2.

1.4.1 *Translation between Ontologies and Relational Schemas*

The translation between ontologies and relational schemas is a well-studied process which is similar to the classical transformation of Entity-Relationship diagrams to relational schemas. The main difference of OnTrIS to other approaches for the translation between ontologies and relational schemas [Vysniauskas and Nemuraite (2006); Li *et al.* (2006); Cullot *et al.* (2007)] is that we allow to select a subset of concepts of the ontology which should be represented as data elements in the database. All other concepts are ignored. This results in a more efficient database design, avoiding too many or empty relations and attributes.

For the translation of the trial ontology into the database schema, we first translate the concepts into corresponding relations. If a selected concept is related to another non-selected concept by an object property, then for the latter concept a relation is also created to preserve semantic consistency. In subclass hierarchies, each selected concept is translated into a separate relation. Otherwise, inherited properties are added directly to the relation of the selected sub-concept.

We assume that domains and ranges are defined for all properties. Datatype properties are translated to attributes of the corresponding relation with appropriate data types. For object properties, we check the cardinalities: for a 1:1-relationship (functional and inverse functional property), we will add the attributes of the relation of the property range to the relation of the property domain, the relation for property range will be deleted; a 1:N-relationship will be translated into a foreign key constraint; an n:m-relationship will be represented by a separate relation with two foreign keys referencing the relations for the domain and range of the property.

If the ontology contains individuals, then they are translated into records of the corresponding relations.

The reverse translation of relational source schemas into ontologies is almost exactly inverse. However, object properties which are functional and inverse functional (1:1-relationships) will not be created as it is not possible to detect where concepts have been merged due to 1:1-relationships.

All transformations are logged in a metadatabase to identify the columns, tables and relationships to which concepts and properties have been translated to or stem from later. This information is used for the query generation as we will describe in the next section.

1.4.2 *Mapping Creation and Query Generation*

For the description of the mappings we use the following constituents of the OWL Lite language: the ontology is composed of a set of domain concepts C. The concepts are arranged in a concept hierarchy reflecting specialization relationships. Each concept defines a set of properties P (possibly the empty set). The set of properties consists of a set of object properties O and a set of datatype properties D, such that $P = D \cup O$. Each object property has a range concept C_r and a domain concept C_d. We assume $C_r \neq Thing$ and $C_d \neq Thing$, i.e., range and domain of a property must be defined. In the following, we denote concepts, properties and relations and attributes of a data source with the subscript S and concepts and properties of the integrated database (the target) with the subscript T, respectively. A relation R in the data source or data target is defined by a set of attributes $a_1, \ldots, a_n$, written $R = (a_1, \ldots, a_n)$.

A property in an ontology modeled according to the rules described in Section 1.3 can be one of the following kinds:

- a datatype property d
- an inverse functional and functional object property o_{if}, representing a 1:1-relationship
- a functional object property o_f, representing a 1:N-relationship
- an object property without restrictions o_w, representing a n:m-relationship

The query on the data source has to be assembled depending on the mapped properties of the data source ontology and the relations and attributes they represent. Hence, there are 16 possibilities to map properties between source and target (see Table 1.1). This also implies, that if properties representing columns of different data source tables are mapped to the same target table, joins between the data source tables have to be included in the query. Depending on the combination of mapped properties, there

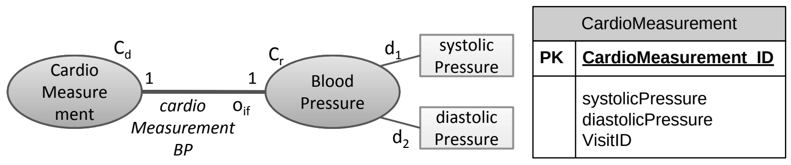


Fig. 1.5: Example for a 1:1-Relationship Mapping

are different problems to overcome which we will address in the following subsections. We assume that a relational language is used for queries. OnTrIS implemented the query construction for SQL queries, but this does not influence the generality of our approach as it can also be adapted to other languages such as XQuery for XML documents.

1.4.2.1 *Mapping of Datatype Properties*

Datatype properties are always transformed to simple relational attributes. If a datatype property d_S of the source ontology is mapped to a property p_T of the trial ontology, the attribute a_d corresponding to d_S has to be included into the assembled query by adding it with the respective relation $R_S = (a_1, \dots, a_d, \dots, a_n)$ to the projection clause (the SELECT-FROM part in SQL). Formalized in relational algebra:

$$\pi_{d_S}(R_S) \tag{1.1}$$

If a datatype property d_T is the target of a mapping, no further action has to be taken. The extracted data from the attribute a_p represented by a property p_S will be imported into the attribute represented by d_T (if their domains are compatible).

