MIX - A Representation Model for the Integration of Web-based Data

Christof Bornhövd

Technical Report DVS98-1

DVS1, Department of Computer Science
Darmstadt University of Technology, D-64283 Darmstadt, Germany
bornhoev@dvs1.informatik.tu-darmstadt.de

November 5, 1998

Abstract

Today, the Internet can be seen as a global marketplace populated by a huge number of providers and consumers that exchange data from a wide range of domains. A combination of data from different sources for further automatic processing is often hindered by differences in the underlying modeling assumptions and representation. In addition, the available sources are in most cases semistructured, i.e., provide no fixed and explicitly specified schema. Thus, an integrated use of Web-based data requires explicit information about its organization and meaning.

In this paper we present a representation model that combines concepts for an explicit description of the structure and semantics of the available data with concepts of a flexible, self-describing model suitable for the representation of semistructured data. We show how data from different sources can be represented in a uniform way on the basis of this model, by integrating data from different online reservation systems as an example. We present a particular way of integrating heterogeneous sources from the Web which is not claimed to be generally applicable, but provides a fairly simple solution for many application domains.

1 Introduction

Since the World Wide Web popularized its existence, the Internet has grown exponentially, leaving its roots as a researchers' forum and entering the collective consciousness. In addition to being a way for individuals and organizations to provide information, businesses have embraced the Internet as a way to offer their services. Today, the Internet is both a vehicle for advertising and a global marketplace of goods and services, ranging from electronic publications to traditional books, from financial services to travel planning, and the online monitoring of traditional logistics and physical distribution of goods.

In all these forms of electronic commerce we can identify common patterns or metaphors: In the *business-to-consumer* metaphor the business advertises and provides a service and an individual typically accesses and extracts the relevant information directly. For this kind of interaction the popular approach of presenting the information in the form of HTML pages is sufficient. The casual user browses, interprets the information and interacts with the provider in a point-and-click paradigm.

In the *business-to-business* interaction the business partners tend to rely on previously established protocols that have been in use for longer time, such as protocols for interbank fund transfers or for reservation of air travel through one of the major reservation systems, for example SABRE or Amadeus.

A third form of interaction is emerging, that may be characterized as *business-to-business-to-consumer*. A typical example of this paradigm is the search for lowest possible fares by a travel agent on behalf of a client. The travel agent is a business that acts as a knowledgeable intermediary. For this kind of service provider the typical point-and-click interaction is too time consuming while the interaction with individual reservation systems is too restrictive since many interesting opportunities are provided by ticket consolidators, last minute providers or are provided only through typical end-user oriented HTML pages. Therefore, in the business-to-intermediary-to-consumer metaphor it is necessary to extract information, consolidate it and use it for further electronic processing.

Unfortunately, the available data sources represent the same or related information in different forms. This is because of different political and cultural contexts, or because of different intentions concerning the use of the data. In particular, an integrated use of the available resources is rarely possible because:

- Most of the data sources available online provide data in a semistructured form. This data has no obligatory and rigid data schema associated with it on which an automatic processing can be based. Thus, the underlying structure and semantics have to be extracted and made explicit before the data can be used.
- In addition, a semantically meaningful use of the available data requires explicit knowledge about the underlying modeling assumptions concerning the organization and the meaning of the data. In most cases, these modeling assumptions are given implicitly only, and thus have to be made explicit.

Therefore, an integrated use of Web-based data requires the extraction of structure and meaning of the data, the explicit characterization of the corresponding metadata, and the consolidation of the extracted information in a common model for further electronic processing.

Our present research was motivated by concrete problems faced by the travel industry in the business-to-intermediary-to-consumer metaphor. It is our belief that this is a major growth area of electronic commerce, and that a mechanism for extracting both structure and semantics of Web-based data and making this information explicit through metadata is an essential enabler for this business model.

In our approach we advocate the use of existing common vocabularies or ontologies as a basis for the interpretation of Web-based data. In the travel industry these are for example the common three letter codes or the UNICORN protocol. Ideally, providers should adhere to those. However, in an imperfect real world, it becomes necessary to extend the existing vocabularies on the consumer side. This is quite realistic in a business-to-intermediary-to-consumer setting, since the intermediary will deal with a finite number of content providers on a regular and extensive basis. Therefore, some initial effort on the part of the consumer of the information is justified if it enables further automatic processing of the information.

In this paper we discuss the role of metadata in describing the structure and semantics of available data. We motivate our approach through a typical but simplified scenario from the travel industry. We introduce a representation model that combines aspects of a flexible, self-describing data model for the representation of semistructured data sources, with aspects concerning an explicit description of the organizational and semantic assumptions underlying the available data. In this way, the proposed model is well suited for the integration of semistructured, heterogeneous Web-based data sources for further processing.

1.1 MIX - A Data Model for the Representation of Semistructured Data

Many of the data sources available online provide information in the form of semistructured data [Abit97, Bune97], such as SGML [ISO86] and HTML [RHJ97] pages, or BibTex [Lamp85] files. Generally, they provide no obligatory, explicitly specified schema in the sense of conventional database systems to which all data must adhere. Accordingly, the available data is represented in a highly irregular way, i.e., data objects that represent the same real world phenomenon may describe different aspects of that phenomenon, or information that may be classified as semantically equivalent is represented differently.

Although this data has no strict and regular schema, it provides some internal structure and is based on certain modeling assumptions that might be used when processing the data. However, in most cases the underlying schema information is given only implicitly, i.e., is given as special tags or by the headings of sections and subsections. This schema information is not as rigid and complete as schema information found in traditional relational or object-oriented database systems.

Furthermore, because the available schema information is part of the data itself, as in the case of SGML or HTML pages for example, it may be modified as frequently as the data it describes. Therefore, in the context of semistructured data frequent modifications of the underlying structure and semantics of the data have to be taken into account. This aggravates the explicit specification of a rigid schema, and has to be taken into consideration when representing and processing these data.

The introduction of flexible data models for the representation of semistructured data in the context of an integrated use of autonomous, heterogeneous data sources, as they are typical for the Internet, is important for two reasons:

- **Management and Representation of Data from Semistructured Data Sources**
  Because semistructured data sources generally provide no such strict and stable schema to which all data must adhere, the available data cannot be easily mapped onto classical data models that are strongly typed. The use of NULL values for the representation of irregular structured data and the need for tracking frequent schema modifications result in unnatural and inefficient representations of the data, and require the introduction of more flexible data models for their efficient representation, management, and exchange.

- **Representation Models for the Integration of Heterogeneous Data Sources**
  In addition, the need for representing irregular structured data arises naturally in the context of integrating data from multiple, heterogeneous sources, even if the available sources themselves are well structured.

During the integration of data sources that have been designed independently, we may encounter different kinds of heterogeneity which can not be easily, if at all, resolved by simply converting local data structures to representation constructs of a common data model. Even if the participating sources are all based on the same perception of a certain domain, different sources typically record different attributes of the same real world phenomena. In addition, semantically equivalent information may be represented by using different modeling constructs or concepts, e.g., using different systems of measurement, scaling, or derivation formula. In most cases, discrepancies of these kinds cannot be resolved directly by the system, because missing information aspects typically can not be obtained, or application-specific information must be taken into consideration to resolve these heterogeneities.

For this reason, data that results from the integration of heterogeneous sources in many cases provides no regular, obligatory structure and representation for all data objects. In addition to that, modifications of the underlying data sources or the integration of additional resources frequently lead to modifications of the integrated data and may necessitate additional integration effort. Therefore, this data may be classified as similar to semistructured data.

Representation models that support a natural representation of data with highly irregular structure, as well as an efficient modification of the schema information underlying these data, obviously provide a significant simplification for the integration of heterogeneous, possibly semistructured data sources [CGH+94, MAG+97].

Most previous approaches for automatic processing of semistructured data concentrate on structural characteristics of the data. They are mainly based on the specification of grammars [ACC+97, AK97, HGC+97] for making the underlying structure explicit, or use browsing-oriented schemas [AMM97, MMM97, SL97] that represent HTML pages as objects with several attributes like URL, title, and author. These approaches do not take the information content, i.e., the meaning of the data into account in a satisfactory way. However, for an integrated use of semistructured data coming from different sources we have to take both the structure and the semantics of the data into consideration.

1.2 Representation of Semantic Context Information by Using Metadata

A semantically correct use and a meaningful exchange of data available from autonomous data sources requires both information concerning their organization and structure, as well as the assumptions about their meaning made by the person or organization owning the data. This information, which we call context information [GMS94], provides the basis for determining the relationships between the data and the real world aspects it describes. With regard to the kind of aspects described by this context information we may distinguish between structural and semantic context information. However, in this paper we will concentrate on the semantic context information.

In most cases, this context information is given only implicitly, i.e., it is in the minds of the responsible designer, is specified in textual documentations not available externally, or is reflected in the local applications operating on the corresponding data. This context information is lost when data is exchanged across institutional boundaries, and thus has to be made available explicitly as metadata.

Atomic or complex data objects in the sense of object-oriented database systems correspond to instances of certain data types defined for the respective data source. The meaning and possible use of such data objects are first determined by their underlying data type. However, even if the type of a given data object describes some semantic aspects of the corresponding data objects, many of the modeling assumptions remain implicit.

As a prerequisite, to integrating data from different sources these semantic assumptions have to be made explicit. For this, different approaches have been discussed in the literature [CRE87, SM91, SSR92, ON94, GMS95, RS95, KS96, HMV96], which all fall into the spectrum characterized by the following extremes: the coding of semantic context information by introducing specific data types (e.g., [CRE87]), and the representation of context information in the form of additional "meta-attributes" outside of the underlying type system.

The latter approach corresponds to the specification of variable attribute sets similar to Lisp property lists [Stee90], which describe different aspects of a given data object. These meta-attributes may be represented as explicitly stored or virtual attributes [SSR94, KS96, RS95] which can be made available directly as part of the data, or may be given in the form of rules defining the association of data objects with their semantic properties [SM91, ON94, GMS95]. By using additional meta-attributes for an explicit description of the semantic properties of a data object the conversion between different semantic contexts has to be implemented by the explicit introduction of appropriate conversion functions.

In contrast, when using specific data types for the representation of semantic context information all semantic properties of a data object are determined by its associated type. In this case, the transformation of a data object between two representations concerning different semantic contexts may be reduced to the appropriate type conversions between the underlying type system of the source and the sink of the data, respectively. These conversions are well known from object-oriented programming languages.

Accordingly, the use of specific data types for the representation of semantic context information provides the following advantages:

- This approach corresponds to the representation of semantics in object-oriented language models, and therefore allows a natural representation of context information on the basis of object-oriented type models. No additional concept *meta-attributes* has to be introduced.
- The mechanism of type conversion is a concept well known from many programming languages. In particular, we may rely on technologies for automatic type conversion, derivation of resulting data types, and static type checking.

On the other hand the following reasons can be given against this representation approach:

- The need for the specification of an according type for each data object generally leads to a significant growth of the underlying type system. In the worst case, the number of data types corresponds to the product of all semantic aspects to be represented. For example, to represent the format (e.g., *LaTeX*, *Postscript*, *Word Perfect*, etc.), as well as the possible compression of text files we may specify the following type system: *LaTeX_Uncompr*, *LaTeX_Compr*, *Postscript_Uncompr*, etc., for the classification of the possible instances. In this case, the underlying type system is no longer suitable for structuring the available types, and describing the structural and semantic similarities of the corresponding data objects in a clear way [RS95, SSR94].

- Generalization and specialization relationships, as they are specified in an object-oriented model, are, more often than not, not suitable for the representation of information about the semantic relationships between different data objects. Specialization relationships provide inheritance for attribute and method declarations only. In many cases, the "inheritance" of certain attribute values would be more appropriate. In this way, the class *MonetaryQuantity* may be understood as a subclass of *PhysicalQuantity* with the attribute value $PhysicalDimension = ``Monetary''$, for example.

- In addition, type hierarchies of this kind do not support bidirectional conversion functions, i.e., automatic type conversion is only possible from sub- to super-types. Accordingly, mapping functions for type conversions between data types belonging to the same hierarchical level have to be specified explicitly. Therefore, in general, type conversions in hierarchically structured type systems seem to be as costly as in the case of meta-attributes for the representation of context information [RS95].

- In general, it is impossible to explicitly describe all modeling assumptions underlying a given data object [SG89]. Thus, an explicit representation of context information always has to be seen as incomplete. For this reason, the representation of context information by using specific data types is often not suitable, because it requires the specification of all context aspects to be described before the mapping of data to the type system. In contrast, the use of a variable set of meta-attributes provides a much more flexible representation approach, because the set of context aspects may be extended any time, dependent on the available information and the intended use of the data.

- Furthermore, many of the data sources available via the Internet provide data in semistructured form only. These sources have no explicitly specified schema to which all data corresponds. Thus, structure and semantics of data objects may vary, even if these describe entities of the same class of real world phenomena. For this reason, context information concerning the organization and meaning of this data has to be given on an extensional level, i.e., on the level of data values and instances. Accordingly, the use of variable sets of meta-attributes allows a much more natural and efficient representation of context information for semistructured data.

For these reasons, the approach presented in this paper for the representation of data with their semantic context lies in the middle of the two alternatives discussed above. As depicted in Figure 1, the represented data is grouped under an appropriate concept, similar to a domain-specific data type. This concept is taken from a predefined conceptualization, which we call an ontology here. This ontology can be understood as a domain-specific vocabulary onto which the available data has to be mapped.

However, the set of context aspects, i.e., meta-attributes, given for a data object are not determined by the associated ontology concept, but are only suggested by the semantics of this concept. With this, the context information is represented on an extensional level, and may be seen as additional attributes that are accessible by the user or application program directly, i.e., without requiring additional interpretation steps.

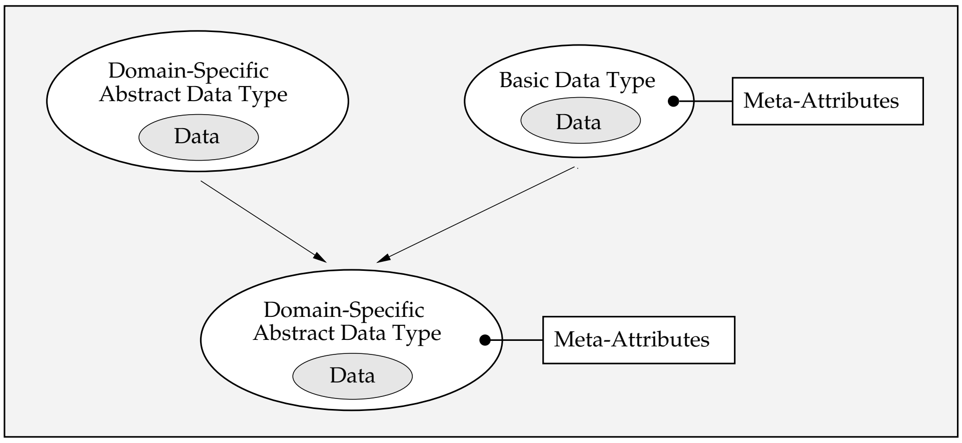

Domain-Specific Abstract Data Type
Data
Basic Data Type
Data
Meta-Attributes
Domain-Specific Abstract Data Type
Data
Meta-Attributes

Figure 1: Representation Alternatives for Semantic Context Information

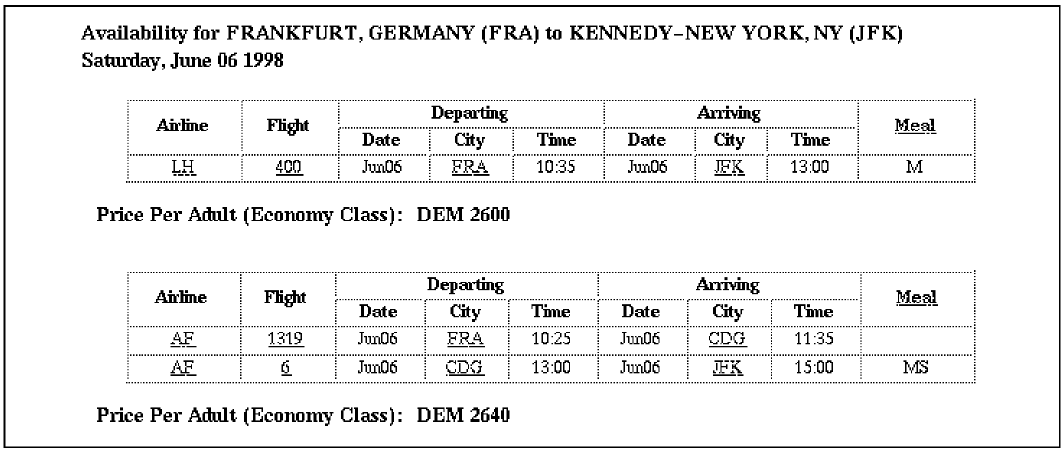
The association of ontology concepts corresponds to the classification of the available data according to the model underlying this ontology. This enables an explicit description of the relationships between a given data object and the real world phenomenon it describes, and thus provides the basis for the detection of semantic similarities of data objects from different sources. The disadvantages of a purely type-based context representation (e.g., the excessive growth of the underlying type system, and the inflexible association of context aspects) are avoided.

The data model presented in the this paper provides a semantic description model which allows an explicit description of the semantic assumptions underlying a given data object on the basis of the representation approach discussed above.

