

A Retrospective on Semantics and Interoperability Research

Bernhard Haslhofer and Erich J. Neuhold

Abstract Interoperability is a qualitative property of computing infrastructures that denotes the ability of sending and receiving systems to exchange and properly interpret information objects across system boundaries. Since this property is not given by default, the interoperability problem and the representation of semantics have been an active research topic for approximately four decades. Early database models such as the Relational Model used schemas to express semantics and implicitly aimed at achieving interoperability by providing programming independence of data storage and access. Thereafter the Entity Relationship Model was introduced providing the basic building blocks of modeling real-world semantics. With the advent of distributed and object-oriented databases, interoperability became an obvious need and an explicit research topic. After a number of intermediate steps such as hypertext and (multimedia) document models, the notions of semantics and interoperability became what they have been over the last ten years in the context of the World Wide Web. With this article we contribute a retrospective on semantics and interoperability research as applied in major areas of computer science. It gives domain experts and newcomers an overview of existing interoperability techniques and points out future research directions.

1 Introduction

Whenever an application processes data it must reflect the meaning—the *semantics*—of these data. Since this awareness is not given by default, the application designer needs to define a model, identify and *structure* atomic data units, and describe their meaning. Only if an application is aware of the structure and semantics of data, can it process them correctly. In this context, we often find the distinction between *data*, *information*, and *knowledge*, which has been the subject of intensive discussions in the Information Science literature for years. For a more comprehensive and actual discussion of these terms we refer to Rowley [53]. Here, we simply

B. Haslhofer (✉)

Department of Distributed and Multimedia Systems, University of Vienna, Liebiggasse 4/3-4,
1010 Vienna, Austria

e-mail: bernhard.haslhofer@univie.ac.at

define *data* as being symbols without any meaning and *information objects* as being a collection of data that carry semantics, which is a pre-condition for correct interpretation.

Interoperability problems arise when distinct applications communicate and exchange information objects with each other: often the structure and semantics of these objects is defined by autonomous designers, each having an individual interpretation of the real world in mind. When an object leaves the boundary of a sending system or application, the interpretation of these objects in a receiving application is often not possible due to the *heterogeneities* between the involved applications.

The problem of *how to represent semantics* and *how to establish interoperability* between information objects in distinct autonomous, distributed, and heterogeneous information systems has been a central and very active topic in database and information systems research throughout the past four decades. While the motivation in early database systems was to achieve data independence and interoperation for data-oriented applications, the topic has become increasingly important with the advent of distributed (multimedia) databases and information systems. Today it is still a major research issue in the largest currently existing (multimedia) information system—the World Wide Web.

The heterogeneities that impede systems and applications from being interoperable were investigated several times in different domains (e.g., [50, 54, 60, 62]). Although the notions vary, we can broadly categorize them as follows:

- *Technical Heterogeneities*: denotes all system platform and exchange protocol differences that prevent applications from sending and receiving information objects.
- *Structural and Syntactic Heterogeneities*: occur when data units in information objects are represented using different structures and syntactic conventions.
- *Semantic Heterogeneities*: are conflicts that occur because of the differences in the semantics of data units.

Analogous to these heterogeneity definitions we can define the various types of interoperability that can be achieved: *technical*, *structural and syntactic*, and *semantic interoperability*. In the following, when we use the term interoperability, we mainly refer to the latter two notions.

Before proceeding with our analysis of the various approaches that were developed for achieving interoperability, we introduce an illustrative example, which we will use throughout this work to explain the technical characteristics of these approaches. We assume a scenario in which two film studios, denoted as Studio A and Studio B, independently set up internal movie databases. Over the years both studios collected a large amount of data about movies; they have now decided to share and exchange these data. Figure 1 depicts the differences in how these two studios represent information about the same real-world movie. We allow that an actor can play in several movies and a movie has several actors. The notation we are using here is abstract and represents only the available information. It is not bound to any semantic modeling technique because this is what we want to achieve in the subsequent sections.