1.4.2.2 *Mapping of Object Properties Representing 1:1-Relationships*

The handling of 1:1-relationships differs for properties of the trial ontology and for properties of the source ontology. When an inverse functional object property o_{if_T} of the concepts of the trial ontology is translated using the procedure described in Section 1.4.1, the properties of its range C_r will be added as attributes to the relation representing the domain C_d and no relation will be created for the C_r. Therefore, a mapping involving a property o_{if_T} requires additional input by the user.

The example in Figure 1.5 details the problem. The concept *BloodPressure* contains two datatype properties *systolicPressure* and

Source (S)	**Target (T)**	p_{detail_S}	p_{lookup_S}	p_{detail_T}	p_{lookup_T}
d	d				
d	o_{if}			✘	
d	o_f			✘	✘
d	o_w			✘	✘
o_{if}	d				
o_{if}	o_{if}	✘	✘	✘	
o_{if}	o_f	✘	✘	✘	✘
o_{if}	o_w	✘	✘	✘	✘
o_f	d	✘	✘		
o_f	o_{if}	✘	✘	✘	
o_f	o_f	✘	✘	✘	✘
o_f	o_w	✘	✘	✘	✘
o_w	d	✘	✘		
o_w	o_{if}	✘	✘	✘	
o_w	o_f	✘	✘	✘	✘
o_w	o_w	✘	✘	✘	✘

Table 1.1: The Mapping Matrix

diastolicPressure. The concept *CardioMeasurement* defines an inverse functional object property *cardioMeasurementBP* with *BloodPressure* as range concept. During the translation, a relation *CardioMeasurement* is created which contains two columns *systolicPressure* and *diastolicPressure*. No relation *BloodPressure* is created. If the property o_{if_T} is now mapped, e.g., by a source datatype property d_S, it is not clear, if the data has to be imported into the *systolicPressure* attribute or the *diastolicPressure* attribute. Therefore, the user additionally has to select a data target detail property d_{detail_T} to clarify the situation. The mapping between the source property p_S and d_{detail_T} is then included in the mappings.

If a source property o_{if_S} is mapped to any other property, it can be treated like an object property o_f representing a 1:N-relationship, because 1:1-relationships in the relational schema of the data source are considered as restricted 1:N-relationships. A summary of all mapping combinations and the required mapping components are listed in Table 1.1.

1.4.2.3 *Mapping of Object Properties Representing 1:N-relationships*

We represent 1:N-relationships as functional object properties o_f defined for the concept representing the side with cardinality N (or child side) of the relationship. A mapped property o_{f_T} representing a 1:N-relationship requires more than one attribute and relation to be added to the assembled relational query. First of all, the relations representing the domain concept C_d and the range concept C_r have to be added to the query. Then the attribute representing the foreign key and the primary key attribute of the referenced relation are added to the projection clause. Finally, a left outer join between the two relations is added. In relational algebra this is expressed as:

$$\pi_{a_{fk},a_{pk}}(R_d ⟕ R_r) \tag{1.2}$$

In case of nesting, i.e., if foreign key relationships have more than one level, this is also reflected in the assembled query. For example, a two level foreign key relationship would be expressed in relational algebra as follows:

$$\pi_{a_{fk},a_{pk},a_{fk1},a_{fk2}}((R_d ⟕ R_r) ⟕ R_{r2}) \tag{1.3}$$

Additionally, the user has to select a data source detail property p_{detail_S} for the mapping (which is also added to the projection clause of the statement), because the keys used in the data source relation referenced by the 1:N-relationship are not the same keys used in the target relation. Therefore, the relationships between record sets of the two relations have to be resolved in the data source and again established in the data target (if the target property is also a 1:N or N:M-relationship). This situation is exemplified in Figure 1.6 (a). The illustrated relationships are represented by the functional object properties in the respective ontologies. The detail property on the data source side, e.g., *Disease* in the relation *Disease_S*, is used to identify the correct record in the referenced target relation (*Disease_T* in the example) and to lookup the key of the desired record (note: d_{detail_S} does not necessarily have to be the primary key of the source relation). This key is then used as value for the foreign key attribute of the target relation R_T (*Primary_Disease_ID* in the relation *Patient_T* in the example) representing the domain concept of the respective object property.