1.3 Scenario

In the business-to-intermediary-to-consumer scenario we are dealing with, to find the lowest possible airfare available, or to put together a trip involving more than one airline, travel agencies must access and share data from different online reservation systems. However, different airlines and consolidators, like American Airlines/SABRE or Teleport Travel, represent their information differently which makes the use of data from different providers difficult and time consuming. We were approached by a major travel agency that needed help in extracting data from the web and preparing it for further processing. In the scenario we were given, the consumer of the information accesses a heterogeneous but relative stable combination of reservation systems and web sites.

Figures 2 and 3 show flight information from two different online reservation systems as they are available on the Web. The available data is provided as semistructured data in the form of HTML pages. Because there is no obligatory data schema associated with this data, the structure underlying it is irregular, e.g., some offers are composed of multiple flight segments, and information concerning certain aspects is not given for all flights or is represented differently, as is the case with information concerning meal services in reservation system A.



Availability for FRANKFURT, GERMANY (FRA) to KENNEDY–NEW YORK, NY (JFK)
Saturday, June 06 1998

Airline	Flight	Departing			Arriving			Meal
		Date	City	Time	Date	City	Time	
LH	400	Jun06	FRA	10:35	Jun06	JFK	13:00	M

Price Per Adult (Economy Class): DEM 2600

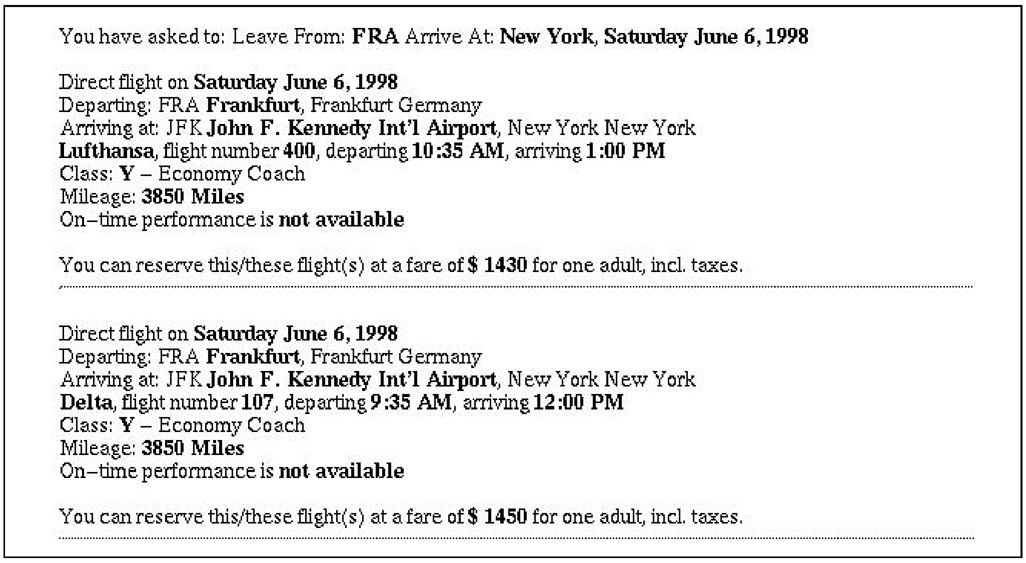
Airline	Flight	Departing			Arriving			Meal
		Date	City	Time	Date	City	Time	
AF	1319	Jun06	FRA	10:25	Jun06	CDG	11:35	
AF	6	Jun06	CDG	13:00	Jun06	JFK	15:00	MS

Price Per Adult (Economy Class): DEM 2640

Figure 2: Reservation System A

Although the available data obviously has some internal structure, this structural information is not accessible as a separately specified schema, but is given in the form of HTML tags, and thus is part of the data itself. Therefore, the underlying structural information has to be extracted first to become useful for automated processing.

In contrast to conventional database schemas, the structure underlying the data is much more flexible and may be modified as easily as the data itself. Accordingly, frequent modifications of the chosen representation have to be expected and taken into consideration by using a very flexible description model when mapping these data to a common representation.



You have asked to: Leave From: **FRA** Arrive At: **New York, Saturday June 6, 1998**

Direct flight on **Saturday June 6, 1998**
Departing: FRA **Frankfurt**, Frankfurt Germany
Arriving at: JFK **John F. Kennedy Int'l Airport**, New York New York
Lufthansa, flight number **400**, departing **10:35 AM**, arriving **1:00 PM**
Class: **Y** – Economy Coach
Mileage: **3850 Miles**
On–time performance is **not available**

You can reserve this/these flight(s) at a fare of **$ 1430** for one adult, incl. taxes.

Direct flight on **Saturday June 6, 1998**
Departing: FRA **Frankfurt**, Frankfurt Germany
Arriving at: JFK **John F. Kennedy Int'l Airport**, New York New York
Delta, flight number **107**, departing **9:35 AM**, arriving **12:00 PM**
Class: **Y** – Economy Coach
Mileage: **3850 Miles**
On–time performance is **not available**

You can reserve this/these flight(s) at a fare of **$ 1450** for one adult, incl. taxes.

Figure 3: Reservation System B

In addition, the data sources we are looking at describe equivalent information differently. They provide different aspects of the flights, and represent the same real world aspects using different structural constructs or semantic concepts. For example, information about the flight distance is recorded in source B only. Additionally, both reservation systems identify airlines with different coding conventions. The detection and resolution of these semantic heterogeneities obviously requires knowledge about the exact semantics underlying the represented data.

The rest of the paper is organized as follows: In the next section we present a model that enables the explicit description of the structure and semantics of semistructured data by using metadata in a manner similar to that discussed in [SSR94, GMS94, KS96]. In Section 3 we show how data from semistructured, heterogeneous data sources can be represented and integrated in an easy and natural way using our model. For this, we use the example of online reservation systems introduced above. Section 4 gives a short comparison of our model with related research efforts. Finally, Section 5 provides conclusions.

2 MIX - Metadata based Integration model for data X-change

In this section we introduce a model for the representation of data together with an explicit description of their underlying interpretation context. This model, which we call ***M**etadata based **I**ntegration model for data **X**-change*, or MIX for short, can be seen as a self-describing data model in the sense of [MR90, LMR90]. This is because information about the structure and semantics of the data is not provided as a separately specified data schema, but is given as part of the available data itself. Thus, MIX allows a flexible association of context information in the form of metadata on an extensional level, and is well suited for the representation of semistructured data.

By using the MIX model we may represent structured and semistructured data in a uniform way, that is, we may represent them on a common interpretation basis. This supports the automatic processing of the available data. In addition, it allows the detection and (at least partially) resolution of heterogeneities between different data sources, and favors a semantically meaningful data exchange because the underlying interpretation contexts are made explicit.

The model is based on the concept of a *semantic object*. A semantic object represents a data item together with its underlying *semantic context*. This semantic context consists of a flexible set of meta-attributes (also represented as semantic objects) that explicitly describe the implicit assumptions about the meaning of the data item. Preferably, this metadata has to be provided by the source when mapping the data to semantic objects, or, in its absence, may be added by the consumer of the data for future automatic processing. However, because we cannot explicitly describe all modeling assumptions, the semantic contexts always has to be understood as a partial representation.

In addition, each semantic object has a *concept label* associated with it that specifies the relationship between the object and the real world aspects it describes. To support the interpretation of the available data and metadata, these labels have to be taken from a commonly known vocabulary, or ontology, that represents an agreement about the exact meaning of the given concepts. In this way, the concept labels, as well as the semantic context of a semantic object explicitly describe the supposed meaning of the data. Our somewhat simplified notion of context allows us to keep the model simple, and supports an economic representation.

The following sections introduce the fundamental concepts of the MIX model. In Section 2.1 we discuss the role of domain-specific ontologies as a common interpretation basis for data and metadata. We distinguish between simple and complex semantic objects. The concept of simple semantic objects, which are used for the representation of atomic data values is introduced in Section 2.2. Section 2.3 deals with the idea of semantic conversion and shows how simple semantic objects can be converted among different semantic contexts. Based on these concepts, Section 2.4 shows how conversion functions can be used for the comparison of semantic objects represented with regard to different contexts. Section 2.5 defines arithmetic operations on simple semantic objects.

In Section 2.6 we introduce complex semantic objects for the representation of complex data objects. The concepts of semantic conversion and semantic comparability are generalized for the class of complex semantic objects in Sections 2.7 and 2.8, respectively. Finally, Section 2.9 defines the concept of semantic identity which provides a prerequisite for the integration of semantic objects that represent the same real world phenomenon.

2.1 Ontologies as a Common Interpretation Basis

To ensure a semantically correct interpretation of the available metadata we use domain-specific ontologies [MMS98, MKIS98]. An ontology provides an agreement about a shared conceptualization of a given application domain [Grub95, Guar96]. The concepts specified in the ontology provide a common vocabulary for which no further negotiation concerning their meaning is necessary. In addition, the ontology provides information about the representation of the data described on the basis of the model. In this way, the ontology can serve as a common basis for the interpretation of context information in the form of metadata. We assume a tight coupling of different ontologies by assembling them following the module-based approach described in [FFR96]. A more loose approach of correlating different ontologies has been discussed in [MKIS98].

In an ideal situation, all instances that make use of data and metadata from a given domain should adhere to the corresponding ontology. In an imperfect real world, we must allow ontologies on consumer side that are tailored to specific needs and provide for extensibility of the model. Ontologies should follow existing description standards (like the Unicorn standard [UNI94] for travel information, or the well known two letter airline code) as much as possible. Aspects for which no such standards exist require new ontology concepts to be specified by the corresponding data source, or by the consumer of the data. Depending on the application domain at hand, this can be done following a top-down approach as proposed in [FFR96], or a bottom-up approach as introduced in [VM98]. By providing the means for adding metadata and extending the ontology on the receiver side, we believe we can claim a reasonable combination of rigor and flexibility.

However, such a conceptualization of a given application domain has to be understood as an abstraction which is always incomplete and which incorporates significant simplifications because of efficiency reasons or because of the specific needs of the institution providing the data. Unimportant aspects of the application domain being conceptualized are ignored. Therefore, the set of semantic properties associated with a given semantic object via the corresponding ontology concepts always has to be seen as a subset of the semantic properties that correspond to the real world phenomenon it describes.

Definition 1 (Ontology)
In the MIX model, we simplify by understanding an ***ontology*** $\mathbb{O}$ *as a vocabulary for the notation and representation of a given application domain, which is given in the form of a finite set of concepts:*

$$\mathbb{O} := \{C_1, \ldots, C_n\} \; , \; n \in \mathbb{N} \; .$$

Each ontology concept C_i has a physical representation type ***RepType(C_i)*** *associated with it, which is either atomic (e.g., string, integer, real, etc.), or "complex", in which case the exact physical representation is not determined by the concept. The domain of the representation type* ***Dom(RepType(C_i))*** *specifies the possible values for the representation of data corresponding to C_i.*

There is a significant difference between the terms *concept* and *physical representation type*, as they are used here. An ontology concept may be understood as an abstraction of a (homogeneous) set of real world phenomena, and thus describes the correspondence between data of a given concept and the and the respective real world domain. In contrast, the physical representation type determines the representation of a data value of a certain concept in the system. An example for the concept of an ontology as it is used here is given by the AirTravel ontology in the appendix.

2.2 Simple Semantic Objects

In this section we introduce the concept of a simple semantic object. A simple semantic object may be understood as an atomic data item, like a simple number value or a character string, with enough additional information attached so that its semantics can be figured out by a human reader or automatically. The meaning of a simple semantic object is determined by a concept label which specifies the real world aspect the object refers to, and an explicit description of its semantic context. This semantic context consists of a flexible set of meta-attributes that explicitly describe implicit modeling assumptions concerning the meaning of the represented data value.

Definition 2 (Simple Semantic Objects)
Given $\mathbb{O}$ as the ontology underlying the representation. The ***simple semantic object*** *SemObj representing the atomic data value v is a 3-tuple of the form:*

$$SemObj \;:=\; <C, \acute{v}, \mathbb{S}> \,,$$

where $C \in \mathbb{O}$ denotes the ontology concept to which the semantic object adheres, and $\acute{v} \in Dom(RepType(C))$ is the physical representation of v according to the physical representation type of C[1]*. $\mathbb{S}$ specifies the semantic context associated with SemObj.*

The ***semantic context*** *$\mathbb{S}$ of a simple semantic object explicitly specifies the interpretation context of a data value, and is defined as:*

$$\mathbb{S} \;:=\; \{<C_1,\, v_1,\, \mathbb{S}_1>, \ldots, <C_k,\, v_k,\, \mathbb{S}_k>\}\,,\; k \in \mathbb{N}_0\,,$$

where $<C_i,\, v_i,\, \mathbb{S}_i>$, $1 \le i \le k$, are the semantic objects that describe different aspects of the semantic object. In turn, they may provide additional context information that further describes the respective semantic aspect specified.

[1]Different data sources may provide different physical representations of the data items they offer. To deal with these physical heterogeneities in a clean manner in the definition above we have to distinguish between v, represented with regard to the local type system of the respective source, and $\acute{v}$, represented according to the physical representation type specified in the common ontology.

In the rest of this paper, $\mathbb{SSO}_{\mathbb{O}}$ denotes the (infinite) set of simple semantic objects that are definable based on ontology $\mathbb{O}$. By $\mathbb{P}_{\mathbb{O}}$ we denote the power set of all possible semantic contexts that are definable based on ontology $\mathbb{O}$.

The following definition introduces the concept of a semantic type of a simple semantic object. Each simple semantic object corresponds to a semantic type that specifies the ontology concept to which the object adheres, as well as its corresponding physical representation. However, the information given as the semantic context of a simple semantic object is not determined by its semantic type. This allows data items from different sources that refer to the same real world aspects to be subsumed under the same semantic type, even if they differ in their underlying contextual assumptions. Otherwise we would have to introduce a new semantic type for each set of semantic properties of a semantic object which would lead to an excessive growth of the underlying type system.

Accordingly, the domain of a semantic type encompasses the set of all semantic objects that semantically correspond to the specified ontology concept and are represented according to the associated physical representation type.

Definition 3 (Semantic Type of a Simple Semantic Object)
Given $\mathbb{O}$ as the ontology underlying the representation.The ***semantic type*** *$SemType$ of a simple semantic object $<C,\ v,\ \mathbb{S}>$ is defined as the tuple:*

$$SemType(<C,\ v,\ \mathbb{S}>)\ :=\ <C,\ RepType(C)>$$

with domain:

$$Dom(<C,\ RepType(C)>)\ :=\ \{<C,\ v,\ \mathbb{S}>\ |\ C \in \mathbb{O},\ v \in Dom(RepType(C)),\ \mathbb{S} \in \mathbb{P}_{\mathbb{O}}\}\ .$$

Following this, the semantic type of context $\mathbb{S} = \{<C_1,\ v_1,\ \mathbb{S}_1>, \ldots, <C_k,\ v_k,\ \mathbb{S}_k>\}$ is defined as:

$$\begin{aligned} SemType(\mathbb{S})\ :=\ &\{\ SemType(<C_1,\ v_1,\ \mathbb{S}_1>), \ldots, SemType(<C_k,\ v_k,\ \mathbb{S}_k>)\ \}\ = \\ &\{\ <C_1,\ RepType(C_1)>, \ldots, <C_k,\ RepType(C_k)>\ \} \end{aligned}$$

with domain:

$$\begin{aligned} Dom(SemType(\mathbb{S})) = Dom(\{\ <C_1,\ RepType(C_1)>, \ldots, <C_k,\ RepType(C_k)>\ \})\ := \\ Dom(<C_1,\ RepType(C_1)>)\ \times\ \ldots\ \times\ Dom(<C_k,\ RepType(C_k)>)\ . \end{aligned}$$

The elements $<C_i,\ RepType(C_i)>$ of $SemType(\mathbb{S})$ are called ***semantic aspects*** *of the semantic type of $\mathbb{S}$.*

A semantic aspect corresponds to a category of meta-information that describes a certain property of a semantic object. These properties include fundamental assumptions about the meaning and possible use of a given data object. The ontology concept associated with a given semantic object implies which semantic aspects are meaningful. In general, however, the particular semantic aspects described in a semantic context are only a subset of all meaningful semantic aspects of an ontology concept. Thus, dependent on the available context information, two semantic objects of the same concept may have different semantic aspects associated.

Example:
The following example illustrates the representation of a simple data value by a corresponding simple semantic object. The AirTravel ontology, as given in in the appendix, describes the meaning and representation of the concepts *Distance* (represented as a *real* value), *Unit* as the underlying unit of measure (represented as a string value, e.g., "mile", "km", "m", etc.), and *Scale* as the scale factor of a numerical value (also represented as a *real* value). Based on these concepts, the flight distance between Frankfurt/Main and New York as given in Figure 3 may be represented as:

$$dist \;=\; <Distance,\; 3850,\; \{<Unit, \text{“}mile\text{”}>^{2},\; <Scale, 1>\}> \;.$$

The corresponding semantic type is given by:

$$SemType(dist) \;=\; <Distance,\; real>$$

with domain:

$$Dom(SemType(dist)) = \{<Distance,\; v,\; \$> \mid Distance \in AirTravel,\; v \in Dom(real),\; \$ \in \mathbb{P}_{AirTravel}\}.$$

2.2.1 Contentderived and Contentdescribing Semantic Aspects

Concerning the concept of a semantic aspect as introduced in the last subsection, we may distinguish between aspects that can be derived from the information represented by the given data object directly, which we call contentderived semantic aspects, and semantic aspects that provide additional information concerning the meaning of the data that can not be derived from the data content. These aspects are called context describing semantic aspects.