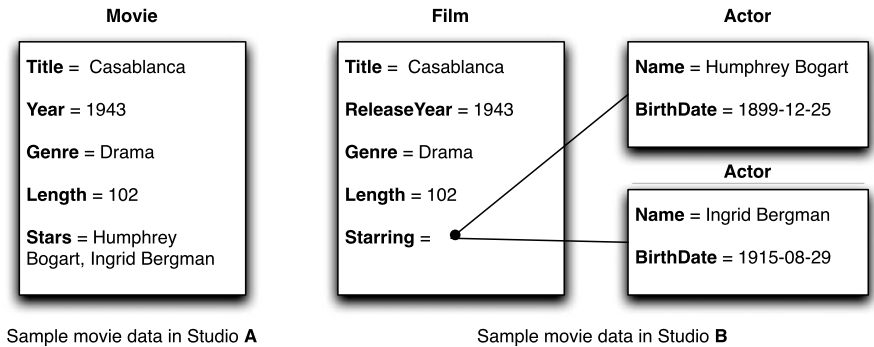


Fig. 1 Illustrative Example. *Studio A* records for each *Movie* its title, the year when it was first presented, the genre, its length, and the stars playing in the movie. *Studio B* records for each *Film* the title, the releaseYear, the genre, the length, and for each starring role the name and birthDate of the *Actor*

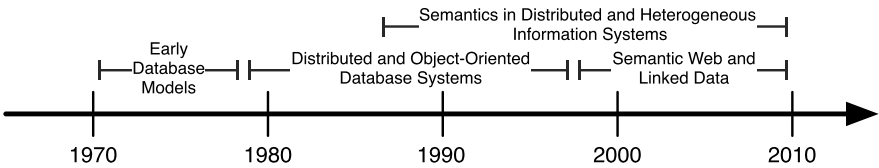


Fig. 2 Semantics and Interoperability Research in Computer Science

The aim of this chapter is to provide a retrospective on the developments in semantics and interoperability research throughout the past four decades from the perspective of database and information system research. Solutions developed by other disciplines (e.g., Information Retrieval, Data Mining, or Artificial Intelligence), that of course encounter similar problems, are out of the scope of this paper. We will present a selected but, as we believe, representative set of approaches that enable the expression of data semantics and/or allow us to deal with the heterogeneities between applications. Our illustrative example will help us to explain the technical characteristics of some of these approaches.

As illustrated in Fig. 2, we start our retrospective in the early 1970s and present early database models in Sect. 2. Then, in Sect. 3, we move along to distributed databases and object-oriented database models, which allow application-oriented and context-dependent design of databases. In Sect. 4, we describe major models and languages for the representation of semantics in distributed and heterogeneous information systems. Then, in Sect. 5, we describe the Semantic Web and the ideas behind the currently on-going Linked Data movement as a way to represent data semantics on the Web. Finally, we summarize our retrospective in Sect. 6 and give an outlook on future research topics in the area of semantics and interoperability research.

2 Early Database Models

Very early in the development of file systems and databases it was realized that a model-driven approach to data storage would allow a better separation between the data stores and the application programs using those data. In a way, this so-called data independence was a first step towards data-oriented interoperation of programs. At the same time this data independence brought explicit semantics into play in the sense that a data model reflected the real world and allowed programmers and end users to better share and understand the meaning of those data and therefore to utilize them more effectively.

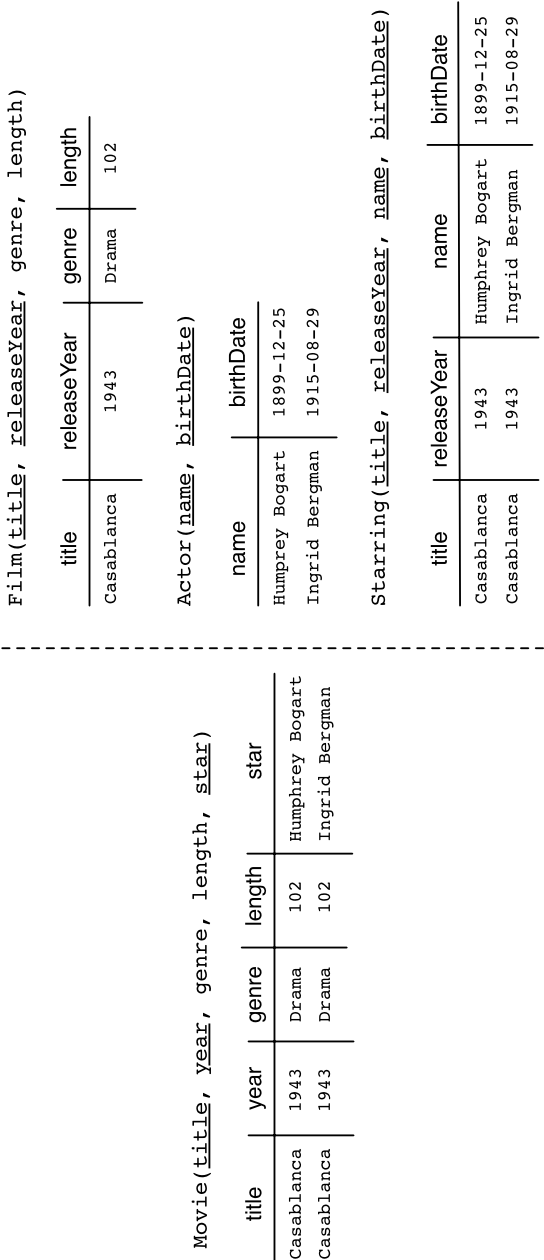
In this section, we first focus on the Relational Model (Sect. 2.1), which forms the current predominant formal basis for modern database systems. Then we describe the Entity Relationship Model (Sect. 2.2) and other related logical and conceptual data models (Sect. 2.3) from that period.