In Figure 1.6 (b) the respective mappings are shown. On the left side the input attributes of the data source and on the right side the attributes of the lookup relation *Disease_T* are shown. The mapping is defined between

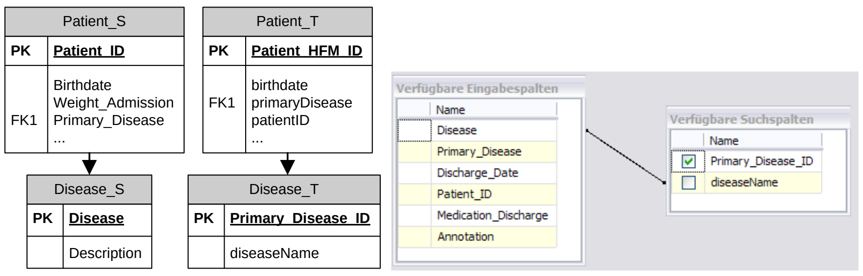


Fig. 1.6: Example for a 1:N-Relationship Mapping

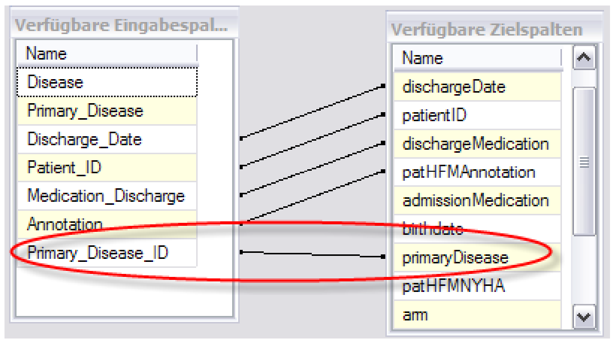


Fig. 1.7: Mapping of Object Properties Representing 1:N-Relationships.

Disease and *diseaseName*, where the former is an attribute of the relation *Disease_S*.

On the target side this implies that an additional step has to be done to lookup the respective key in the referenced relation. The lookup joins the two target relations on a specified attribute (the lookup attribute p_{lookup}) and produces an output attribute, which contains the keys of the rows looked up in the referenced relation. The respective mappings including the described 1:N-relationship mapped to a 1:N-relationship are illustrated in Figure 1.7.

If a datatype property d_S is mapped to a target object property o_{f_T} representing a 1:N-relationship, it is assumed, that d_S already represents the detail property field for the data source used for the join to get the respective key from the target relation R_T. On the other hand, if an object property o_{f_S} representing a 1:N-relationship is mapped to a property d_T, the data from the attribute representing the data source detail property is written to the respective target field.

1.4.2.4 *Mapping of object properties representing n:m-relationships*

n:m-relationships are represented in relational schemas as intermediate relations which have foreign keys referencing each related relation. The set of foreign keys constitutes the primary key of the intermediate relation. Hence, N:M-relationships can be handled similar to 1:N-relationships. In this work N:M-relationships are modeled as unrestricted object properties o_w. Because an intermediate relation has more than one foreign key, the respective data source object property has to be mapped twice to one or more target properties. The user has to take care of the completeness of the mappings. Again, the user has to select a property $d_{detailS}$ to specify the right property to be mapped.

If a property o_{wT} defining an N:M-relationship is mapped, the property has to be mapped twice and the user must also take care of the completeness of the mappings. Furthermore, a detail property has to be defined for each mapping where such a target object property is mapped. If an unrestricted data source object property is mapped to an unrestricted target object property, this mapping has to be done twice with different detail properties on both sides.

1.4.2.5 *Inheritance Relationships*

The case of inheritance relationships in the trial ontology does not have to be taken into account in OnTrIS, because the properties can only be mapped where they occur, i.e., if a property is defined for a concept A, which is specialized by a concept B, the property can only be mapped at concept A. This applies also to the translation rules used: if concept A and concept B are selected by the user, relation B only contains attributes for the properties which have been defined only for concept B; if only concept B has been selected, additionally all properties inherited from parent concepts are translated to attributes of relation B. The same is true for the data source ontologies.

1.5 Assembly of ETL Processes Based on Ontology Mappings

Our approach for ontology-based data integration is a holistic approach which also includes the definition of ETL processes. ETL processes are crucial components of data warehouses and other integration architectures to integrate data from several sources into one global schema. With the information defined in the previous steps, the definition of ETL processes can be done almost automatically. Only administrative information, e.g., database connections or location of files, has to be specified by the user.