Definition 4 (Contentderived and Contentdescribing Semantic Aspects)
Given C_1 denoting the semantic aspect $<C_1, RepType(C_1)>$. $<C_1, RepType(C_1)>$ is called a ***contentderived semantic aspect*** *with regard to concept C, if for all semantic objects of semantic type $<C, RepType(C)>$ we can define a mapping function F_{C,C_1} of the following signature:*

$$F_{C,C_1} \;:\; Dom(RepType(C)) \longrightarrow Dom(RepType(C_1)) \;,$$

with:

$$F_{C,C_1}(v) \;=\; v_1 \;.$$

Otherwise, we call $<C_1, RepType(C_1)>$ a ***contentdescriptive semantic aspect*** *with regard to concept C.*

This means, C_1 denotes a contentderived semantic aspect, if we can find a corresponding function that maps each semantic object of semantic object type $<C, RepType(C)>$ to the according value v_1 given only the meaning of concept C and the information represented by v.

[2]If a semantic object does not provide additional context information we use the simpler notation $SemObj = <C,\, v>$ instead of $SemObj = <C,\, v,\, \emptyset>$.

2.3 Semantic Conversion

The association of context information with a given data value serves as an explicit specification of the implicit assumptions about the meaning of the data. This allows the determination of semantic objects that represent the same information, even if they are represented differently, i.e., with relation to different contexts. For example, intuitively:

$$< \mathit{Distance},\ 3850,\ \{<\mathit{Unit}, \text{“}\mathit{mile}\text{”}>, <\mathit{Scale}, 1>\} > \quad \text{and}$$
$$< \mathit{Distance},\ 3.85,\ \{<\mathit{Unit}, \text{“}\mathit{mile}\text{”}>, <\mathit{Scale}, 1000>\} >$$

are semantically equivalent; that is, they represent the same information. They may be classified semantically equivalent because we can specify a conversion according to the mapping rule *“v (scale x) = v $\frac{x}{y}$ (scale y)”* by which one representation can be transformed into the other. Conversion functions of this kind provide the prerequisite for a semantically meaningful comparison of semantic objects. In addition, conversion functions are used when semantic objects have to be exchanged between institutions that expect different semantic contexts of the data, or when semantic objects from different sources have to be integrated. In this section we look at semantic conversion of simple semantic objects. In Section 2.7 the statements given here are generalized for the semantic conversion of complex semantic objects.

2.3.1 Elementary Conversion Functions

The following definitions introduce the concept of elementary conversion functions. Elementary conversion functions build the foundation of composite conversion functions which are introduced in Section 2.3.3.

Definition 5 (Elementary Conversion Functions)
Given the semantic aspect $< C_1,\ \mathit{RepType}(C_1) >$ and a simple semantic object $\mathit{SemObj} = < C,\ v,\ \mathbb{S} >$ with $< C_1, \mathit{RepType}(C_1) > \in \mathit{SemType}(\mathbb{S})$. An ***elementary conversion function*** *ϕ_{C_1} concerning the semantic aspect C_1 is defined as a function of the following signature:*

$$\phi_{C_1} \ :\ \mathit{Dom}(\{<C_1, \mathit{RepType}(C_1)>\}) \times \mathbb{SSO}_{\Phi} \longrightarrow \mathbb{SSO}_{\Phi}\ ,$$

that converts the data value $v \in \mathit{Dom}(\mathit{RepType}(C))$ represented with regard to context $\mathbb{S} = \{< C_1,\ v_1 >, \dots\}$ to the data value $\acute{v} \in \mathit{Dom}(\mathit{RepType}(C))$ represented on the basis of context $\acute{\mathbb{S}} = \{< C_1,\ \acute{v}_1 >, \dots\}$, with $v_1,\ \acute{v}_1 \in \mathit{Dom}(\mathit{RepType}(C_1))$, i.e.:

$$\phi_{C_1}(\{<C_1,\ \acute{v}_1>\},\ < C,\ v,\ \mathbb{S} >) \ = \ <C,\ \acute{v},\ \acute{\mathbb{S}}> \ .$$

For any semantic aspect $<\tilde{C}_1, \mathit{RepType}(\tilde{C}_1)> \notin \mathit{SemType}(\mathbb{S})$ not represented in the semantic context of the semantic object to be considered the semantic object remains unaffected by the corresponding elementary conversion function $\phi_{\tilde{C}_1}$, i.e.:

$$\phi_{\tilde{C}_1}(\{<\tilde{C}_1,\ \tilde{v}_1>\},\ <C,\ v,\ \mathbb{S}>) = < C,\ v, \mathbb{S} > \ .$$

In this case $< \tilde{C}_1, \mathit{RepType}(\tilde{C}_1) >$ is called a ***not significant*** *semantic aspect of the semantic object in consideration.*

With this, the elementary conversion function ϕ_{C_1} may be understood as a mapping function that transforms a simple semantic object, represented with regard to a certain value v_1 of the significant aspect C_1, into another semantically corresponding[2] representation with regard to $\acute{v}_1$ for C_1. Other semantic aspects represented in the context of the semantic object to be converted are not considered for the conversion.

Example:
Concerning the example from Section 2.2, if ϕ_{Unit} defines a conversion function for the semantic aspect denoted by *Unit*, we get:

$$\phi_{Unit}(\{<Unit,\ \text{``}km\text{''}>\},\ <Distance,\ 3850,\ \{<Unit,\ \text{``}mile\text{''}>, <Scale, 1>\}>) = \\ <Distance,\ 6194.65,\ \{<Unit,\ \text{``}km\text{''}>, <Scale, 1>\}>,$$

with *"1 mile = 1.609 km"* being the underlying mapping rule.

In general, there can be more than one semantically meaningful conversion function defined for a given semantic aspect. For example, the definition of a conversion function for the mapping among different currency units depends on the specific exchange rates that have to be used by an application. Conversion functions can be specified in the underlying ontology, or may be stored in an application-specific conversion library. Which conversion functions to be used may depend on the application using the integrated data, and has to be specified by the respective application.

2.3.2 Properties of Elementary Conversion Functions

Conversion functions may provide properties that, as will be shown in Section 2.4, are important for the application of comparison operators on semantic objects. In particular, a conversion function may be total, lossless with regard to the information represented by the according semantic object as well as orderpreserving concerning a given order relation [SSR94].

Definition 6 (Total and Partial Conversion Functions)
Given a simple semantic object SemObj $=<C, v, \$>$ *with* $\$ = \{<C_1, v_1>, \ldots, <C_k, v_k>\}$*, and given* $<C_1,\ RepType(C_1)> \in SemType(\$)$ *a significant semantic aspect concerning SemObj. A* ***total (elementary) conversion function*** ϕ_{C_1} *represents a mapping function of the form:*

$$\phi_{C_1}(\{<C_1,\ \acute{v}_1>\}, <C,\ v, \{<C_1,\ v_1>, \ldots, <C_k,\ v_k>\}>) = <C,\ \acute{v}, \{<C_1,\ \acute{v}_1>, \ldots, <C_k,\ v_k>\}>,$$

that is defined for all arguments $v_1, \acute{v}_1 \in Dom(RepType(C_1))$ *from the domain of the concept underlying aspect* $<C_1,\ RepType(C_1)>$ *as well as for all data values* $v \in Dom(RepType(C))$*. Otherwise* ϕ_{C_1} *is called a* ***partial (elementary) conversion function.***

Examples:
The elementary conversion function ϕ_{Unit} introduced in Section 2.3.1 provides an example of a total conversion function, because for every transformation of a given physical quantity among different units of measure, we can give a general and semantically meaningful mapping rule.

[2]The semantically corresponding representation is imposed by the semantics of the application domain to be considered, and is specified by the respective mapping rule.

An elementary conversion function concerning the semantic aspect of the *granularity* of a geographic location, e.g., *Boston*, *Paris*, *Alaska*, etc., gives an example of a partial, i.e., non-total, conversion function [SSR94]. The set of values specifying different locations can be seen as hierarchically ordered similar to the ontology specified in [KS96]. Thus, *Paris* may be understood as being more specific than the corresponding location *France*, for example. Hence, for the explicit specification of the granularity of a location we introduce the semantic aspect *Granularity* with $Dom(Granularity) = \{\text{“}city\text{”}, \text{“}country\text{”}, \ldots\}$. A conversion function $\phi_{Granularity}$ may provide a conversion of a location value between different granularities, e.g.:

$$\begin{aligned}\phi_{Granularity}\,(\{<Granularity,\ \text{“}country\text{”}>\},\ <Location,\ \text{“}Paris\text{”},\ \{<Granularity,\ \text{“}city\text{”}>\}>) &=\\ <Location,\ \text{“}France\text{”},\ \{<Granularity,\ \text{“}country\text{”}>\}>\ ,\end{aligned}$$

However, the conversion function $\phi_{Granularity}$ is not total according to Definition 6, because in general we may not be able to define a semantically meaningful mapping from a coarse to a fine granularity.

Definition 7 (Lossless and Lossy Conversion Functions)
Given a simple semantic object $SemObj = <C,\ v,\ \$>$ *with* $\$ = \{<C_1,\ v_1>, \ldots\}$, *and given* $<C_1,\ RepType(C_1)> \in SemType(\$)$ *a significant semantic aspect concerning* $SemObj$. *A conversion function* ϕ_{C_1} *is* ***lossless****, iff for all* $v \in Dom(RepType(C))$ *and for all* $v_i \in Dom(RepType(C_1)), 1 \leq i \leq m$*, for which* ϕ_{C_1} *is well defined we have:*

$$\begin{aligned}&\phi_{C_1}(\{<C_1,\ v_m>\},\ <C,\ v, \{<C_1,\ v_1>, \ldots\}>) \ =\\ &\phi_{C_1}(\{<C_1,\ v_m>\},\ \phi_{C_1}(\{<C_1,\ v_{m-1}>\}, \ldots\ \phi_{C_1}(\{<C_1,\ v_2>\},\ <C,\ v, \{<C_1,\ v_1>, \ldots\}>)\ldots))\ .\end{aligned}$$

Otherwise we call ϕ_{C_1} *a* ***lossy (elementary) conversion function****.*

That means, a lossless conversion function ϕ_{C_1} defines a mapping function whose result does not depend on whether a semantic object will be transformed between two values of the according semantic aspect directly, or in a sequence of intermediate conversion steps. In the special case of $v_1 = v_m$, with ϕ_{C_1} being a lossless conversion function, each data value $v \in Dom(RepType(C))$ may be converted back to its original context without any information loss. Thus, a lossless conversion function is a mapping function for which a corresponding inverse function can be specified.

Examples:
The elementary conversion function ϕ_{Unit} provides an example of a lossless conversion function, because starting from a given length value, any sequence of transformation steps among different units of measure leads to the same conversion result as transforming it to the unit corresponding to the final conversion step directly.

Examples of lossy conversion functions include functions for transformations between different document formats, between different discrete representations of continuous data or lossy compression functions like JPEG and MPEG.

Definition 8 (Orderpreserving Conversion Functions)
Given a simple semantic object $SemObj = <C,\ v,\ \$>$ *with* $\$ = \{<C_1,\ v_1>, \ldots, <C_k,\ v_k>\}$, *and given* $<C_1,\ RepType(C_1)> \in SemType(\$)$ *a significant semantic aspect concerning* $SemObj$. *An elementary conversion function* ϕ_{C_1} *for the semantic aspect denoted by* C_1 *is*

*called **orderpreserving** with respect to order relation Θ, iff for all $v_1, \acute{v}_1 \in Dom(RepType(C_1))$ and for all $v_x, v_y, \acute{v}_x, \acute{v}_y \in Dom(RepType(C))$ with $v_x \; \Theta \; v_y$ for which ϕ_{C_1} is well defined we have:*

$$\begin{aligned}
<C,\, \acute{v}_x, \{<C_1,\, \acute{v}_1>, \ldots, <C_k,\, v_k>\}> &= \phi_{C_1}(\{<C_1,\, \acute{v}_1>\}, <C,\, v_x, \{<P_1,\, p_1>, \ldots, <P_k,\, p_k>\}>)\,, \\
<C,\, \acute{v}_y, \{<C_1,\, \acute{v}_1>, \ldots, <C_k,\, v_k>\}> &= \phi_{C_1}(\{<C_1,\, \acute{v}_1>\}, <C,\, v_y, \{<C_1,\, v_1>, \ldots, <C_k,\, v_k>\}>) \\
\textit{with}: \quad & \acute{v}_x \;\; \Theta \;\; \acute{v}_y \;.
\end{aligned}$$

This means, conversion function ϕ_{C_1} is orderpreserving with regard to order relation Θ, if the corresponding order relation for the values v_x, v_y is well defined and will not be violated by the application of ϕ_{C_1}.

Examples:
The elementary conversion function ϕ_{Unit} also provides an example for an orderpreserving conversion function with regard to "<" and ">", because the according relationships between two data values represented with regard to the same unit of measure are independent of the actual unit.

In [SSR94] Sciore, Siegel, and Rosenthal give a mapping function concerning the semantic aspect *CodeType* specifying the underlying character code, e.g., ASCII or EBCDIC, of a given integer value as an example for an elementary conversion function that is not orderpreserving with regard to "<" and ">". The following semantic objects:

$$\begin{aligned}
&<CharCode,\ 48,\ \ \{<CodeType,\ \text{“}ASCII\text{”}>\}> \text{ and} \\
&<CharCode,\ 240,\ \{<CodeType,\ \text{“}EBCDIC\text{”}>\}>
\end{aligned}$$

both provide a representation of character '0', and the semantic objects:

$$\begin{aligned}
&<CharCode,\ 65,\ \ \{<CodeType,\ \text{“}ASCII\text{”}>\}> \text{ and} \\
&<CharCode,\ 193,\ \{<CodeType,\ \text{“}EBCDIC\text{”}>\}>
\end{aligned}$$

are representants of character 'A'.

Thus, the corresponding conversion function $\phi_{CodeType}$ is not orderpreserving with regard to "<" and ">", because by using ASCII as the underlying character code we have '0' represented by a smaller integer than character 'A', whereas for using code type EBCDIC the inverse relationship holds.

2.3.3 Semantic Conversion of Multivalued and Multileveled Semantic Contexts

In this section we will generalize the concepts introduced for the semantic conversion of single-valued semantic contexts according to the conversion concerning multivalued and multileveled semantic contexts.

2.3.3.1 Conversion Concerning Multivalued Semantic Contexts

In Section 2.3.1 we introduced the concept of elementary conversion functions, for which the target context describes only a single semantic aspect of the semantic object to be considered.

This restriction will be removed with regard to the general case of conversion functions concerning multivalued semantic contexts, that is, semantic contexts that provide more than one semantic aspect.

Definition 9 (Composite Conversion Functions)
Given a simple semantic object SemObj $= <C,\ v,\ \$>$ *with* $\$ = \{<C_1,\ v_1>, \ldots, <C_n,\ v_n>\}$, *and given* $\acute{\$} = \{<C_1,\ \acute{v}_1>, \ldots, <C_k,\ \acute{v}_k>\}$. *A* ***composite conversion function*** ϕ *concerning the semantic type* $SemType(\acute{\$})$ *of the multivalued semantic context* $\acute{\$}$*, is a function of the following signature:*

$$\phi \ :\ Dom(SemType(\acute{\$})) \ \times\ \$\$\mathbb{O}_{\mathbb{O}} \ \longrightarrow\ \$\$\mathbb{O}_{\mathbb{O}}\ ,$$

that maps SemObj to the simple semantic object SemObj′ which corresponds to a conversion of SemObj with regard to all significant semantic aspects $<C_i,\ RepType(C_i)> \in SemType(\acute{\$})$. *With* $v, \acute{v} \in Dom(RepType(C)), \acute{\$}$ *as specified above, and* $\tilde{\$} = \{ <C_1, \acute{v}_1>, \ldots, <C_m, \acute{v}_m> \} \subseteq \acute{\$}$, *being the set of all significant semantic aspects concerning SemObj from* $\acute{\$}$*, we define:*

$$\phi(\ \{<C_1,\ \acute{v}_1>, \ldots, <C_k,\ \acute{v}_k>\},\ \ <C,\ v, \{<C_1,\ v_1>, \ldots, <C_n,\ v_n>\}\ >)\ :=$$

$$\phi(\ \{<C_1,\ \acute{v}_1>, \ldots, <C_m,\ \acute{v}_m>\},\ <C,\ v, \{<C_1,\ v_1>, \ldots, <C_n,\ v_n>\}\ >)\ :=$$

$$<C,\ \acute{v}, \{<C_1, \acute{v}_1>, \ldots, <C_m,\ \acute{v}_m>, <C_{m+1},\ v_{m+1}>, \ldots, <C_n,\ v_n>\}\ >\ .$$

The resulting semantic object SemObj′ may be obtained by applying the corresponding elementary conversion functions ϕ_{C_i} *concerning the semantic aspects* $<\ C_i,\ RepType(C_i)\ > \in SemType(\$)$. *Hence, we define the composite conversion function* ϕ *as:*

$$\phi(\ \{<C_1,\ \acute{v}_1>, \ldots, <C_k,\ \acute{v}_k>\},\ <C,\ v, \{<C_1,\ v_1>, \ldots, <C_n,\ v_n>\}\ >)\ :=$$

$$\phi_{C_k}(\ \{<C_k, \acute{v}_k>\}, \ldots \phi_{C_1}(\ \{<C_1,\ \acute{v}_1>\},\ <C,\ v, \{<C_1,\ v_1>, \ldots, <C_n,\ v_n>\}\ >) \ldots)\ ,$$

or according to Definition 5 with $\tilde{\$}$ *specified as above:*

$$\phi(\ \{<C_1,\ \acute{v}_1>, \ldots, <C_k,\ \acute{v}_k>\},\ <C,\ v, \{<C_1,\ v_1>, \ldots, <C_n,\ v_n>\}\ >)\ :=$$

$$\phi(\ \{<C_1,\ \acute{v}_1>, \ldots, <C_m,\ \acute{v}_m>\},\ \ <C,\ v, \{<C_1,\ v_1>, \ldots, <C_n,\ v_n>\}\ >)\ =$$

$$\phi_{C_m}(\ \{<C_m,\ \acute{v}_m>\}, \ldots \phi_{C_1}(\ \{<C_1,\ \acute{v}_1>\},\ <C,\ v, \{<C_1,\ v_1>, \ldots, <C_n,\ v_n>\}\ >) \ldots)\ .$$

Thus, a composite conversion function ϕ of the given form, is a function that maps a simple semantic object, represented concerning context $\$$, to a semantically corresponding representation concerning context $\breve{\$} \in Dom(SemType(\$))$[3]. Semantic aspects of context $\acute{\$}$ that are not specified in $\$$, i.e, that are not significant concerning *SemObj*, are ignored. Thus, elementary conversion functions, as introduced in Definition 5, may be understood as a special case of a composite conversion function with regard to a context $\acute{\$}$ with $|\,\acute{\$}\,| = 1$.