2.1 *The Relational Model*

In the 1970s, a large number of modeling approaches were proposed, and quite a number of them are still in use today. The Relational Model [19] had a seminal influence on this field because the simplicity of the table-oriented visualization allowed easy understanding and use of the data in a data storage independent way. Each row (tuple) in a relational table describes an entity with named attributes. With its keys, as identifiers, and normal forms (2nd, 3rd, Boyce-Codd etc.), representing functional dependencies, early examples of semantics, i.e., reflections on the properties of the real world, became expressible. Figure 3 shows our illustrative example represented in the Relational Model.

After the invention of the Relational Model, several efforts to develop languages for manipulating and retrieving data stored in relational database management systems (RDBMS) started. Initial proposals such as SEQUEL [15], which was developed by IBM, and QUEL, which was part of the INGRES effort [55], merged into the standardized Structured Query Language (SQL), which, until today, has remained the predominant data definition and manipulation language for RDBMS. The goal of the SQL standardization was to provide interoperability for applications so that they could access and manipulate data independently from the underlying RDBMS implementation. Today there are three different SQL standards (SQL, SQL2, SQL-99) in existence and several vendor-specific dialects, which is a major drawback from an interoperability perspective.

Soon it became clear that the Relational Model was too restrictive to allow for an easy expression of more sophisticated semantic situations that would be needed when designing databases for multiple applications and usage environments. As a consequence, semantically richer models were developed. One of the first conferences oriented strongly towards semantics was the IFIP TC 2 Working Conference on Database Management Systems held 1974 in Corsica. There, Abrial introduced the Binary Relational Model [4] by defining objects as models of concrete or abstract objects of the real world and binary relations between them. In doing so he



Sample movie data in Studio A

Sample movie data in Studio B

Fig. 3 Relational Model Sample. This example shows how *Studio A* and *B* could structure their data in relations. *Studio A* stores the information about movie and stars in a single relation, which can lead to data redundancies as well as update and deletion anomalies. *Studio B* decomposed its data into separate relations and thereby eliminates these shortcomings. The choice of keys in *B* causes a large data load in the *Starring* relation because the relationship between films and actors is established via their keys and foreign keys

introduced unique internal identifiers and showed that binary relations were sufficient to model the data-related properties of the real world. Semantics was expressed by object properties like synonyms, equivalence or relational symmetry, reflexivity and transitivity but also by handling three valued logic (true, false, unknown) to allow for an open world assumption. Today we can still find some of these concepts in the RDF model (see Sect. 5.1).

At the same conference, Sungren introduced his thesis [57] where he applied, for the first time, the Meta-Information concept for database models. This allows for the representation of even richer semantics about the real world modeled in the database including formal and informal information about objects, properties and relations. Another important aspect of metadata is information like quality of the data, changeability of the model, reliability of the information, the source of the data, etc. Metadata help the designer of the database to decide on the proper schema and help the user when locating relevant information in the database.

2.2 The Entity Relationship Model

In 1975, Chen published the Entity Relationship Model (see [17] and [18]). This model streamlined a number of the earlier approaches into the somewhat simpler to understand concepts of Entities, Attributes and Relations as the basic building blocks for modeling the real world. Again, the constraints placed on entities (e.g., cardinality, atomicity), relations (e.g., $n : m$) and attributes (single- or multi-valued, types) allow for the expression of semantics. The ER model gave rise to a series of conferences starting in 1979 and continuing up until today. The semantic modeling aspect for designing databases as well as the interoperability of programs using those databases was considered in the development and the extensions of the ER model. Up until about 1985 the ER model did not discuss, for instance, is-a and inheritance [23]. Figure 4 shows the Entity Relationship model for our illustrative example.

2.3 Other Models

Through the 1970s and to some degree since then, quite a number of additional models were proposed. The Object Role Model (ORM) originally proposed by Falkenberg [25] and Nijssen as NIAM [46] was later adopted for the ORM modeling technique which in turn influenced (e.g., Halpin [33]) the data modeling part of the nowadays predominant Unified Modeling Language (UML). Most of these models were developed to allow semantic-oriented design of databases and data independence. Interoperability aspects were only mentioned as borderline criteria.