In general, an ETL process can roughly be divided into two main parts: the data flow and the control flow. The *data flow* describes the actual flow of data from the sources to the integrated schema, the data target. A wrapper extracts the required data from each source and "pumps" the data into the data flow.

The data flow can be viewed as a part of the *control flow*. In the control flow, organizational operations, such as copying of files or the execution of additional software, can be modeled. We used Microsoft SQL Server Integration Services (MS SSIS) to implement the ETL processes. There are also other commercial tools (see [Friedman *et al.* (2008)] for an overview), open source solutions (e.g., Pentaho [Pentaho Corporation (2009)]) and research prototypes (e.g., ARKTOS [Vassiliadis *et al.* (2005, 2001b)]), but the MS SSIS suited our needs best. They can access various types of data sources and targets, mappings and transformations can be defined easily, and there is not as much administrative overhead as with other integration solutions.

In MS SSIS, data and control flows are organized in integration modules (or packages). A module contains several *components*, each representing a specific "part" of the integration process, such as a data source component or an extraction component. For each data source a separate data integration module is created.

The integration modules in OnTrIS are able to access various source types, such as databases, Excel spreadsheets, and flat files. We provide a wizard to assist the user in creating integration modules.

Firstly, a wrapper for the target database has to be chosen and connection details have to be specified. Secondly, the order has to be determined, in which the data is extracted from the sources by data flow components and loaded into the integrated trial database. The order is important since some relations have to be filled before others, e.g., for lookup of key values or in case of foreign key constraints. Before the data flow components are created, all mapped relations of the target schema are retrieved from the user-defined mappings. Then, the tables are sorted according to their reference order. For example, if a relation *Patient* references a relation *Disease* by a foreign key constraint, the relation *Disease* has to be filled first.

The actual data extraction, transformation and loading is executed in the data flow components. For each relation of the target schema, a separate data flow component is assembled in which all mapped source attributes have to be retrieved from the sources using relational queries as described in Section 1.4. Based on the user defined mappings, foreign key relationships in the target schema are identified and a corresponding lookup component is created which matches a required key in the respective relation to resolve the relationship. Furthermore, it is checked whether a record is already existing in the database, also using a lookup component. Finally, the target component is created, which delivers the data to the target database.

1.6 Case Study and Evaluation

The architecture and its functional components have been evaluated with respect to several criteria in the course of a clinical pilot trial conducted by the Philips Research Europe Laboratories in Aachen.

In addition, the evaluation of the CTDMO with respect to the requirements and quality criteria we described in Section 1.3, we also evaluated the schema, mapping and integration module creation functionality according to usability and feasibility.

1.6.1 *The Heart Failure Management (HFM) Pilot Trial*

MyHeart is an FP6 Integrated Project, aiming to develop intelligent systems for the prevention and monitoring of cardiovascular diseases (CVD). The idea of the project is to develop smart devices for monitoring the health status of patients. Early diagnosis and guidance to a preventive lifestyle through electronic devices help to reduce the risk of CVD [Philips Research Europe (2008)].

The goal of the subproject Heart Failure Management (HFM) is to examine, how a decompensation of heart failure patients, i.e., a severe degradation of the state of health, can be predicted by collecting different vital signs using sensors and manual methods. Within the HFM project, a pilot study is conducted, which examines the quality and usability of special sensors to measure ECG, heart rate and other vital signs of an observed person. The pilot trial is conducted by the Philips research labs and the University Hospital in Aachen.

In addition to the sensor data, the physicians on site capture information gathered during measurements and findings in spreadsheets. There are different cycles of measurements during the trial. Some measurements are executed twice a day, some are running during night and some only take place at admission and discharge of the patients. The data is sent to the research labs once a day where statisticians analyze the data to get insights and to check the applicability. The data and files acquired during this trial have been used to evaluate the OnTrIS architecture. The first step has been the extension of the CTDMO with concepts required to fully describe the study.

1.6.2 *Evaluation of the CTDMO*

As described in Section 1.3, the CTDMO has already some concepts, which structure the ontology on a high abstraction level and that should help the user to easily add new concepts to the ontology.

The domain experts have been asked to extend the CTDMO with concepts and properties of the HFM study, in particular, representing the data which has been gathered by the physicians and scientists on-site.