Example:
The following example will illustrate the relationships between elementary and composite conversion functions. Concerning the example from Section 2.2, if ϕ defines a conversion function for the

[3]Formal: $\breve{\$} = \tilde{\$}\ \cup\ \{<C_i,\ v_i> \in \$ \mid <C_i, RepType(C_i)> \notin SemType(\tilde{\$})\}$.

semantic aspects denoted by *Unit* and *Scale*, with context $\acute{\$} = \{< \mathit{Unit}, \text{“}\mathit{km}\text{”} >, < \mathit{Scale}, 10 >\}$ we get:

$$\phi(\{<\mathit{Unit},\ \text{“}\mathit{km}\text{”}>, <\mathit{Scale},\ 10>\},\ <\mathit{Distance},\ 3850, \{<\mathit{Unit},\ \text{“}\mathit{mile}\text{”}>, <\mathit{Scale},\ 1>\}>) =$$

$$\phi_{\mathit{Scale}}(\{<\mathit{Scale},\ 10>\},\ \phi_{\mathit{Unit}}(\{<\mathit{Unit},\ \text{“}\mathit{km}\text{”}>\},\ <\mathit{Distance},\ 3850, \{<\mathit{Unit},\ \text{“}\mathit{mile}\text{”}>, <\mathit{Scale},\ 1>\}>)) =$$

$$\phi_{\mathit{Scale}}(\{<\mathit{Scale},\ 10>\},\ <\mathit{Distance},\ 6194.65, \{<\mathit{Unit},\ \text{“}\mathit{km}\text{”}>, <\mathit{Scale},\ 1>\}>) =$$

$$<\mathit{Distance},\ 61.9465, \{<\mathit{Unit},\ \text{“}\mathit{km}\text{”}>, <\mathit{Scale},\ 10>\}> .$$

Where ϕ is based on the corresponding mapping rules *“1 mile = 1.609 km”* for *Unit* and *“v [scale x] = v $\frac{x}{y}$ [scale y]”* for *Scale*, respectively.

In this example, the execution order of the elementary conversion functions does not matter for the conversion result, i.e., both execution sequences lead to the same object representation. The elementary conversion functions used here are therefore called *commutative*. However, conversion functions are not commutative, generally, i.e., the conversion result may depend on the execution order of the conversion functions used. For example, a conversion function for the conversion concerning different units of measure and a function for data compression are not commutative in general.

Definition 10 (Commutative Conversion Functions)
Given two conversion functions ϕ_1 and ϕ_2. ϕ_1 and ϕ_2 are ***commutative****, iff for all semantic objects SemObj $\in \mathbb{SSO}_{\mathbb{D}}$ and all semantic contexts $\$_1$, $\$_2$ with SemType($\$_1$) = SemType($\$_2$) for which ϕ_1 and ϕ_2 are well defined, we have:*

$$\phi_1(\ \$_1,\ \phi_2(\ \$_2,\ \mathit{SemObj}\ )\ )\ =\ \phi_2(\ \$_2,\ \phi_1(\ \$_1,\ \mathit{SemObj}\ )\ )\ .$$

If the elementary conversion functions underlying a given composite conversion function ϕ are not pairwise commutative we have to define the execution order for them in ϕ. In [SSR94] the introduction of priorities specifying the relative execution order of conversion functions is proposed. Following this approach, commutative conversion functions may be assigned the same priority.

2.3.3.2 Properties of Multivalued Composite Conversion Functions

A generalization of the properties of elementary conversion functions, as described in Section 2.3.2, for the class of conversion functions concerning multivalued semantic contexts follows in a natural way from the definition of composite conversion functions as sequences of elementary conversion functions. In the following definitions we suppose, without loss of generality, that all semantic aspects specified by the semantic context used in a conversion function are significant in the sense of Definition 5. Generally, the properties specified apply to a conversion function concerning $\acute{\$}$, iff they apply to the corresponding conversion function with regard to all significant aspects $\tilde{\$} \subseteq \acute{\$}$ of the semantic object to be considered.

Definition 11 (Total and Partial Composite Conversion Functions)
Given a simple semantic object SemObj $= < C,\ v,\ \$ >$ *and given* ϕ *a composite conversion function concerning the semantic type* $SemType(\acute{\$})$ *of context* $\acute{\$} = \{< C_1,\ \acute{v}_1 >, \ldots,\ < C_k,\ \acute{v}_k >\}$ *with* $SemType(\acute{\$}) \subseteq SemType(\$)$*. That means, all semantic aspects* $< C_i,\ RepType(C_i) > \in SemType(\acute{\$})$ *are significant with regard to SemObj. A **total composite conversion function*** ϕ *is a mapping function of the form:*

$$\phi(\{< C_1,\ \acute{v}_1 >, \ldots, < C_k,\ \acute{v}_k >\},\ < C,\ v, \{< C_1,\ v_1 >, \ldots, < C_n,\ v_n >\} >) \ = \\ < C,\ \acute{v},\ \{< C_1,\ \acute{v}_1 >, \ldots, < C_k,\ \acute{v}_k >, < C_{k+1},\ v_{k+1} >, \ldots, < C_n,\ v_n >\} > \ ,$$

which is defined for all arguments $v_i, \acute{v}_i \in Dom(RepType(C_i))$*,* $1 \leq i \leq k$*, and all* $v \in Dom(RepType(C))$*.*

Following the definition of a composite conversion function as a sequence of elementary conversion functions, as given in Definition 9:

$$\phi(\{< C_1,\ \acute{v}_1 >, \ldots, < C_k,\ \acute{v}_k >\},\ < C,\ v, \{< C_1,\ v_1 >, \ldots, < C_n,\ v_n >\} >) \ =$$

$$\phi_{C_k}(\{< C_k,\ \acute{v}_k >\}, \ldots\ \phi_{C_1}(\{< C_1,\ \acute{v}_1 >\},\ < C,\ v, \{< C_1,\ v_1 >, \ldots, < C_n,\ v_n >\} >) \ldots) \ ,$$

we get that a composite conversion function ϕ is total, iff for all elementary conversion functions $\phi_{C_i}, 1 \leq i \leq k$, it is composed of, we have:

ϕ_{C_i} is defined for all $v_i, \acute{v}_i \in Dom(RepType(C_i))$ and all $SemObj \in$

$$\{\ \phi_{C_{i-1}}(\{< C_{i-1},\ v_{i-1} >\}, \ldots \phi_{C_1}(\{< C_1,\ \acute{v}_1 >\},\ < C,\ v, \{< C_1,\ v_1 >, \ldots, < C_n,\ v_n >\} >) \ldots) \\ \mid v_j, \acute{v}_j \in Dom(RepType(C_j)),\ 1 \leq j < i \leq k,\ v \in Dom(RepType(C))\ \} \ .$$

This means, a composite conversion function ϕ is total, iff each of the underlying elementary conversion functions is well defined for all values of the domain of its corresponding semantic aspect as well as for all semantic objects of the possible result set of the elementary conversion function directly preceding it. In particular, ϕ is total, if all elementary conversion functions it is composed of are total according to Definition 6.

Definition 12 (Lossless and Lossy Composite Conversion Functions)
Given a simple semantic object SemObj $= < C,\ v,\ \$ >$ *and given* ϕ *a composite conversion function concerning the semantic type* $SemType(\acute{\$})$ *of context* $\acute{\$}$ *with* $SemType(\acute{\$}) \subseteq SemType(\$)$*. That means, all semantic aspects* $< C_i,\ RepType(C_i) > \in SemType(\acute{\$})$ *are significant with regard to SemObj.* ϕ *is **lossless** (otherwise **lossy**), if the conversion result is the same, whether SemObj is converted among two semantic contexts directly, or with any sequence of intermediate conversion steps. Formally,* ϕ *is lossless, iff for all* $v \in Dom(RepType(C))$ *and for all semantic contexts* $\$_i \in Dom(SemType(\acute{\$})) \subseteq Dom(SemType(\$))$*,* $1 < i \leq n$*, for which* ϕ *is well defined, we have:*

$$\phi(\ \$_n, < C,\ v,\ \$ >) \ = \ \phi(\ \$_n,\ \phi(\ \$_{n-1}, \ldots\ \phi(\ \$_1, < C,\ v,\ \$ >) \ldots)) \ .$$

According to the definition of a composite conversion function as a sequence of elementary conversion functions, a composite conversion function ϕ is lossless, iff all elementary conversion functions it is composed of are lossless according to Definition 7.

Definition 13 (Orderpreserving Composite Conversion Functions)
Given a simple semantic object $SemObj = < C,\ v,\ \$ >$ and given ϕ a composite conversion function concerning the semantic type $SemType(\acute{\$})$ of context $\acute{\$} = \{<C_1,\ \acute{v}_1>, \ldots, <C_k,\ \acute{v}_k>\}$ with $SemType(\acute{\$}) \subseteq SemType(\$)$. That means, all semantic aspects $< C_i, RepType(C_i) > \in SemType(\acute{\$})$ are significant with regard to $SemObj$.

*ϕ is **orderpreserving** concerning order relation Θ, if the relationship of two data values $v_x, v_y \in Dom(RepType(C))$, represented the same according to the semantic aspects specified in $SemObj(\acute{\$})$, remains unaffected by its application. Formally, ϕ is orderpreserving concerning Θ, iff for all $\acute{\$} \in Dom(SemType(\$))$ and for all $v_x, v_y, \acute{v}_x, \acute{v}_y \in Dom(RepType(C))$ with $v_x \,\Theta\, v_y$, for which ϕ is well defined, we have:*

$$< C,\ \acute{v}_x,\ \{<C_1,\ v_1>, \ldots, <C_k,\ v_k>, \ldots\}> \ = \\ \phi(\ \{<C_1,\ \acute{v}_1>, \ldots, <C_k,\ \acute{v}_k>\},\ < C,\ v_x, \{<C_1,\ \acute{v}_1>, \ldots, <C_k,\ \acute{v}_k>, \ldots\}>\ )\ ,$$

$$< C,\ \acute{v}_y,\ \{<C_1,\ v_1>, \ldots, <C_k,\ v_k>, \ldots\}> \ = \\ \phi(\ \{<C_1,\ \acute{v}_1>, \ldots, <C_k,\ \acute{v}_k>\},\ < C,\ v_y, \{<C_1,\ \acute{v}_1>, \ldots, <C_k,\ \acute{v}_k>, \ldots\}>\ )$$

$$with: \qquad \acute{v}_x \ \Theta\ \acute{v}_y\ .$$

Following from the definition of a composite conversion function ϕ as a sequence of elementary conversion functions ϕ_{C_i}, a composite conversion function is orderpreserving concerning the order relation Θ, if all elementary conversion functions it is composed of are well defined for the respective data values and are orderpreserving concerning Θ in the sense of Definition 8 (prove omitted). However, the inverse conclusion is not true in general, because applying two elementary conversion functions not orderpreserving with regard to Θ in sequence may neutralize their order inversing property.

2.3.3.3 Conversion Concerning Multileveled Semantic Contexts

Similar to multivalued contexts, multileveled contexts, that is, semantic contexts whose semantic aspects provide additional context information, lead to a generalization of elementary conversion functions as introduced in Section 2.3.

Definition 14 (Multileveled Semantic Contexts and Conversion Functions)
*Given a simple semantic object $SemObj = < C,\ v,\ \$ >$ with $\$ = \{< C_1,\ v_1, \{< C_2,\ v_2, \ldots, \{< C_n,\ v_n >\} \ldots >\} >\}$[4] as its associated context. We call $< C_i,\ RepType(C_i) >$, $1 \leq i < n$, **multileveled semantic aspects**, and accordingly $\$$ a **multileveled semantic context**.*

*Given further a multileveled context $\acute{\$} = \{< C_1,\ \acute{v}_1, \{< C_2,\ \acute{v}_2, \ldots, \{< C_k,\ \acute{v}_k >\} \ldots >\} >\}$ that describes the semantic aspect C_1 up to level $k \leq n$. The **multileveled conversion function** $\breve{\phi}$ with signature:*

$$\breve{\phi} \ :\ Dom(SemType(\acute{\$}))\ \times\ \$\$\mathbb{O}_{\mathbb{O}} \ \longrightarrow\ \$\$\mathbb{O}_{\mathbb{O}}$$

provides a function that maps $SemObj$ to a semantic object $SemObj'$, that corresponds to the conversion of $SemObj$ concerning all semantic aspects $< C_i,\ RepType(C_i) >$ of $SemType(\acute{\$})$, $1 \leq i \leq k \in \mathbb{N}$, i.e., with $v, \acute{v} \in Dom(RepType(C))$ we have:

[4] Note that the semantic aspects denoted by C_i, $1 \leq i < n$, provide additional semantic context information.

$$\begin{aligned}&\breve{\phi}(\ \{<C_1,\ \acute{v}_1,\ \{<C_2,\ \acute{v}_2,\ldots,\{<C_k,\ \acute{v}_k>\}\ldots>\}>\},\\ &\qquad <C,\ v,\{<C_1,\ v_1,\{<C_2,\ v_2,\ldots,\{<C_n,\ v_n>\}\ldots>\}>\}>\ )\ =\\ &<C,\ \acute{v},\{<C_1,\ \acute{v}_1,\{<C_2,\ \acute{v}_2,\ldots,\{<C_k,\ \acute{v}_k,\{<C_{k+1},\ v_{k+1},\ldots\{<C_n,\ v_n>\}\ldots>\}>\}\ldots>\}>\}>\ .\end{aligned}$$

In contrast to composite conversion functions, $\breve{\phi}$ can not be specified as a composition of elementary conversion functions, because in the case of a multileveled conversion function concerning semantic aspect $< C_i,\ RepType(C_i) >$ we have to take into consideration all semantic aspects $<C_j,\ RepType(C_j)>$, $1 \leq i < j \leq k$. That means, the conversion result of the conversion concerning aspect $<C_i,\ RepType(P_i)>$ may directly depend on the particular values given for the semantic aspects $< C_j,\ RepType(C_j) >$. In the case of multileveled conversion functions, semantic aspects not specified for a given semantic aspect of $\acute{\$}$ are supposed as equal to those given for the corresponding aspect from $\$$.

2.4 Semantic Comparability

In this section, we introduce the concept of semantic comparability of simple semantic objects. In particular, we discuss how semantic equivalence of different semantic objects has to be defined, and how semantic comparability of semantic objects is determined. The example given in Section 2.3 shows two semantic objects that intuitively appear to be semantically equivalent, i.e., represent the same information:

$$\begin{aligned}&< Distance,\ 3850,\ \{<Unit,\text{“}mile\text{”}>,<Scale,1>\}> \quad \text{and}\\ &< Distance,\ 3.85,\ \{<Unit,\text{“}mile\text{”}>,<Scale,1000>\}> \quad .\end{aligned}$$

They may be classified semantically equivalent, because by applying the corresponding conversion function ϕ_{Unit} they can be converted from one representation into the other. However, consider the two semantic objects below:

$$\begin{aligned}&< Price,\ 1430,\ \{< Currency,\text{“}USD\text{”} >\} > \ \text{and}\\ &< Price,\ 2600,\ \{< Currency,\text{“}DEM\text{”} >\} > ,\end{aligned}$$

and the conversion function $\phi_{Currency}$ that converts monetary quantities according to a given exchange rate. As usual for money exchange, we have to take into consideration the asymmetry of conversion that may exist between buying and selling rates. Supposing $\phi_{Currency}$ converts US dollar to German marks on the basis of the exchange rates *“1 USD = 1.778 DEM”* and *“1 DEM = 0.55 USD”*. Then we get the following results:

$$\begin{aligned}&\phi_{Currency}\,(\{< Currency,\text{“}DEM\text{”} >\},< Price,\ 1430,\{< Currency,\text{“}USD\text{”} >\} >) =\\ &\qquad < Price,\ 2542.54,\{< Currency,\text{“}DEM\text{”} >\} > ,\end{aligned}$$

$$\begin{aligned}&\phi_{Currency}\,(\{< Currency,\text{“}USD\text{”} >\},< Price,\ 2600,\{< Currency,\text{“}DEM\text{”} >\} >) =\\ &\qquad < Price,\ 1430,\{< Currency,\text{“}USD\text{”} >\} > \ .\end{aligned}$$

Because of the asymmetry of the conversion function and the difference that results when converting from *“DEM”* to *“USD”* or vice versa it may be reasonable to classify these objects semantically equivalent with regard to *“USD”* as the underlying currency, but not semantically equivalent if currency *"DEM"* is used. This intuitive judgment concerning the semantic equivalence of the objects shown in this example, leads to the formal definition of semantic comparability as given in the next definition.