That, however, changed with the Architecture Model of the ANSI/X3/SPARC proposal [5]. This model differentiates between three levels of database schemas:

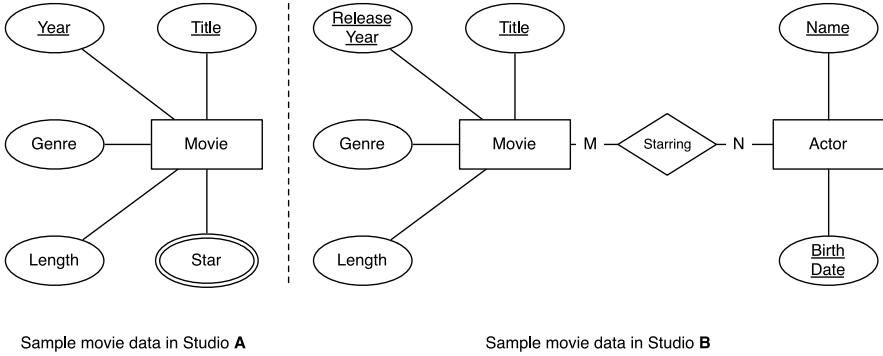


Fig. 4 Entity Relationship Model Sample. The example shows how *Studio A* and *B* could model their data structures using the Entity Relationship Model (in Chen’s original notation). *Studio A* models the names of the movie stars as *multi-valued* attributes (marked with double circles). *Studio B* models the associations between instances of movies and actors as a *relationship*. The underlined attributes indicate primary keys

an *internal model* (e.g., a relational model), a *conceptual model* (e.g., a global ER Model), and multiple external models representing the usage views of the database and reflecting the individual semantic needs of the usage in a heterogeneous interoperability environment. This immediately led to a number of research issues on how to map the different levels into each other without loss of essential information.

3 Distributed and Object-Oriented Database Systems

The powerful (relational) database systems developed in the 1970s ensured data independence and interoperability of application programs. At the same time, it was realized that more powerful data models were needed that expose more of the semantics of these data and allow application-oriented and context-dependent design of databases. In the late 1970s and early 1980s the rise of powerful computer networks began. It was henceforth possible to place data on various computer nodes, either locally or distributed throughout larger networks.

In this section, we first describe the research area of distributed databases and how they deal with semantic heterogeneities (Sect. 3.1). Then we introduce the central characteristics of object-oriented database models and systems (Sect. 3.2).

3.1 Distributed databases

In the late 1970s, the research field of distributed databases¹ grew rapidly in importance. Early papers on distributed databases were Distributed INGRES [56] and

¹See Ceri et al. [14] for an overview of distributed databases.

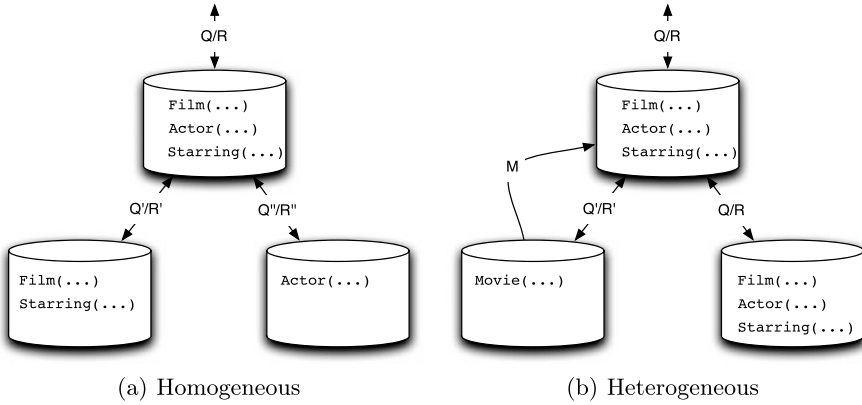


Fig. 5 Homogeneous and Heterogeneous Distributed Databases Sample. In (a) we assume that *Studio B* distributes the relations of its schema to two distinct database systems. In (b) the schema of *Studio B* serves as global schema and also as local export model of *B*'s database. A mapping *M* between the global schema and the local schema of database of *Studio A* needs to be established in order to bridge the heterogeneities between the involved databases

POREL [45], both approaches based on the Relational Data Model. They introduced the concept of global versus local schemas and the three-level architecture for centralized database systems, which was later extended to five layers: the (*multiple*) *local internal models*, the *local conceptual models*, the *local (conceptual) export models*, the *global (conceptual) model*, and the (*multiple*) *external models*. In order to design such a system, additional semantic meta-information was needed, as, for example, on the data distribution, the size and break-up of entity sets, the relations between them, the cardinality and selectivity of attributes, etc. The data models had to be extended accordingly, but in many cases those extensions were attached to an underlying relational model and not to the conceptual models of the various layers. The interoperability of applications and databases was then assured via the single global schema that would be used both by the local databases as well as by all of the global applications.