A questionnaire has been elaborated, which is based on the suggestions for the evaluation of ontologies by Gomez-Perez [Gomez-Perez (2004)]. The questionnaire consists of five parts concerning the consistency, completeness, conciseness, extensibility and usability, and the documentation of the ontology.

Important outcomes according to usability were, that it is important for the users to get a personal introduction to ontologies, the editor, and the CTDMO structures. Due to the level of abstraction and the specific principles the CTDMO is based on, the inexperienced users had difficulties to extend the ontology only with the help of a written guideline.

After the instruction sessions the experts were able to model on their own, but still had some questions. They were, for example, unsure whether they had chosen the right way to model specific concepts and relationships. Their understanding of how to extend the ontology also improved a lot after they had seen the generated database schema. This indicates, that the modelling principles and goals are too strongly connected to the aim to convert the ontology into a relational schema.

1.6.3 *Evaluation of the Mapping Definition and Integration Module Creation*

The data integration part of this work has been evaluated according to two aspects. Firstly, the usability of this part of the application has been assessed. Furthermore, the feasibility of the data integration part is evaluated by discussing the challenges and issues while creating a data integration package for an example data source of the HFM pilot trial.

1.6.3.1 *Evaluation of Usability*

To assess the usability of the ontology-based data integration part of the application a questionnaire based on the ISO norm 9241 part 110 (cf. [International Organization for Standardization (1998)]) has been used. It turned out, that the users could easily generate a data source ontology and also had no problems with the creation of the mappings. The course of required steps were clear to them after a very short introduction. However, the learnability of the tool could still be improved as it required much time to understand the software. Nevertheless, we were satisfied with the results as the software tool is still a prototype, and it addresses a complex task which is hard to understand for non-database experts.

1.6.3.2 *Evaluation of Feasibility*

The data integration mechanism has been tested for the import of a complex Excel file, which is used in the HFM pilot trial to gather information about patients, findings,and the measurements conducted. The Excel file consists of 17 sheets, which are loosely related (relations are defined like in relational databases by keys, but without constraints, which are not supported by Excel). Therefore, this file format was a good example to test the mapping of simple datatype properties as well as mappings of object properties. To integrate the data from the example Excel source files in the database, 49 mappings have been defined. All possible cases (see Figure 1.1) have been tested, and all tests were successful. A typical issue, which occurred during mapping of the Excel data source ontology and the selected concepts, was the question on which level to map the properties. For example, if a concept *Subject* and its child concept *Patient_HFM* have been selected, and the user tries to map inherited properties on the *Patient_HFM* concept level, this will result in an error, because the corresponding table *Patient_HFM* does not contain the mapped property, but its parent table *Subject* does. This can be prevented in the mapping editor by hiding these inherited properties.

The algorithm to create the integration modules also ensures, that all values are inserted, as far as possible. If there is invalid data in an extracted data source row and the integration module cannot convert it implicitly, the row is ignored and redirected to an error log file. If one row is invalid, all other rows can be processed without stopping the ETL process. However, if records of other tables need to reference this record with a foreign key, the required record cannot be found and, thus, a NULL will be inserted instead of a correct ID. It has to be considered in daily use, if it is better to abort the ETL process, when an invalid row occurs, or to log the rows and go on with processing.

1.7 Related Work

Intensive efforts are made by commercial providers creating sophisticated ETL tools [Friedman *et al.* (2008)], which are also applied in the medical and pharmaceutical community [Gardner (2005)]. Nevertheless, ETL process creation based on metadata and holistic data integration approaches are still interesting topics in research. Especially, closing the gap between conceptual and logical ETL representations is still an active research area [Rizzi *et al.* (2006)].

Data Integration Based on Ontologies Ontology-based data integration approaches can be classified according to the number of ontologies used in the approach (single, multiple, and hybrid) and according to the degree of automation of the mapping definition (manual, semi-automatic, and automatic) [Bellatreche *et al.* (2004); Wache *et al.* (2001)].

Examples for single ontology approaches are SIMS [Arens *et al.* (1996)] and COIN [Madnick and Zhu (2005)], which both use a virtual mediator architecture. Other approaches, such as the OBSERVER system [Mena *et al.* (2000)], use multiple existing domain ontologies to describe the data repositories. The virtual integration architecture rewrites user queries based on relationships defined between the ontologies. Bellatreche *et al.* propose an automatic hybrid ontology architecture in which the ontologies for each data source are stored in an ontology-based database [Bellatreche *et al.* (2004)]. Furthermore, a global ontology is defined, which is referenced and can be extended by the local ontologies. The integration is made in two steps: first the ontologies are automatically integrated and subsequently the data.