Definition 15 (Semantic Comparability of Simple Semantic Objects)
Given two simple semantic objects $SemObj_1 = < C, v_1, \$_1 >$ *and* $SemObj_2 = < C, v_2, \$_2 >$*, and given context* $\$$ *with* $SemType(\$) \subseteq SemType(\$_1) \cap SemType(\$_2)$*. This means, all semantic aspects specified in* $\$$ *are significant with regard to* $SemObj_1$ *and* $SemObj_2$*, respectively. Then for a comparison operator* Θ *well defined on* $Dom(RepType(C))$ *we define:*

$$< C, v_1, \$_1 > \ \Theta_{\$} \ < C, v_2, \$_2 >$$

concerning context $\$$ *and the composite conversion function* ϕ*, iff we get:* $\acute{v}_1 \ \Theta \ \acute{v}_2$

$$\begin{array}{lrcl} with: & < C,\ \acute{v}_1,\ \acute{\$}_1 > & = & \phi(\$,\ < C,\ v_1,\ \$_1 >) \ \text{ and} \\ & < C,\ \acute{v}_2,\ \acute{\$}_2 > & = & \phi(\$,\ < C,\ v_2,\ \$_2 >)\ , \end{array}$$

where $\acute{\$}_i, i \in \{1, 2\}$*, is given as* $\acute{\$}_i = \$ \cup \{< C_j,\ v_j > \in \$_i \ \ | < C_j,\ RepType(C_j) > \notin SemType(\$)\}$.

Following [SSR94], the semantic context $\$$ *used for the comparison is referred to as the* ***target context****, and the conversion function* ϕ *is called* ***reference conversion function****, or* ***reference function*** *for short, of the semantic comparison.*

Thus, the result of the semantic comparison of two simple semantic objects is defined through the conversion of both objects to a common context by using an appropriate conversion function, and the comparison of the data values underlying the converted semantic objects. Generally, the result of a semantic comparison depends on the respective target context as well as on the reference function to be used.

The set of semantic aspects specified in target context may be different from those given by the contexts of the semantic objects to be compared. Semantic aspects specified in those contexts not specified in the target context are ignored for the comparison.

2.4.1 Decidable and Non-Decidable Semantic Aspects

In the last section we supposed that two semantic objects may be compared by converting them to a common target context, and comparing the resulting data values underlying them. However, this algorithm does not lead to semantically meaningful results in every case.

For example, with regard to the semantic aspect *Precision*, which specifies the maximal absolute imprecision of a numerical value, the following simple semantic objects:

$$\begin{array}{lcl} SemObj_1 & = & < Price,\ 3.1,\ \{< Precision,\ 0.1 >, < Currency,\ \text{“}USD\text{”} >\} >\ , \\ SemObj_2 & = & < Price,\ 2.9,\ \{< Precision,\ 0.1 >, < Currency,\ \text{“}USD\text{”} >\} > \end{array}$$

represent two monetary quantities that are known to be, because of their imprecision, in the range 3.0 and 3.2 USD (inclusive), and 2.8 and 3.0 USD (inclusive), respectively. According to Definition 15, we have $SemObj_1 >_{\$} SemObj_2$ with $\$ = \{< Precision, 0.1 >,$

$< Currency, \text{“}USD\text{”} >\}$ as its target context. This does not necessarily correspond to the meaning of the semantic objects considered. Without additional information we cannot decide which relationships between $SemObj_1$ and $SemObj_2$ have to be assumed semantically meaningful. Therefore, in [SSR94] the distinction between *resolvable* and *nonresolvable* semantic properties is made. We provide a similar distinction here.

Definition 16 (Decidable and Non-Decidable Semantic Aspects)
Given the semantic aspect $< \tilde{C}, RepType(\tilde{C}) >$, and two semantic objects represented with regard to context $\$ = \{ <\tilde{C},\ \tilde{v}> \}$:

$$\begin{aligned} SemObj_1 &= <C,\ v_1,\ \$> , \\ SemObj_2 &= <C,\ v_2,\ \$> . \end{aligned}$$

*Given further Θ a comparison operator well defined for $Dom(RepType(C))$. Then we call $<\tilde{C},\ RepType(\tilde{C})>$ a **decidable semantic aspect**, iff for all $v_1, v_2 \in Dom(RepType(C))$ with $v_1\ \Theta\ v_2$, it is semantically meaningful to deduce $SemObj_1\ \Theta_{\$}\ SemObj_2$ with regard to context $\$$. Otherwise, we call $<\tilde{C},\ RepType(\tilde{C})>$ a **nondecidable semantic aspect** concerning comparison operator Θ.*

Intuitively, for a decidable semantic aspect the semantic comparison of two semantic objects can always be deduced from the corresponding comparison of the data values underlying them. In the case of nondecidable semantic aspects, additional information about the meaning of the objects to be compared, e.g., in form of application-specific comparison functions, is necessary to get semantically meaningful comparison results. In the rest of this paper, we suppose that all semantic aspects considered are decidable for the respective comparison operator.

2.4.2 Absolute Semantic Comparability

The evaluation of semantic comparison operators may lead to the observation that in some cases the comparison result only depends on the semantic aspects specified in the target context, and not on the particular values given for them, that is, the comparison result is the same for all semantic contexts of a given semantic type. For example, the semantic objects:

$$\begin{aligned} &< Distance,\ 3850,\ \{<Unit, \text{“}mile\text{”}>, <Scale, 1>\} > \quad \text{and} \\ &< Distance,\ 3.85,\ \{<Unit, \text{“}mile\text{”}>, <Scale, 1000>\} > \quad . \end{aligned}$$

are semantically equivalent with regard to each target context that specifies the semantic aspects *Unit* and *Scale*, i.e., they are semantically equivalent regardless of what particular values are specified for *Unit* and *Scale*. Therefore, in [SSR94] this kind of semantic relationship is called *absolute* concerning the given target context.

Definition 17 (Absolute Semantic Equivalence)
*Given two simple semantic objects $SemObj_1 = < C,\ v_1,\ \$_1 >$ and $SemObj_2 = < C,\ v_2,\ \$_2 >$, and given the semantic context $\$$ with $SemType(\$) \subseteq SemType(\$_1) \cap SemType(\$_2)$. That means, all semantic aspects specified in $\$$ are significant with regard to $SemObj_1$ and $SemObj_2$, respectively. Then $SemObj_1$ and $SemObj_2$ are **absolute semantically equivalent** concerning the semantic type $SemType(\$)$ and the corresponding reference function ϕ, denoted as:*

$$SemObj_1 \;=_{SemType(\$)}\; SemObj_2 \;\;,$$

iff holds: $\quad \forall\ \acute{\$} \in Dom(SemType(\$)) \;:\; <C,\, v_1,\, \$_1> \;=_{\acute{\$}}\; <C,\, v_2,\, \$_2> \;\;.$

This means, two simple semantic objects are absolute semantic equivalent concerning the semantic type *SemType*($) and a corresponding reference function, iff the result of the semantic comparison is independent of the particular values of the semantic aspects specified in any target context of semantic type *SemType*($). A generalization of this definition for the comparison operators "<", ">", "≠", etc., can be given analogously.

In [SSR94] the correlation between the properties of totality, losslessness, and orderpreservation of a reference function and the absolutelyness of a semantic composition is shown. Transfered to the context of the representation model presented here, the following propositions can be determined as well.

Proposition 1
Given two simple semantic objects $SemObj_1 = <C,\, v_1,\, \$_1>$ and $SemObj_2 = <C,\, v_2,\, \$_2>$, and given the semantic context $ of the semantic type $SemType(\$) \subseteq SemType(\$_1) \cap SemType(\$_2)$. That means, all semantic aspects specified in $ are significant with regard to $SemObj_1$ and $SemObj_2$. Then we can show that, if the corresponding conversion function ϕ is total and lossless according to Definition 11 and 12, respectively, each semantic comparison operation $\Theta \in \{``=", ``\neq"\}$ between $SemObj_1$ and $SemObj_2$ is absolute concerning *SemType*($) and reference function ϕ.

In particular, with the statements given in Section 2.3.3, if all elementary conversion functions ϕ_{C_i} specified for the corresponding semantic aspects $< C_i, RepType(C_i) >$ of *SemType*($) are total, and lossless, each semantic comparison operation $\Theta \in \{``=", ``\neq"\}$ between $SemObj_1$ and $SemObj_2$ is absolute with regard to *SemType*($) and reference function ϕ composed from the elementary conversion functions ϕ_{C_i}.

Proof:
The proof for Proposition 1 is outlined with regard to the special case of the equivalence operator. A proof for "≠" follows analogously. Given two simple semantic objects of the same semantic object type with:

$$<C,\, v_1,\, \$_1> \;=_{\$}\; <C,\, v_2,\, \$_2>$$

concerning the target context $ with $SemType(\$) = SemType(\$_1) = SemType(\$_2)$[5]. Given further the total und lossless reference function ϕ with:

$$\begin{aligned} <C,\, \acute{v}_1,\, \acute{\$}_1> &= \phi(\,\$, <C,\, v_1,\, \$_1>\,) \text{ and} \\ <C,\, \acute{v}_2,\, \acute{\$}_2> &= \phi(\,\$, <C,\, v_2,\, \$_2>\,)\;. \end{aligned}$$

Then, following Definition 15, we have: $\qquad \acute{v}_1 \;=\; \acute{v}_2\;.$

[5]Because all semantic aspects that are not significant are irrelevant for the comparison, without loss of generality we assume that all aspects of $ are significant with regard to $<C,\, v_1,\, \$_1>$ and $<C,\, v_2,\, \$_2>$.

According to the totality assumed for the underlying reference function ϕ, it is defined for all semantic contexts $\tilde{\$} \in \mathit{Dom}(\mathit{SemType}(\$))$ as well. With the losslessness of ϕ, with $\acute{v}_1 = \acute{v}_2$, and with the equivalence of the semantic contexts $\acute{\$}_1$ and $\acute{\$}_2$ ($\acute{\$}_1 = \acute{\$}_2 = \$$) we get:

$$\begin{array}{ll}
\phi(\tilde{\$},\ <C,\ v_1, \$_1>) & = \\
\phi(\tilde{\$},\ \phi(\$, <C,\ v_1,\ \$_1>)) & = \\
\phi(\tilde{\$},\ <C,\ \acute{v}_1,\ \acute{\$}_1>) & =_{\$} \\
\phi(\tilde{\$},\ <C,\ \acute{v}_2,\ \acute{\$}_2>) & = \\
\phi(\tilde{\$},\ \phi(\$, <C,\ v_2,\ \$_2>)) & = \\
\phi(\tilde{\$},\ <C,\ v_2,\ \$_2>) & .
\end{array}$$

Thus we have:

$$<C,\ v_1,\ \$_1> \ =_{\tilde{\$}}\ <C,\ v_2,\ \$_2> \ .$$

for any semantic context $\tilde{\$} \in \mathit{Dom}(\mathit{SemType}(\$))$.

With regard to the comparison operators "<" and ">", we have to demand that the underlying conversion function is orderpreserving according Definition 13 as well. This additional restriction ensures that the relative order of the semantic objects to be compared will be preserved by the corresponding conversion.

Proposition 2
Given two simple semantic objects $\mathit{SemObj}_1 = <C,\ v_1,\ \$_1>$ and $\mathit{SemObj}_2 = <C,\ v_2,\ \$_2>$, and given the semantic context $\$$ with $\mathit{SemType}(\$) \subseteq \mathit{SemType}(\$_1) \cap \mathit{SemType}(\$_2)$. That means, all semantic aspects specified in $\$$ are significant with regard to SemObj_1 and SemObj_2. Then we can show that, if the corresponding conversion function ϕ is total, lossless, and orderpreserving, each semantic comparison operation $\Theta \in \{$"<", ">"$\}$ between SemObj_1 and SemObj_2 is absolute concerning $\mathit{SemType}(\$)$ and reference function ϕ.

Proof:
The according proof follows the one given for Proposition 1. The additional restriction concerning the orderpreservation of reference function ϕ ensures that for each operator $\Theta \in \{$"<", ">"$\}$ and for all $\tilde{\$} \in \mathit{Dom}(\mathit{SemType}(\$))$ the following holds:

$$\acute{v}_1\, \Theta\, \acute{v}_2 \quad \Rightarrow \quad \phi(\tilde{\$}, <C,\ \acute{v}_1,\ \acute{\$}_1>) \ \ \Theta_{\tilde{\$}} \ \ \phi(\tilde{\$}, <C,\ \acute{v}_2,\ \acute{\$}_2>)\ ,$$

that is, the relative order with regard to Θ is preserved by the corresponding conversion concerning $\tilde{\$}$ and ϕ.

Similarly to Proposition 1, if all elementary conversion functions ϕ_{C_i} specified for the corresponding semantic aspects $<C_i,\ \mathit{RepType}(C_i)>$ of $\mathit{SemType}(\$)$ are total, lossless, and orderpreserving, then each semantic comparison operator $\Theta \in \{$"<", ">"$\}$ between SemObj_1 and SemObj_2 is absolute with regard to $\mathit{SemType}(\$)$ and reference function ϕ composed from the elementary conversion functions ϕ_{C_i}.

2.5 Arithmetic Operations

The concept of arithmetic operations such as addition and multiplication can be transfered to the MIX representation model as well, presupposed the semantic objects in consideration are represented on the basis of numeric data types. In general, similar to the comparison operators discussed in the last section, arithmetic operations have to be applied with respect to a common semantic context of their operands to obtain semantically meaningful results. Thus, generally the result of an arithmetic operation also depends on the target context and the reference function chosen for the necessary conversion. For example, "adding" the following semantic objects:

$$\begin{array}{l} <\!\mathit{Price},\ 1430,\ \{<\!\mathit{Currency},\ \text{``}\mathit{USD}\text{''}\!>\}\!> \ +_{\$} \\ <\!\mathit{Price},\ 2600,\ \{<\!\mathit{Currency},\ \text{``}\mathit{DEM}\text{''}\!>\}\!> \end{array}$$

with regard to target context $\$_1 = \{<\mathit{Currency},\ \text{``}\mathit{DEM}\text{''}>\}$ and the according reference function ϕ_{Currency} given in Section 2.4 results in:

$$\mathit{SemObj}_1 \ = \ <\!\mathit{Price},\ 5142.54,\ \{<\!\mathit{Currency},\ \text{``}\mathit{DEM}\text{''}\!>\}\!> \ .$$

But using $\$_2 = \{<\mathit{Currency},\ \text{``}\mathit{USD}\text{''}>\}$ as the underlying target context and the same conversion function as above we will obtain the object given below:

$$\mathit{SemObj}_2 \ = \ <\!\mathit{Price},\ 2860,\ \{<\!\mathit{Currency},\ \text{``}\mathit{USD}\text{''}\!>\}\!> \ ,$$

which, according to Definition 15, is not semantically equivalent to SemObj_1 with regard to $\$_1$ and $\$_2$, respectively.

Definition 18 (Arithmetic Operations on Simple Semantic Objects)
Given two simple semantic objects $\mathit{SemObj}_1 = <C,\ v_1,\ \$_1>$ and $\mathit{SemObj}_2 = <C,\ v_2,\ \$_2>$, and given a semantic context $\$$ with $\mathit{SemType}(\$) \subseteq \mathit{SemType}(\$_1) \cap \mathit{SemType}(\$_2)$. That means, all semantic aspects specified in $\$$ are significant with regard to SemObj_1 and SemObj_2. Then for an arithmetic operation $\oplus$ well defined on $\mathit{Dom}(\mathit{RepType}((C))$ we define:

$$<\!C,\ v_1,\ \$_1\!> \ \oplus_{\$} \ <\!C,\ v_2,\ \$_2\!> \ := \ <\!C,\ \acute{v}_1 \oplus \acute{v}_2, \$\!> \ ,$$

$$\mathit{with}: \quad \begin{array}{rcll} <\!C,\ \acute{v}_1,\ \acute{\$}_1\!> & = & \phi\,(\$,\ <\!C,\ v_1,\ \$_1\!>) & \text{and} \\ <\!C,\ \acute{v}_2,\ \acute{\$}_2\!> & = & \phi\,(\$,\ <\!C,\ v_2,\ \$_2\!>) & \end{array}$$

concerning the target context $\$$ and the reference function ϕ.

This means, the result of the arithmetic operation $\oplus_{\$}$ is defined by the conversion of both operands with regard to the given target context and reference function, followed by the application of the corresponding arithmetic operation $\oplus$ defined on the data values $\acute{v}_1, \acute{v}_2$ underlying the converted semantic objects. Thus, generally the result of an arithmetic operation on semantic objects depends on the target context and conversion function chosen.

As is the case for semantic comparison operations, semantic aspects not specified in the target context are not considered. However, in contrast to semantic comparison the context of the semantic object representing the operation result is always the same as the target context used, which may provide only a subset of the semantic aspects specified for the given operands.

2.6 Complex Semantic Objects

In this section we introduce the concept of a complex semantic object. Complex semantic objects can be understood as heterogeneous collections of semantic objects that are grouped under a corresponding ontology concept. Each subobject describes exactly one aspect of the represented real world phenomenon. The attributes given for a complex semantic object, except those necessary for the identification of the object, are not determined by its ontology concept, and thus may vary between different complex semantic objects of the same concept.

Definition 19 (Complex Semantic Objects)
Given $\mathbb{O}$ as the ontology underlying the representation, and given a complex data object o consisting of multiple, possibly heterogeneous data elements. Then we define the corresponding ***complex semantic object*** *CompSemObj that represents o as the tuple:*

$$CompSemObj \ := \ <C,\, \mathbb{A}> \ ,$$

where $C \in \mathbb{O}$ is the ontology concept underlying the semantic object, and $\mathbb{A}$ corresponds to the set of semantic objects associated with CompSemObj that provide a representation of the subobjects of o. The set of attributes given for a complex semantic object:

$$\mathbb{A} \ := \ \{\underline{A}_1, \ldots, \underline{A}_m, A_{m+1}, \ldots, A_{m+n}\} \ = \ \underline{\mathbb{A}} \cup \mathbb{A}_R,\ m \in \mathbb{N},\ n \in \mathbb{N}_0$$

is divided into two distinct subsets $\underline{\mathbb{A}}$ and $\mathbb{A}_R$. Whereat $\underline{\mathbb{A}} := \{\underline{A}_1, \ldots, \underline{A}_m\}$ is the set of attributes that are, similar to a set of key attributes in the relational model, used to identify a complex semantic object of concept C. Subset $\mathbb{A}_R := \{A_{m+1}, \ldots, A_{m+n}\}$ provides the set of additional attributes recorded for CompSemObj according to the information given by o. Again, the attributes represented by $\underline{\mathbb{A}}$ and $\mathbb{A}_R$ are given as semantic objects that may be either simple or complex.

In the following sections by $\mathbb{CSO}_{\mathbb{O}}$ we denote the (infinite) set of all complex semantic objects, and by $\mathbb{SO}_{\mathbb{O}} := \mathbb{SSO}_{\mathbb{O}} \cup \mathbb{CSO}_{\mathbb{O}}$ the set of all semantic objects that are represented on the basis of ontology $\mathbb{O}$.

The ***semantic type*** *SemType of CompSemObj is defined as the tuple:*

$$SemType(CompSemObj) \ := \ <C,\, SemType(\mathbb{A})> \quad \text{with}$$
$$SemType(\mathbb{A}) \ = \ SemType(\underline{\mathbb{A}}) \cup SemType(\mathbb{A}_R) \ :=$$
$$\{\ SemType(\underline{A}_1), \ldots, SemType(\underline{A}_m), SemType(A_{m+1}), \ldots, SemType(A_{m+n})\ \}$$

and domain:

$$Dom(SemType(CompSemObj)) := \{\ <C,\ \acute{\mathbb{A}}> \mid \acute{\mathbb{A}} \in Dom(SemType(\mathbb{A}))\ \} \ \text{ with}$$
$$Dom(SemType(\mathbb{A})) := Dom(SemType(\underline{A}_1)) \ \times \ \ldots \ \times \ Dom(SemType(\underline{A}_m)) \ \times$$
$$Dom(SemType(A_{m+1})) \ \times \ \ldots \ \times \ Dom(SemType(A_{m+n})) \ .$$

The attributes to be described in $\underline{\mathbb{A}}$ are determined by concept C associated with the complex semantic object, and therefore are specified in the underlying ontology. The semantic object type of $\underline{\mathbb{A}}$ *SemType*($\underline{\mathbb{A}}$) specifies the number and concepts underlying the corresponding semantic objects. This provides the prerequisite for the definition of semantic identity as it is given in Section 2.9. In contrast, the set of attributes specified in $\mathbb{A}_R$ may vary between different semantic objects of the same ontology concept, as is shown in the following example. In this way, complex semantic objects provide a flexible way to represent data with irregular structure, as it may be given by semistructured sources, or may result from the integration of different heterogeneous data sources.

```
CompSemObj_A1 =
    < FlightOffer, {
        < ClassOfService, "Economy",         {<ClassOfServiceCode, "FullClassName">} >,
        < Price, 2600,                       {<Currency, "DEM">, <Scale, 1>} >,
        < FlightSegment, {
            < FlightNumber, 400 >,
            < AirlineIdentifier, "LH",       {<AirlineIdentifierCode, "TwoLetterAirlineCode">} >,
            < DepartureDate, "Jun 06 1998",  {<DateFormat, "Mon DD YYYY">} >,
            < DepartureTime, "10:35",        {<TimeFormat, "HH:MM">} >,
            < DepartureAirport, "FRA",       {<AirportIdentifierCode, "ThreeLetterCode">} >,
            < ArrivalAirport, "JFK",         {<AirportIdentifierCode, "ThreeLetterCode">} >,
            < ArrivalTime, "13:00",          {<TimeFormat, "HH:MM"> } >,
            < Service, "M",                  {<ServiceCode, "OneLetterServiceCode">} >   } >   } >

CompSemObj_A2 =
    < FlightOffer, {
        < ClassOfService, "Economy",         {<ClassOfServiceCode, "FullClassName">} >,
        < Price, 2640,                       {<Currency, "DEM">, <Scale, 1>} >,
        < FlightSegment, {
            < FlightNumber, 1319 >,
            < AirlineIdentifier, "AF",       {<AirlineIdentifierCode, "TwoLetterAirlineCode">} >,
            < DepartureDate, "Jun 06 1998",  {<DateFormat, "Mon DD YYYY">} >,
            < DepartureTime, "10:25",        {<TimeFormat, "HH:MM">} >,
            < DepartureAirport, "FRA",       {<AirportIdentifierCode, "ThreeLetterCode">} >,
            < ArrivalAirport, "CDG",         {<AirportIdentifierCode, "ThreeLetterCode">} >,
            < ArrivalTime, "11:35",          {<TimeFormat, "HH:MM">} >   } >,
        < FlightSegment, {
            < FlightNumber, 6 >,
            < AirlineIdentifier, "AF",       {<AirlineIdentifierCode, "TwoLetterAirlineCode">} >,
            < DepartureDate, "Jun 06 1998",  {<DateFormat, "Mon DD YYYY">} >,
            < DepartureTime, "13:00",        {<TimeFormat, "HH:MM">} >,
            < DepartureAirport, "CDG",       {<AirportIdentifierCode, "ThreeLetterCode">} >,
            < ArrivalAirport, "JFK",         {<AirportIdentifierCode, "ThreeLetterCode">} >,
            < ArrivalTime, "15:00",          {<TimeFormat, "HH:MM">} >,
            < Service, "M",                  {<ServiceCode, "OneLetterServiceCode">} >,
            < Service, "S",                  {<ServiceCode, "OneLetterServiceCode">} >   } >   } >
```

Figure 4: MIX Representation of Source A

Example:
On the basis of the ontology described in the appendix, the data given by system A of the air travel example may be represented as complex semantic objects of concept *FlightOffer*[6] as linearized in Figure 4. Each offer is identified by its service class, price, and the constituting flight segments. In turn, flight segments are distinguished by their flight number, airline, and departure date. Additional properties, such as departure time, arrival airport, and meal services are not required for the unique identification of a flight segment and might not be given for all flight segments.

[6]Attributes used for the identification, i.e., attributes from $\underline{\mathbb{A}}$, are underlined for easier recognition.

Definition 20 (Simple and Complex Semantic Aspects)
Given the concept of a complex semantic object, we generalize the definition of the semantic context $\mathbb{S}$ *of a simple semantic object to be the set:*

$$\mathbb{S} := \{S_1, \ldots, S_k\} \ , \ k \in \mathbb{N}_0 \ ,$$

where the semantic aspects S_i *specified in* $\mathbb{S}$ *are represented as simple semantic objects according to Definition 2, or as complex semantic objects as defined in Definition 19. In the first case we call the corresponding semantic aspect* $< C,\ RepType(C) >$ *a* ***simple semantic aspect.*** *In the second case we call* $<C,\ SemType(\mathbb{A})>$ *a* ***complex semantic aspect.***

The definitions given so far, can be generalized according to this extended definition of a semantic context in a straightforward manner.

2.7 Semantic Conversion of Complex Semantic Objects

Following Section 2.6, a complex semantic object does not provide additional context information. Thus, the semantic context of a complex semantic object is given through the context information specified for its subobjects. This has been defined to keep the model simple. Accordingly, the concept of a conversion function can be directly extended for the application on complex semantic objects.

Definition 21 (Conversion Functions for Complex Semantic Objects)
Given a complex semantic object CompSemObj $= <C,\ \mathbb{A}>$*, and given context* $\mathbb{S}$*. A* ***(complex) conversion function*** Φ *with regard to* $SemType(\mathbb{S})$ *is a mapping function of the signature:*

$$\Phi \ : \ Dom(SemType(\mathbb{S})) \ \times \ \mathbb{CSO}_{\Phi} \ \longrightarrow \ \mathbb{CSO}_{\Phi}$$

that converts a complex semantic object CompSemObj into the complex semantic object CompSemObj′ that corresponds to the conversion of all subobjects $A \in \mathbb{A}$ *of CompSemObj according to all significant semantic aspects* $< C,\ RepType(C) > \ \in \ SemType(\mathbb{S})$*. Given* $\mathbb{A} = \{\underline{A}_1, \ldots, \underline{A}_m, A_{m+1}, \ldots, A_{m+n}\}$ *we define* Φ *recursively as*[7]*:*

$$\Phi\,(\,\mathbb{S},\ <C,\ \{\underline{A}_1, \ldots, \underline{A}_m, A_{m+1}, \ldots, A_{m+n}\} >) \ := \ <C,\ \{\,\breve{\Phi}(\mathbb{S},\ \underline{A}_1), \ldots, \breve{\Phi}(\mathbb{S},\ A_{m+n})\,\} >$$

with:

$$\breve{\Phi}\,(\,\mathbb{S}, A_i\,) \ := \begin{cases} \tilde{\phi}\,(\,\mathbb{S}, A_i\,), & \textit{if } A_i \text{ is a simple semantic object,} \\ \Phi\,(\,\mathbb{S}, A_i\,), & \textit{if } A_i \text{ is a complex semantic object,} \end{cases}$$

where $\tilde{\phi}$ *is the corresponding conversion function for simple semantic objects with regard to* $SemType(\mathbb{S})$*.*

Thus, a complex conversion function is a mapping function that converts a complex semantic object between different contexts by being recursively applied to all of its subobjects. If a given subobject is a simple semantic object we use the corresponding conversion function for simple semantic objects.

[7]We suppose here that all subobjects of $\mathbb{A}$ can be converted independently.

Example:
This example will illustrate the interrelations of the concepts introduced. Considering the conversion of the complex semantic object $SemObj_{A_1}$, as depicted in Figure 4. According to context $\{<Currency,\ \text{"}USD\text{"}>\}$ and conversion function $\Phi_{Currency}$ we get:

$$
\begin{aligned}
&\Phi_{Currency}\,(\{<Currency, \text{"}USD''>\},\\
&\qquad <FlightOffer,\,\{\\
&\qquad\quad <\underline{ClassOfService},\ \text{"}Economy'',\ \{<ClassCode, \text{"}FullName''>\}>,\\
&\qquad\quad <\underline{Price},\ 2600,\ \{<Currency, \text{"}DEM''>, <Scale, 1>\}>,\\
&\qquad\quad <\underline{FlightSegment},\ \{\ \dots\ \}>)\ =
\end{aligned}
$$

$$
\begin{aligned}
&<FlightOffer,\,\{\\
&\quad \phi_{Currency}\,(\{<Currency, \text{"}USD''>\},\ <\underline{ClassOfService},\ \text{"}Economy'',\ \{<ClassCode, \text{"}FullName''>\}>),\\
&\quad \phi_{Currency}\,(\{<Currency, \text{"}USD''>\},\ <\underline{Price},\ 2600,\ \{<Currency, \text{"}DEM''>, <Scale, 1>\}>),\\
&\quad \Phi_{Currency}\,(\{<Currency, \text{"}USD''>\},\ <\underline{FlightSegment},\ \{\ \dots\ \}>)\}>\ =
\end{aligned}
$$

$$
\begin{aligned}
&<FlightOffer,\,\{\\
&\quad <\underline{ClassOfService},\ \text{"}Economy'',\ \{<ClassCode, \text{"}FullName''>\}>,\\
&\quad <\underline{Price},\ 1430,\ \{<Currency, \text{"}USD''>, <Scale, 1>\}>,\\
&\quad <\underline{FlightSegment},\ \{\ \dots\ \}>\}>\ .
\end{aligned}
$$

Whereat, we suppose $\tilde{\phi} = \phi_{Currency}$, with $\phi_{Currency}$ being the conversion function from Section 2.4.

2.7.1 Properties of Composite Conversion Functions for Complex Semantic Objects

According to the definition of a composite conversion function for the conversion of complex semantic objects, the properties of totality, and losslessness of composite conversion functions as discussed in Section 2.3.3.2, can be extended to the class of the conversion functions for complex semantic objects.

Definition 22 (Total and Partial Conversion Functions)
Given a complex semantic object $CompSemObj = <C, \{\underline{A}_1, \dots, \underline{A}_m, A_{m+1}, \dots, A_{m+n}\}>$, and given Φ a (complex) conversion function with regard to the semantic type $SemType(\$)$. Then according to Definition 21

$$\Phi\,(\$, <C, \{\underline{A}_1, \dots, \underline{A}_m, A_{m+1}, \dots, A_{m+n}\}>)\ :=\ <C, \{\breve{\phi}\,(\$,\ \underline{A}_1), \dots, \breve{\Phi}\,(\$,\ A_{m+n})\}>$$

*is **total**, iff the composite conversion function $\breve{\Phi}$ is total according to Definition 11 or, recursively, Definition 22. Otherwise, we call Φ a **partial** conversion function.*

Definition 23 (Lossless and Lossy Conversion Functions)
Given a complex semantic object $CompSemObj = <C, \{\underline{A}_1, \dots, \underline{A}_m, A_{m+1}, \dots, A_{m+n}\}>$, and given Φ a (complex) conversion function with regard to the semantic type $SemType(\$)$. Then

$$\Phi\,(\$, <C, \{\underline{A}_1, \dots, \underline{A}_m, A_{m+1}, \dots, A_{m+n}\}>)\ :=\ <C, \{\breve{\Phi}\,(\$,\ \underline{A}_1), \dots, \breve{\Phi}\,(\$,\ A_{m+n})\}>$$

*is **lossless**, iff for all $\$_j \in Dom(SemType(\$))$, $1 \le j \le k$, and all $CompSemObj \in \mathbb{CSO}_{\Phi}$, for* *which* $\breve{\Phi}(A_i)$, $A_i \in \mathbb{A}$, $1 \le i \le m+n$, *is well defined, we have:*

$$\Phi(\$_k,\ \mathit{CompSemObj}) = \Phi(\$_k,\ \Phi(\$_{k-1}, \ldots \Phi(\$_1,\ \mathit{CompSemObj}) \ldots)) \ .$$

Otherwise, we call Φ *a* ***lossy*** *conversion function.*

From this and the definition of Φ, it follows that Φ is lossless, iff $\tilde{\phi}$, the corresponding conversion function for simple semantic objects, (see Definition 21) underlying Φ is lossless according to Definition 12.

2.8 Semantic Comparability of Complex Semantic Objects

In Section 2.4 we introduced the concept of semantic comparability of two simple semantic objects. In particular, simple semantic objects that are semantically equivalent with regard to a given target context and reference function, may be understood as representing the same information, i.e., they describe the same real world aspects. The statements given there can be generalized for the class of complex semantic objects in a straight forward manner.

Definition 24 (Semantic Comparability of Complex Semantic Objects)
Given two complex semantic objects $\mathit{CompSemObj}_A = <C,\ \{\underline{A}_1, \ldots, \underline{A}_m, A_{m+1}, \ldots, A_{m+n}\}>$ *and* $\mathit{CompSemObj}_B = <C,\ \{\underline{B}_1, \ldots, \underline{B}_m, B_{m+1}, \ldots, B_{m+n}\}>$ *of the same semantic type. That is, both objects represent the same aspects of the respective real world entity. Then for context* $\$$ *and a comparison operator* $\Theta \in \{``=",\ ``\neq"\}$ *defined for all* $\mathit{Dom}(\mathit{SemType}(A_i))$ *and* $\mathit{Dom}(\mathit{SemType}(B_i)), 1 \le i \le m+n$, *we define:*

$$<C,\ \mathbb{A}> \ \Theta_{\$}\ <C,\ \mathbb{B}>$$

with respect to context $\$$ *and the conversion function* Φ, *iff:*

$$\acute{A}_i\ \ \Theta_{\$}\ \ \acute{B}_i,\ 1 \le i \le m+n \ ,$$

$\acute{A}_i$, $\acute{B}_i$ *given by:*

$$<C, \{\underline{\acute{A}}_1, \ldots, \underline{\acute{A}}_m, \acute{A}_{m+1}, \ldots, \acute{A}_{m+n}\}> \ = \ \Phi(\$, <C, \{\underline{A}_1, \ldots, \underline{A}_m, A_{m+1}, \ldots, A_{m+n}\}>) \ ,$$
$$<C, \{\underline{\acute{B}}_1, \ldots, \underline{\acute{B}}_m, \acute{B}_{m+1}, \ldots, \acute{B}_{m+n}\}> \ = \ \Phi(\$, <C, \{\underline{B}_1, \ldots, \underline{B}_m, B_{m+1}, \ldots, B_{m+n}\}>) \ .$$

This means, the result of the semantic comparison of two complex semantic objects of the same semantic type is defined by the conversion of both objects to a common context $\$$, followed by the semantic comparison of the attributes of the converted subobjects. According to Definition 15, we call the semantic context $\$$ the target context, and Φ the reference function of the semantic comparison.