Another hybrid but semi-automatic approach is MOMIS (Mediator Environment for Multiple Information Sources) [Beneventano *et al.* (2003)]. The system implements a Global-As-View approach. First, the source schemas are created by wrappers and annotated with terms from WordNet. A common thesaurus is generated containing intra- and interschema relationships derived from the annotated source schemas. Based on the thesaurus and the source schemas, the global schema and mappings to the sources are automatically created.

Cruz and Xiao present a hybrid, virtual data integration approach [Cruz and Xiao (2009)], where users can issue queries to the global ontology and these are rewritten and send to ontologies describing the local schemas.

Creating ETL Processes Using Ontologies An early approach [Critchlow *et al.* (1998)] for creating ETL processes uses a single ontology to define the metadata for mediators on a high abstraction level. Concepts of the data sources and target database are mapped with each other by defining mappings inside the ontology. Another approach [Skoutas and Simitsis (2006)] automates the construction of ETL processes using a hybrid ontology principle. A common vocabulary is used to describe the data sources and the target database. Each data source is annotated with the vocabulary and notes for integration. This annotation constitutes the mapping for the data sources to the target database. An application ontology is created for each source according to the annotations. Based on this ontology, ETL processes can be automatically assembled using specifically defined transformation rules.

A current approach [Bergamaschi *et al.* (2009)] combines the data integration approach MOMIS described above with the clustering approach RELEVANT to create the mappings between the global schema and the data sources. They also provide a transformation function which enables the "semantic reconciliation of attribute values", i.e., values of source attributes mapped to an attribute in the global schema are merged to clusters including values with similar meaning. The cluster names are then used as values for the global schema attribute.

Discussion In contrast to the described systems, in this work an *end-to-end architecture* is offered to the user. The user can define the integrated schema as well as the mappings to the integrated schema in one system using an ontology-based approach. In addition, she can define a complete ETL process, which includes the mediator as well as the wrappers (the physical access to the data source). She can also define logistic tasks, such as file handling. Furthermore, the definition of the integrated schema by selecting concepts from the CTDMO, maintains the ontology as it is, hence, making it reusable for other applications. No metadata describing the target or the sources have to be added to the ontology which retains the ontology characteristic of describing domain knowledge.

1.8 Conclusion

Data management in clinical trials poses many challenges: several legal regulations and guidelines have to be obeyed while still providing an efficient and easy-to-use data management system. Furthermore, the users of such a system are often not data management and modeling experts, but they still have to manage the data of clinical trials efficiently such that reports and analyses based on the data can be done easily.

We have presented our ontology-based data integration system OnTrIS. It supports the modeling, management, and integration of data for a specific domain which we showed in a case study in clinical trials. The basis of the system is a core ontology; in the case study, the Clinical Trial Data Management Ontology (CTDMO), describing the field of clinical trials with medical devices. The CTDMO has been developed according to well-known guidelines for ontology development [Noy and McGuinness (2001)], and the evaluation has shown that the developed ontology is suitable for the given task. Modeling principles, such as modularity and structuring by the FDA device classification, have been rated useful within the evaluation.

The creation of the integrated database and the definition of mappings between sources and integrated schema is done on a conceptual level, using an ontology which is derived from the CTDMO by the user. This adaptation process requires additional effort by the user, but it is necessary for effective data management as each clinical trial has specific requirements for the data to be managed. In addition, by giving the user the ability to select a subset of the ontology as the basis for the integrated schema, we avoid unnecessarily complex database schemas and generate a schema which can be easily understood and efficiently queried.

The mappings are translated into executable, relational queries which are the basic building blocks of the ETL processes generated by OnTrIS. The ETL processes are implemented as data integration modules in MS SQL Server Integration Services.

During the usability testing it turned out, that the steps to create an integration module (define data sources, specify mappings etc.) are easily understandable to the users. Still, some users had problems with understanding and learning the mapping and integration functionality, because the full complexity of the integration process could not be completely hidden by our application.

The feasibility check using the HFM pilot trial, revealed strengths and weaknesses of the approach. The automatic creation of the packages and the easy execution is comfortable for the user. The handling of invalid rows of the data source is an issue, because skipping invalid rows can lead to inconsistencies in the data.

In a nutshell, we have shown that integration modules can be automatically created based on ontology mappings. In future extensions of the architecture further ETL tasks should be included, e.g., aggregation functions, the mapping of multiple source properties to one target property, or data type conversions. A generic metamodel [Kensche *et al.* (2007)] might enable the use of a wider range of modeling languages as source and target (e.g., XML Schema or object-oriented modeling languages). Schema matching [Rahm and Bernstein (2001); Quix *et al.* (2007)] can simplify the mapping process by providing candidate matches.

Acknowledgements: This work has been supported by the European research project *MyHeart* (`http://www.hitech-projects.com/euprojects/myheart/`) funded by the European Commission (6th framework, IST 507816) and by the DFG Research Cluster on Ultra High-Speed Mobile Information and Communication *UMIC* at RWTH Aachen University (`http://www.umic.rwth-aachen.de`).

Bibliography

Arens, Y., Hsu, C.-N. and Knoblock, C. A. (1996). Query Processing in the Sims Information Mediator, in *Advanced Planning Technology*, pp. 61–69.

Bellatreche, L., Pierra, G., Xuan, D. N., DehainsalaHondjack and Ait-Ameur, Y. (2004). An a Priori Approach for Automatic Integration of Heterogeneous and Autonomous Databases, in *Proc. Database and Expert Systems Applications (DEXA)*, pp. 475–485.

Beneventano, D., Bergamaschi, S., Guerra, F. and Vincini, M. (2003). Synthesizing an integrated ontology, *IEEE Internet Computing* **7**, 5, pp. 42–51.

Bergamaschi, S., Guerra, F., Orsini, M., Sartori, C. and Vincini, M. (2009). An etl tool based on semantic analysis of schemata and instances, in *Knowledge-Based and Intelligent Information and Engineering Systems*, *LNCS*, Vol. 5712/2009 (Springer), pp. 58–65.

Bestehorn, K., Hönig, R., Clemens, N. and Kirch, W. (2006). Register für klinische Studien - eine kritische Bestandsaufnahme, *Medizinische Klinik* **101**, 2, pp. 120–126.

Calvanese, D., Giacomo, G. D. and Lenzerini, M. (2002). Description logics for information integration, in A. Kakas and F. Sadri (eds.), *Computational Logic: Logic Programming and Beyond: Essays in Honour of Robert A. Kowalski*, no. 2408 in Lecture Notes in Computer Science (Springer-Verlag), pp. 41–60.

Critchlow, T., Ganesh, M. and Musick, R. (1998). Automatic generation of warehouse mediators using an ontology engine, in *5th KRDB Workshop*, pp. 8.1–8.8.

Cruz, I. F. and Xiao, H. (2009). *Complex Systems in Knowledge-based Environments: Theory, Models and Applications*, chap. Ontology Driven Data Integration in Heterogeneous Networks, Studies in Computational Intelligence (Springer), pp. 75–98.

Cullot, N., Ghawi, R. and Yétongnon, K. (2007). DB2OWL: A Tool for Automatic Database-to-Ontology Mapping, in M. Ceci, D. Malerba and L. Tanca (eds.), *Proceedings of the Fifteenth Italian Symposium on Advanced Database Systems*, pp. 491–494.

Fischer, R., Chiarugi, F., Schmidt, J., Norgall, T. and Zywietz, C. (2003). Communication and Retrieval of ECG Data: How Many Standards Do We Need? *Computers in Cardiology* **30**, pp. 21–24.

Friedman, T., Beyer, M. A. and Bitterer, A. (2008). Magic quadrant for data integration tools, Tech. rep., Gartner.

Gardner, S. P. (2005). Ontologies and semantic data integration, *Drugs Discovery Today* **10**, 14, pp. 1001–1007.

Geisler, S., Brauers, A., Quix, C. and Schmeink:, A. (2007). An ontology-based system for clinical trial data management, in *Annual Symposium of the IEEE/EMBS Benelux Chapter 2007*, pp. 53–55.

Gomez-Perez, A. (2004). Ontology evaluation, in S. Staab and R. Studer (eds.), *Handbook on Ontologies*, International Handbooks on Information Systems (Springer), ISBN 3-540-40834-7, pp. 250–273.

Gruber, T. (1993). A Translation Approach to Portable Ontology Specifications. *Knowledge Acquisition* **5**, pp. 199–220.

Hahn, U. and Schulz, S. (2004). Building a very large ontology from medical thesauri, in S. Staab and R. Studer (eds.), *Handbook on Ontologies*, International Handbooks on Information Systems (Springer), ISBN 3-540-40834-7, pp. 133–150.