2.8.1 Absolute Semantic Comparability of Complex Semantic Objects

Based on Definition 24, the concept of absolute semantic comparability can be generalized for the class of complex semantic objects as follows.

Definition 25 (Absolute Semantic Comparability of Complex Semantic Objects)
Given two complex semantic objects $CompSemObj_A = <C, \mathbb{A}>$ *and* $CompSemObj_B = <C, \mathbb{B}>$ *of the same semantic type, with* $SemType(A_i) = SemType(B_i),\ A_i \in \mathbb{A},\ B_i \in \mathbb{B},\ 1 \leq i \leq m+n$. *That is, both objects represent the same aspects of the respective real world entity.* $CompSemObj_A$ *and* $CompSemObj_B$ *are* ***absolute semantic equivalent*** *with regard to the semantic type* $SemType(\$)$ *and the corresponding reference function* Φ*, denoted as:*

$$<C,\ \mathbb{A}> \ =_{SemType(\$)}\ <C,\ \mathbb{B}> \ ,$$

iff we have:

$$\forall\ \acute{\$} \in Dom(SemType(\$))\ :\ <C,\ \mathbb{A}> \ =_{\acute{\$}}\ <C,\ \mathbb{B}> \ .$$

Thus, two complex semantic objects are absolute semantic equivalent with regard to a given semantic type $SemType(\$)$ and the corresponding reference function Φ, if the result of the semantic comparison is independent of the particular values given in the target context for the semantic aspects of $SemType(\$)$. The corresponding definition for the comparison operator "$\neq$" can be given analogously.

Similar to the correlation between the properties of the reference function and the absoluteness of semantic comparison operators as discussed in Proposition 1, we can show according relationships for the class of composite conversion functions for complex semantic objects.

Proposition 3
Given two complex semantic objects $CompSemObj_A = <C, \mathbb{A}>$, and $CompSemObj_B = <C, \mathbb{B}>$ of the same semantic type. Then for a given semantic type $SemType(\$)$ we can show, that if the corresponding conversion function Φ is total and lossless according to Definition 22 and 23, each comparison operator $\Theta \in \{$"$=$", "$\neq$"$\}$ between $CompSemObj_A$ and $CompSemObj_B$ is absolute with regard to $SemType(\$)$ and function Φ.

In particular, with Definition 21 we get that, if the composite conversion function $\tilde{\phi}$ underlying Φ (see Definition 21) is total and lossless according Definition 11 and 12, respectively, each comparison operator $\Theta \in \{$"$=$", "$\neq$"$\}$ between $CompSemObj_A$ and $CompSemObj_B$ is absolute with regard to $SemType(\$)$ and reference function Φ.

Proof:
The proof for Proposition 3 is given here with regard to the semantic equivalence operator. The corresponding proof for the comparison operator "$\neq$" follows similarly. Given two complex semantic objects of the same semantic type with:

$$<C,\ \mathbb{A}> \ =_{\$}\ <C,\ \mathbb{B}>$$

concerning target context $\$$ and the total and lossless reference function Φ, with:

$$\begin{aligned} <C, \{\underline{\acute{A}}_1, \ldots, \underline{\acute{A}}_m, \acute{A}_{m+1}, \ldots, \acute{A}_{m+n}\}> \ &= \ \Phi\,(\$, <C, \{\underline{A}_1, \ldots, \underline{A}_m, A_{m+1}, \ldots, A_{m+n}\}>)\ , \\ <C, \{\underline{\acute{B}}_1, \ldots, \underline{\acute{B}}_m, \acute{B}_{m+1}, \ldots, \acute{B}_{m+n}\}> \ &= \ \Phi\,(\$, <C, \{\underline{B}_1, \ldots, \underline{B}_m, B_{m+1}, \ldots, B_{m+n}\}>)\ . \end{aligned}$$

Then, according to Definition 22, we have:

$$A_i \ =_{\$}\ B_i \ \ ,\ \ A_i \in \mathbb{A},\ B_i \in \mathbb{B},\ 1 \leq i \leq m+n\ .$$

Because Φ is total and lossless, it follows from Definition 22 and 23 that the conversion function $\tilde{\phi}$ underlying Φ is total and lossless. Thus, according to Proposition 1, for all simple semantic objects $A_j \in \mathbb{A}$ and $B_j \in \mathbb{B}$ we get:

$$A_j \ =_{SemType(\$)} \ B_j \ , \quad \text{i.e.,} \ \forall \ \acute{\$} \in Dom(SemType(\$)) \ : \ A_j \ =_{\acute{\$}} \ B_j \ .$$

Following from the recursive definition of Φ given in Definition 21, we get:

$$\forall \ \acute{\$} \in Dom(SemType(\$)) \ : \ <C, \ \mathbb{A}> \ =_{\acute{\$}} \ <C, \ \mathbb{B}>$$

and thus according to Definition 25:

$$<C, \ \mathbb{A}> \ =_{SemType(\$)} \ <C, \ \mathbb{B}> \ .$$

2.9 Semantic Identity

Different complex semantic objects of the same ontology concept may refer to different semantic contexts or may describe different aspects of the entity they represent. This raises the question, when can two semantic objects represented differently in this sense be said to describe the same real world phenomenon? The following definition introduces the concept of semantic identity of semantic objects with regard to a given semantic context and conversion function. If two semantic objects are semantically identical on the basis of the underlying ontology and dependent on the same context, we classify them as being two representations of the same real world phenomenon.

Definition 26 (Semantic Identity)
*Given two complex semantic objects $CompSemObj_A = <C, \mathbb{A}>$ and $CompSemObj_B = <C, \mathbb{B}>$ of the same ontology concept. This means, both objects are identified by the same set of attributes, i.e., $SemType(\underline{\mathbb{A}}) = SemType(\underline{\mathbb{B}})$. Then $CompSemObj_A$ and $CompSemObj_B$ are **semantically identical** with regard to context $\$$ and the reference function Φ, denoted as:*

$$CompSemObj_A \ \ id_{\$} \ \ CompSemObj_B \ ,$$

iff recursively:

$$A_i \ \ id_{\$} \ \ B_i, \qquad A_i \in \underline{\mathbb{A}}, \ \ B_i \in \underline{\mathbb{B}}, \ \ 1 \leq i \leq m \ ,$$

for context $\$$ and function Φ. In the case of two simple semantic objects $SemObj_1$ and $SemObj_2$, the semantic identity of these objects is defined as:

$$SemObj_1 \ \ id_{\$} \ \ SemObj_2 \ \ \Leftrightarrow \ \ SemObj_1 \ =_{\$} \ SemObj_2 \ .$$

This means, two complex semantic objects of the same ontology concept are semantically identical with regard to a given context and reference function if, recursively, their *identifying* subobjects are semantically identical with regard to this context and conversion function.

At the lowest level of this recursion, simple semantic objects must be compared. Two simple semantic objects are semantically identical with respect to a given context and conversion function if they are semantically equivalent with regard to this context and conversion function, since identity and equivalence are the same for atomic values.

Thus, *semantic identity* defines whether two semantic objects describe the same real world object. In contrast, *semantic equivalence* describes whether two semantic objects represent the same information. By definition, semantically equivalent semantic objects are semantically identical since they concur in both the identifying and all other attributes. The reverse is not always true since two semantically identical objects may have the same identifying attributes, e.g., airline, flight number and date, but different non-identifying attributes, such as meal service. See Section 3 for an example.

Definition 27 (Absolute Semantic Identity)
Given two semantic objects $Obj_1$and Obj_2. Obj_1 and Obj_2 are ***absolute semantically identic*** *with regard to semantic type $SemType(\$)$ and the corresponding conversion function Φ denoted as:*

$$Obj_1 \ \ id_{SemType(\$)} \ \ Obj_2 \ ,$$

iff we have:

$$\forall \ \acute{\$} \in Dom(SemType(\$)) \ : \ Obj_1 \ \ id_{\acute{\$}} \ \ Obj_2 \ .$$

Thus, two semantic objects are absolute semantically identic with regard to a given semantic type $SemType(\$)$ and the corresponding reference function Φ, if the property of semantic identity between them is independent from the particular values given for the semantic aspects of $SemType(\$)$ in the respective target context.

```
CompSemObjB1 =
    < FlightOffer, {
        < ClassOfService, "Y",              {<ClassOfServiceCode, "OneLetterClassCode">} >,
        < Price, 1430,                      {<Currency, "USD">, <Scale, 1>} >,
        < FlightSegment, {
            < FlightNumber, 400 >,
            < AirlineIdentfier, "Lufthansa",        {<AirlineIdentifierCode, "FullAirlineName">} >,
            < DepartureDate, "Jun 06, 1998",        {<DateFormat, "Mon DD, YYYY">} >,
            < DepartureTime, "10:35 AM",            {<TimeFormat, "HH:MM AM/PM">} >,
            < DepartureAirport, "FRA",              {<AirportIdentifierCode, "ThreeLetterCode">} >,
            < ArrivalAirport, "JFK",                {<AirportIdentifierCode, "ThreeLetterCode">} >,
            < ArrivalTime, "01:00 PM",              {<TimeFormat, "HH:MM AM/PM">} >,
            < Distance, 3850,                       {<Unit, "mile">, <Scale, 1>} >   } >   }>

CompSemObjB2 =
    < FlightOffer, {
        < ClassOfService, "Y",              {<ClassOfServiceCode, "OneLetterClassCode">} >,
        < Price, 1450,                      {<Currency, "USD">, <Scale, 1>} >,
        < FlightSegment, {
            < FlightNumber, 107 >,
            < AirlineIdentfier, "Delta",            {<AirlineIdentifierCode, "FullAirlineName">} >,
            < DepartureDate, "Jun 06, 1998",        {<DateFormat, "Mon DD, YYYY">} >,
            < DepartureTime, "09:35 AM",            {<TimeFormat, "HH:MM AM/PM">} >,
            < DepartureAirport, "FRA",              {<AirportIdentifierCode, "ThreeLetterCode">} >,
            < ArrivalAirport, "JFK",                {<AirportIdentifierCode, "ThreeLetterCode">} >,
            < ArrivalTime, "12:00 PM",              {<TimeFormat, "HH:MM AM/PM">} >,
            < Distance, 3850,                       {<Unit, "mile">, <Scale, 1>} >   } >   }>
```

Figure 5: MIX Representation of Source B

3 Data Integration on the Basis of MIX

The data given by the reservation systems introduced in Section 1.3 may be parsed and represented as semantic objects of concept *FlightOffer* as linearized in Figures 4 and 5. By circumventing the need to agree on all attributes, the two sources will be able to agree on the same meaning for *FlightOffer*. Both data sources make different semantic assumptions which result in different semantic contexts at the sources. We will show how to arrive at a common representation of the data through the integration process.

The process of integrating data represented on the basis of the MIX representation model takes place in two steps. First, the semantic objects have to be converted to a common context, which can be specified by the application interested in the data, by using appropriate conversion functions. For example, for the aspects of *TimeFormat* and *AirlineIdentifierCode* we may specify the following mapping rules:

$$\text{"}hh\!:\!mm\text{"}\ [\text{"HH:MM"}] \Rightarrow \begin{cases} \text{"00:}mm\text{ AM"}\ \ [\text{"HH:MM AM/PM"}], & \text{if } hh = 0 \\ \text{"}hh\!:\!mm\text{ AM"}\ \ [\text{"HH:MM AM/PM"}], & \text{if } 0 < hh < 12 \\ \text{"12:}mm\text{ PM"}\ \ [\text{"HH:MM AM/PM"}], & \text{if } hh = 12 \\ \text{"}(hh-12)\!:\!mm\text{ PM"}\ \ [\text{"HH:MM AM/PM"}], & \text{if } hh > 12 \end{cases}$$

$$\text{"}hh\!:\!mm\ XX\text{"}\ [\text{"HH:MM AM/PM"}] \Rightarrow \begin{cases} \text{"00:}mm\text{"}\ \ [\text{"HH:MM"}], & \text{if } XX = \text{"AM"} \wedge hh = 12 \\ \text{"}hh\!:\!mm\text{"}\ \ [\text{"HH:MM"}], & \text{if } XX = \text{"AM"} \wedge hh \neq 12 \\ \text{"12:}mm\text{"}\ \ [\text{"HH:MM"}], & \text{if } XX = \text{"PM"} \wedge hh = 12 \\ \text{"}(hh+12)\!:\!mm\text{"}\ \ [\text{"HH:MM"}], & \text{if } XX = \text{"PM"} \wedge hh \neq 12\ , \end{cases}$$

and

$$\begin{aligned} &XX\ \ [\text{"TwoLetterAirlineCode"}] &&\Rightarrow\ \ \mathit{FullNameOf}(XX)^{5} &&[\text{"FullAirlineName"}] \\ &name\ \ [\text{"FullAirlineName"}] &&\Rightarrow\ \ \mathit{TwoLetterCodeOf}(name)^{5} &&[\text{"TwoLetterAirlineCode"}]\ . \end{aligned}$$

$ = { < AirlineIdentifierCode, "TwoLetterAirlineCode" >,
< AirportIdentifierCode, "ThreeLetterCode" >,
< DateFormat, "Mon DD YYYY" >,
< TimeFormat, "HH:MM AM/PM" >,
< ClassOfServiceCode, "OneLetterClassCode" >,
< ServiceCode, "OneLetterServiceCode" >,
< Currency, "USD" >,
< Unit, "mile" >,
< Scale, 1 > }

Figure 6: Common Representation Context

In the second step, semantic objects which are semantically identical are identified and integrated into a common representation. Using context $, depicted in Figure 6, as a common representation context and the conversion functions introduced so far, $CompSemObj_{A_1}$ and $CompSemObj_{B_1}$ may be classified as semantically identical according to Section 2.9 because they represent the same flight offer.

[5] These mappings can be realized easily by appropriate mapping tables.

Semantically identical MIX objects may be interpreted as being representatives of the same real world phenomenon. Therefore, they are merged into one semantic object by unification of their attribute sets, whereat attributes that are semantically identical are fused in turn. Attributes described in both objects that are semantically equivalent are represented only once. Applying the corresponding conversion functions, the resulting data can be represented as shown in Figure 7, where $CompSemObj_{A_1}$ and $CompSemObj_{B_1}$ have been merged into $CompSemObj_{AB}$.

```
CompSemObj_AB =
    < FlightOffer, {
        < ClassOfService, "Y",              {<ClassOfServiceCode, "OneLetterClassCode">} >,
        < Price, 1430,                      {<Currency, "USD">, <Scale, 1>} >,
        < FlightSegment, {
            < FlightNumber, 400 >,
            < AirlineIdentfier, "LH",           {<AirlineIdentifierCode, "TwoLetterAirlineCode">} >,
            < DepartureDate, "Jun 06 1998",     {<DateFormat, "Mon DD YYYY">} >,
            < DepartureTime, "10:35 AM",        {<TimeFormat, "HH:MM AM/PM">} >,
            < DepartureAirport, "FRA",          {<AirportIdentifierCode, "ThreeLetterCode">} >,
            < ArrivalAirport, "JFK",            {<AirportIdentifierCode, "ThreeLetterCode">} >,
            < ArrivalTime, "01:00 PM",          {<TimeFormat, "HH:MM AM/PM">} >,
            < Service, "M",                     {<ServiceCode, "OneLetterServiceCode">} >,
            < Distance, 3850,                   {<Unit, "mile">, <Scale, 1>} >   } >   } >
CompSemObj'_A2 =
    < FlightOffer, {
        < ClassOfService, "Y",              {<ClassOfServiceCode, "OneLetterClassCode">} >,
        < Price, 1452,                      {<Currency, "USD">, <Scale, 1>} >,
        < FlightSegment, {
            < FlightNumber, 1319 >,
            < AirlineIdentifier, "AF",          {<AirlineIdentifierCode, "TwoLetterAirlineCode">} >,
            < DepartureDate, "Jun 06 1998",     {<DateFormat, "Mon DD YYYY">} >,
            < DepartureTime, "10:25 AM",        {<TimeFormat, "HH:MM AM/PM">} >,
            < DepartureAirport, "FRA",          {<AirportIdentifierCode, "ThreeLetterCode">} >,
            < ArrivalAirport, "CDG",            {<AirportIdentifierCode, "ThreeLetterCode">} >,
            < ArrivalTime, "11:35 AM",          {<TimeFormat, "HH:MM AM/PM">} >   } >,
        < FlightSegment, {
            < FlightNumber, 6 >,
            < AirlineIdentifier, "AF",          {<AirlineIdentifierCode, "TwoLetterAirlineCode">} >,
            < DepartureDate, "Jun 06 1998",     {<DateFormat, "Mon DD YYYY">} >,
            < DepartureTime, "01:00 PM",        {<TimeFormat, "HH:MM AM/PM">} >,
            < DepartureAirport, "CDG",          {<AirportIdentifierCode, "ThreeLetterCode">} >,
            < ArrivalAirport, "JFK",            {<AirportIdentifierCode, "ThreeLetterCode">} >,
            < ArrivalTime, "03:00 PM",          {<TimeFormat, "HH:MM AM/PM">} >,
            < Service, "M",                     {<ServiceCode, "OneLetterServiceCode">} >,
            < Service, "S",                     {<ServiceCode, "OneLetterServiceCode">} >   } >   } >
SemObj'_B2 =
    < FlightOffer, {
        < ClassOfService, "Y",              {<ClassOfServiceCode, "OneLetterClassCode">} >,
        < Price, 1450,                      {<Currency, "USD">, <Scale, 1>} >,
        < FlightSegment, {
            < FlightNumber, 107 >,
            < AirlineIdentfier, "DL",           {<AirlineIdentifierCode, "TwoLetterAirlineCode">} >,
            < DepartureDate, "Jun 06 1998",     {<DateFormat, "Mon DD YYYY">} >,
            < DepartureTime, "09:35 AM",        {<TimeFormat, "HH:MM AM/PM">} >,
            < DepartureAirport, "FRA",          {<AirportIdentifierCode, "ThreeLetterCode">} >,
            < ArrivalAirport, "JFK",            {<AirportIdentifierCode, "ThreeLetterCode">} >,
            < ArrivalTime, "12:00 PM",          {<TimeFormat, "HH:MM AM/PM">} >,
            < Distance, 3850,                   {<Unit, "mile">, <Scale, 1>} >   } >   } >
```

Figure 7: Unified Data Representation

4 Related Research

There has been a lot of research on the issues of metadata to describe the semantics of data, as well as flexible data models to support the representation and integration of heterogeneous data. However, we mention here only three closely related approaches.

The data model proposed in [SSR94] supports the explicit representation of context information of a given data value, as well as the conversion of this data between different contexts. The semantic context is represented as metadata that describes the organization and meaning of the data value. The model is strictly value-based and limited to the exchange of atomic values. Thus, it lacks the possibility of defining composite objects that can be handled as one unit. Our concept of a semantic object extends the concepts discussed in [SSR94] with regard to complex, maybe irregularly structured data objects. They assume a common vocabulary. MIX makes the common vocabulary *explicit* and provides both for the *exchange* of vocabularies, and their *extensibility*.

XML [XML98] provides a flexible, self-describing data model for the representation and exchange of structured and semistructured data similar to the MIX model. The XML standard supports a textual representation of data by using application-specific tags. These tags may be used to explicitly refer to the meaning of the represented data, and may be specified in a document type definition (DTD). Optional, required or fixed attributes can be determined for every tag to further describe its meaning or function. However, XML does not enforce a semantically meaningful data exchange per se, since different providers can define different tags to represent the same or semantically similar information. Furthermore, because XML is supposed to be a very flexible though simple model for data exchange, it does not support the integration of heterogeneous data. In contrast, MIX supports an explicit representation of semantic differences underlying the data, and specifies how data based on this representation may be converted to a common representation.

In addition, MIX has some similarities with the Object Exchange Model (OEM) [PGW95]. The OEM is a data model well-suited for the representation of data with heterogeneous structure. Besides the actual data value, each data object has a unique object-ID, a type which determines its representation, and a label which provides additional information concerning the meaning associated with it. The OEM, as well as the MIX model, are self-describing data models in the sense that structure and meaning of the data objects are given as part of the available data objects. Both data models provide a highly flexible description model, especially well suited for the representation of semistructured data.

However, there are some important differences. First, in the OEM objects are identified via system-wide object identifiers. In contrast to this, the available data objects in MIX have certain attributes associated with them which support the identification of data objects based on their information content. Second, different from the OEM model, where data objects have source-specific labels, concept labels associated with MIX objects come from domain-specific vocabularies for which a common agreement about their meaning has been reached. These vocabularies exist and are known to users working in specific application domains. Finally, OEM is tailored mainly to the representation of data with irregular structure. In addition to this, the MIX model also supports an explicit representation of the semantics underlying the data, and provides conversion functions to convert data between different semantic contexts.

In this way, the MIX model combines concepts of a flexible, self-describing data model suitable for the representation of semistructured data, with concepts concerning the explicit description of the semantic assumptions underlying this data.

5 Conclusion

An integrated use of the data sources available online requires an explicit description of both the structure and the assumptions about the meaning of the data. In this paper we presented a flexible data model that supports the representation of data together with metadata that describes its organization and semantics. We showed how semistructured data can be represented in a natural way, and on a uniform interpretation basis by using this model. We use the MIX model in a project for integrating structured and semistructured data sources from the Internet. The prototype of a Java-based implementation exists for MIX and the MIX integration environment. Current research is concerned with the extension of the representation of conversion functions, and with the extraction of MIX representations for a wider range of semistructured data.

OEM and XML provide support for the representation and exchange of data in terms of attribute/value pairs, with user defined labels. However, this alone will not provide for semantically meaningful exchange of data, and interoperability among data providers and consumers. This is because, different providers may define their own ways of using attribute/value pairs to represent the same information. In contrast to this, MIX offers data providers and consumers the possibility to refer to a commonly agreed upon vocabulary, and provides hooks for the introduction of conversion functions to convert the available data to a common representation.

Unlike OEM, XML, and semantic values as introduced in [SSR94] which can only represent object state, MIX objects are full fledged data objects in the sense of object-oriented programming languages. This means, domain-specific but application-independent processing code, e.g., conversion functions, can be specified in the common ontology, and associated with these objects. An application may access these data objects without any further parsing. It further offers the possibility of extending common vocabularies on the receiver side.

If the Internet is to develop further to support advanced application requirements, we need data models which allow a flexible representation of state and processing code, as well as a general framework which directly supports the integration of heterogeneous resources. We believe the MIX model provides a first step to satisfy these needs.

References

[Abit97] Abiteboul, S.: *Querying Semi-Structured Data*, In: Proceedings of the International Conference on Database Theory, Delphi, Greece, 1997

[ACC+97] Abiteboul, S.; Cluet, S.; Christophides, V.; Milo, T.; Moerkotte, G.; Simeon, J.: *Querying Documents in Object Databases*, In: Journal on Digital Libraries, Vol.1, No.1, April 1997

[AK97] Ashish, N.; Knoblock, C.: *Wrapper Generation for Semi-structured Internet Sources*, In: Proceedings of the Workshop on Management of Semi-structured Data, Tucson, Arizona, 1997

[AMM97] Atzeni, P.; Mecca, G.; Merialdo, P.: *To Weave the Web*, In: Proceedings of the 23rd VLDB Conference, Athens, Greece, 1997

[Bune97] Bunemann, P.: *Semistructured Data - Tutorial*, In: Proceedings of the Symposium on Principles of Database Systems (PODS), Tucson, Arizona, 1997

[CGH+94] Chawathe, S.; Garcia-Molina, H.; Hammer, J.; et al.: *The TSIMMIS Project: Integration of Heterogeneous Information Sources*, In: Proceedings of IPSJ Conference, Tokyo, Japan, 1994

[CRE87] Czejdo, B.; Rusinkiewicz, M.; Embley, D.W.: *An Approach to Schema Integration and Query Formulation in Federated Database Systems*, In: Proceedings of the 3rd IEEE Conference on Data Engineering, 1987

[FFR96] Farquhar, A.; Fikes, R.; Rice, J.: *The Ontolingua Server: A Tool for Collaborative Ontology Construction*, In: Proceedings of the 10th Knowledge Acquisition for Knowledge-Based Systems Workshop, Alberta, Canada, 1996

[GMS94] Goh, C.; Madnick, S.; Siegel, M.: *Context Interchange: Overcoming the Challenges of Large-Scale Interoperable Database Systems in a Dynamic Environment*, In: Proceedings of the 3rd International Conference on Information and Knowledge Management (CIKM'94), Gaithersburg, 1994

[GMS95] Goh, C.; Madnick, S.; Siegel, M.: *Ontologies, Contexts, and Mediation: Representing and Reasoning about Semantic Conflicts in Heterogeneous and Autonomous Systems*, Sloan School of Management, Working Paper #3848; also CISL Working Paper CISL 95-04

[Grub95] Gruber, T.: *Toward Principles for the Design of Ontologies Used for Knowledge Sharing*, In: International Journal of Human-Computer Studies, Vol.43, No.5/6, 1995

[Guar96] Guarino, N.: *Understanding, Building, and Using Ontologies*, In: Proceedings of the 10th Knowledge Acquisition for Knowledge-Based Systems Workshop, Alberta, Canada, 1996

[HGC+97] Hammer, J.; Garcia-Molina, H.; Cho, J.; Aranha, R.; Crespo, A.: *Extracting Semistructured Information from the Web*, In: Proceedings of the Workshop on Management of Semistructured Data, Tucson, Arizona, 1997

[HMV96] Heiler, S.; Miller, R.; Ventrone, V.: *Using Metadata to Address Problems of Semantic Interoperability in Large Object Systems*, In: Proceedings of the 1st IEEE Metadata Conference, Silver Spring, Maryland, 1996

[ISO86] ISO 8879. *Information Processing – Text and Office Systems – Standard Generalized Markup Language (SGML)*, 1986

[KS96] Kashyap, V.; Sheth, A.: *Semantic and Schematic Similarities between Database Objects: A Context-based Approach*, In: The VLDB Journal, Vol.5, No.4, 1996

[Lamp85] Lamport, L.: *LaTex - A Document Preparation System*, Addison-Wesley, Reading, MA, 1985

[LMR90] Litwin, W; Mark, L.; Roussopoulos, N.: *Interoperability of Multiple Autonomous Databases*, In: ACM Computing Surveys, Vol.22, No.3, Sep. 1990

[MAG+97] McHugh, J.; Abiteboul, S.; Goldman, R.; Quass, D.; Widom, J.: *Lore: A Database Management System for Semistructured Data*, Technical Report, Stanford University Database Group, Feb. 1997

[MKIS98] Mena, E.; Kashyap, V.; Illarramendi, A.; Sheth, A.: *Domain Specific Ontologies for Semantic Information Brokering on the Global Information Infrastructure*, In: Proceedings of the International Conference on Formal Ontology in Information Systems, Trento, Italy, 1998

[MMM97] Mendelzon, A.O.; Mihaila, G.A.; Milo, T.: *Querying the World Wide Web*, In: International Journal on Digital Libraries, 1, 1997

[MMS98] Moulton, A.; Madnick, S.E.; Siegel, M.D.: *Context Mediation on Wall Street*, In: Proceedings of the 3rd IFCIS International Conference on Cooperative Information Systems, (CoopIS'98), New York, 1998

[MR90] Mark, L.; Roussopoulos, N.: *Information Interchange Between Self-Describing Databases*, In: Information Systems, Vol.15, No.4, 1990

[ON94] Ouksel, A.M.; Naiman, C.F.: *Coordinating Context Building in Heterogeneous Information Systems*, In: Journal of Intelligent Information Systems, Vol.3, No.2, 1994

[PGW95] Papakonstantinou, Y.; Garcia-Molina, H.; Widom, J.: *Object Exchange Across Heterogeneous Information Sources*, In: Proceedings of the IEEE International Conference on Data Engineering, Taipei, Taiwan, 1995

[RHJ97] Ragget, D.; Le Hors, A.; Jacobs, I.: *HTML 4.0 Specification*, 1997, `http://www.w3.org/hypertext/WWW/MarkUp/MarkUp.html`

[RS95]	Rosenthal, A.; Sciore, E.: *Description, Conversion, and Planning for Semantic Interoperability*, In: Proceedings of the IFIP WG 2.6 Working Conference on Database Applications Semantics (DS-6), Atlanta, Georgia, USA, 1995
[SG89]	Sheth, A., Gala, S.: *Attribute Relationships: An Impediment in Automating Schema Integration*, In: Proceedings of the NSF Workshop on Heterogeneous Databases, Dec. 1989
[SL97]	Smith, D.; Lopez, M.: *Information extraction for semi-structured documents*, In: Proceedings of the Workshop on Management of Semi-structured Data, Tucson, Arizona, 1997
[SM91]	Siegel, M.; Madnick, S.: *A Metadata Approach to Resolving Semantic Conflicts*, In: Proceedings of the 17th Conference on Very Large Data Bases, Barcelona, Spanien, 1991
[SSR92]	Sciore, E.; Siegel, M.; Rosenthal, A.: *Context Iinterchange using Meta-Attributes*, In: Proceedings of the 1st International Conference on Information and Knowledge Management (CIKM), 1992
[SSR94]	Sciore, E.; Siegel, M.; Rosenthal, A.: *Using Semantic Values to Facilitate Interoperability Among Heterogeneous Information Systems*, In: ACM Transactions on Database Systems, Vol.19, No.2, June 1994
[Stee90]	Steele, G.L.: *Common Lisp – The Language (Second Edition)*, Digital Press, 1999
[UNI94]	UNICORN Maintenance Authority: *UNICORN Application Standard (TTIP03) V4.0*, Travel Technology Initiative Ltd., c/o Cosmos Management Services Department, Bromley Kent, 1994
[VM98]	van der Vet, P.E.; Mars, N.J.I.: *Bottom-Up Construction of Ontologies*, In: IEEE Transactions on Knowledge and Data Engineering, Vol.10, No.4, 1998
[XML98]	World Wide Web Consortium: *Extensible Markup Language (XML) 1.0*, Feb. 10, 1998, `http://www.w3.org/TR/REC-xml`

Appendix A AirTravel Ontology

AirlineIdentifier	represented as String
Description:	Identifies an international airline.
AirlineIdentifierCode	represented as String
Description:	Identification code of an international airline. Every scheduled airline has a two letter code, and most also have a three letter code. E.g., Lufthansa is “LH”.
Domain:	{“TwoLetterAirlineCode”, “ThreeLetterAirlineCode”, “FullAirlineName”}
	TwoLetterAirlineCode = international two letter airline code
	ThreeLetterAirlineCode = international three letter airline code
AirportIdentifier	represented as String
Description:	Identifies an international airport.
AirportIdentifierCode	represented as String
Description:	Identification code of an international airport, for example “FRA” for Frankfurt Main International Airport.
Domain:	{“ThreeLetterAirportCode”, “FullAirportName”}
	ThreeLetterAirportCode = international three letter airport code
	FullAirportName = full official airport name
ArrivalDate	represented as DateTimeOntology.Date
Description:	Specifies the arrival date of a flight.
ArrivalTime	represented as DateTimeOntology.Time
Description:	Specifies the approximate arrival time of a flight.
ClassOfService	represented as String
Description:	Specifies a class of service for a flight, either First, Business, or Economy Class.
ClassOfServiceCode	represented as String
Description:	Identification code of a class of service.
Domain:	{“OneLetterClassCode”, “FullClassName”}
	OneLetterClassCode = {F(First Class), C(Business Class), Y(Economy Class)}

	FullClassName = {“FirstClass”, “BusinessClass”, “EconomyClass”}
DepartureAirport	represented as AirTravelOntology.AirportIdentifier
DESCRIPTION:	The DepartureAirport identifies the departure airport. It may be represented either as a full airport name, for example Tempelhof; Berlin, Germany, or by a specific three letter airport code, for example “THF”.
DepartureDate	represented as DateTimeOntology.Date
DESCRIPTION:	Specifies the departure date of a flight.
DepartureTime	represented as DateTimeOntology.Time
DESCRIPTION:	Specifies the approximate departure time of a flight.
DestinationAirport	represented as AirTravelOntology.AirportIdentifier
DESCRIPTION:	The DestinationAirport identifies the departure airport of a flight. It may be represented either as a full airport name, for example Tempelhof; Berlin, Germany, or by a specific three letter airport code, for example “THF”.
FlightNumber	represented as Integer
DESCRIPTION:	Specifies an airline specific flight identification number.
FlightOffer	represented as a Complex Semantic Object
DESCRIPTION:	A FlightOffer represents a flight offer given by a travel agency, which can be booked by a customer.
IDENTIFIER:	AirTravelOntology.ClassOfService AirTravelOntology.Price set of AirTravelOntology.FlightSegment
FlightSegment	represented as a Complex Semantic Object
DESCRIPTION:	If a flight involves a connection, i.e., a passenger has to change planes, this flight is divided into two flight segments.
IDENTIFIER:	AirTravelOntology.FlightNumber AirTravelOntology.AirlineIdentifier AirTravelOntology.DepartureDate
Price	represented as FinanceOntology.MonetaryQuantity
DESCRIPTION:	The price specifies the fare of a flight offer.
Scale	represented as Real
DESCRIPTION:	Specifies the scale factor of a given numerical value.
Service	represented as String
DESCRIPTION:	Denotes a special flight service, e.g., serving lunch or showing a movie.
ServiceCode	represented as String
DESCRIPTION:	Identification code of a flight service.

DOMAIN:	{“OneLetterServiceCode”, “FullServiceName”}
	OneLetterServiceCode = {D(Dinner), L(Lunch), S(Snack), V(Movie), ...}
	FullServiceName = {“Dinner”, “Lunch”, “Snack”, “Movie”, ...}

Unit	represented as String
DESCRIPTION:	Represents a constant quantity that serves as a unit of measure for some physical dimension.

...