Hanover, J. and Julian, E. H. (2004). U.S. Clinical Trial Management Systems 2004 Vendor Analysis: Leadership Grid and Market Shares, *IDC* **1**, 32166.

International Organization for Standardization (1998). ISO 9241-11:1998 Ergonomic requirements for office work with visual display terminals (VDTs) - Part 11: Guidance on usability, Tech. rep., International Organization for Standardization.

Kensche, D., Quix, C., Chatti, M. A. and Jarke, M. (2007). *GeRoMe*: A generic role based metamodel for model management, *Journal on Data Semantics* **VIII**, pp. 82–117.

Li, H., Gennari, J. H. and Brinkley, J. F. (2006). Model Driven Laboratory Information Management Systems, in *AMIA 2006 Symposium Proceedings*, pp. 484–488.

Madnick, S. and Zhu, H. (2005). Improving data quality through effective use of data semantics, Tech. rep., Composite Information Systems Laboratory (CISL).

Mena, E., Illarramendi, A., Kahyap, V. and Sheth, A. P. (2000). Observer: An approach for query processing in global information systems based on interoperation across pre-existing ontologies, *Distributed and Parallel Databases* **8**, pp. 223–271.

Noy, N. F. and McGuinness, D. L. (2001). Ontology development 101: A guide to creating your first ontology, Tutorial, Stanford University.

Pentaho Corporation (2009). Pentaho open source business intelligence, `http://www.pentaho.com`.

Philips Research Europe (2008). MyHeart, `http://www.hitech-projects.com/euprojects/myheart/`.

Pinto, H. S. and Martins, J. P. (2004). Ontologies: How can they be built? *Knowledge and Information Systems* **6**, 4, pp. 441–464.

Prokscha, S. and Anisfeld, M. H. (2006). *Practical Guide to Clinical Data Man-*

agement, 2nd edn. (CRC Press Inc).

Quix, C., Kensche, D. and Li, X. (2007). Matching of ontologies with xml schemas using a generic metamodel, in *Proc. Intl. Conf. Ontologies, DataBases, and Applications of Semantics (ODBASE)*, pp. 1081–1098.

Rahm, E. and Bernstein, P. A. (2001). A survey of approaches to automatic schema matching, *VLDB Journal* **10**, 4, pp. 334–350.

Rizzi, S., Abelló, A., Lechtenbörger, J. and Trujillo, J. (2006). Research in data warehouse modeling and design: dead or alive? in *Proc. 9th Intl. Workshop on Data Warehousing and OLAP (DOLAP)* (ACM, Arlington, VA, USA), ISBN 1-59593-530-4, pp. 3–10.

Skoutas, D. and Simitsis, A. (2006). Designing etl processes using semantic web technologies, in *Proc. 9th Intl. Workshop on Data Warehousing and OLAP (DOLAP)* (ACM, Arlington, VA, USA), ISBN 1-59593-530-4, pp. 67–74.

Vassiliadis, P., Quix, C., Vassiliou, Y. and Jarke, M. (2001a). Data warehouse process management, *Information Systems* **26**, 3, pp. 205–236.

Vassiliadis, P., Simitsis, A., Georgantas, P., Terrovitis, M. and Skiadopoulos, S. (2005). A generic and customizable framework for the design of etl scenarios, *Information Systems* **30**, 7, pp. 492–525.

Vassiliadis, P., Vagena, Z., Skiadopoulos, S., Karayannidis, N. and Sellis, T. (2001b). ARKTOS: towards the modeling, design, control and execution of ETLprocesses, *Information Systems* **26**, pp. 537–561.

Vysniauskas, E. and Nemuraite, L. (2006). Transforming Ontology Representation from OWL to Relational Database, *Information Technology and Control* **35**, 3A, pp. 333–343.

Wache, H., Vögele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H. and Hübner, S. (2001). Ontology-Based Integration of Information - A Survey of Existing Approaches, in *Proceedings of Intrenational Workshop on Ontologies and Information Sharing*, pp. 108–117.

Zamboulis, L., Poulovassilis, A. and Wang, J. (2008). Ontology-assisted data transformation and integration, in *Proc. 4th Intl. VLDB Workshop on Ontology-based Techniques for DataBases in Information Systems and Knowledge Systems (ODBIS)* (Auckland, New Zealand), pp. 29–36.

Index