It was recognized that in principle two situations for distributed databases can exist: (i) *homogeneous* and (ii) *heterogeneous* distributed databases. Figure 5 shows how our illustrative example can be deployed in a distributed setting.

In the first case, a top-down design is realized by integrating external schemas into a single global schema. Guided by application-oriented metadata the design of the local schemas for the different computers in the network then follows. Here, considerable research effort was spent on strategies for splitting relations horizontally or vertically but, in retrospect, difficulties often arose from the low level of available semantic information. Some other research prototypes next to Distributed Ingres and POREL are SDD-1 of the Computer Corporation of America [37] and R* of IBM [31].

In the second case, heterogeneous systems follow a bottom-up design to cover situations where a number of pre-existing or autonomous databases must be inte-

grated into a single data management system in order to be shared by global applications. Using the information contained in the local conceptual schemas and the global knowledge about the applications, the export schemas can be developed and then be integrated into a single global schema by means of a mapping specification. The research prototype MULTIBASE [41] uses Daplex, a logical data specification language, for modeling the various schemas. Heterogeneous SIRIUS-DELTA [42] uses the Relational Model only and demonstrates the integration of PHLOX, which is a database system of the CODASYL Model family. However, it does not provide the equivalent functionality to databases as no real global schema is assumed, no local users are allowed, and mapping functions are to be provided by the local database management systems.

As it turns out, homogeneous distributed database systems have become a feature of the major database products, whereas heterogeneous systems are still difficult to handle, even today. The main problems arise from the scarcity of explicit semantics that can be provided for the external schemas and the global applications that use those schemas as well as the semantics for the local schemas used for designing the local databases.

With the advent of heterogeneous distributed database systems, the need for *model compatibility*, *data consistency* and *object identity* became apparent when interoperability was to be achieved. In our illustrative example, Studio B uses the attribute *BirthDate* as part of the primary key for the relation *Actors*. Studio A represents actors as a multi-valued attribute with the consequence that actors can only be identified by their names; information about an actor's birthdate is not available in Studio A's database. Therefore, Studio A cannot distinguish between actors having the same names and runs into problems when integrating its data with those of Studio B: if the schema of Studio B is used as global schema, it is not possible to define identity for the actors from Studio A's database, because birth dates are not given.

The models discussed so far neither allow for the specification of behavior nor are they flexible enough to allow for the expression of properties like equivalence, inheritance, and composition. As a consequence, the attention of the database research community shifted to object-oriented databases that allow for the specification of object identity, structures, semantics, behaviors, and constraints for the objects to be stored in the database as described in the following section.

3.2 Object-Oriented database Models and Systems

Even when main stream databases—the relational model based systems, whether central or distributed—were enhanced with Entity Relationship type semantic descriptions, they did not show enough flexibility to support, for instance, the interoperation of heterogeneous systems or the extensibility for new appearing data types like semistructured and unstructured information. BLOBs (Binary Large Objects) used as a first solution actually led to the loss of data independence, a paradigm that originally gave rise to the databases concept.

In the early 1980s, object-oriented programming (Smalltalk, C++) became popular and the need for the persistent storage of those new types of data arose. This triggered research in Object-Oriented Database Management Systems (OODBMS) and Object-Oriented Data Models (OODM), which started simultaneously in many locations. Many prototypes and even some commercial systems became available in the late 80s. An extensive description of those systems can be found in Dogac et al. [21] and also in Bukhres et al. [12].

Basically an object-oriented database model introduces application behavior (semantics) into databases by supporting a number of concepts, some of them well known in the object-oriented programming world, others specific to the persistence mechanism used for storing.

- *Object Identity*: every object has a unique identifier attached permanently at object creation time for object recognition. Unfortunately, this does not solve the object identity problem in heterogeneous systems where for the same real world object two different database objects could have been created.
- *Type Extensibility*: the basic data types in the database can be extended with new basic types and their handling functions. Type constructors would allow for new complex (abstract) data types. The typing systems could allow static binding or dynamic binding of data to the operations.
- *Object Classes*: real-world entities of the same kind, that is, those modeled as objects having the same data types, object attributes, behavior and relationships to other objects, can be collected into a single class.
- *Inheritance*: objects of a subclass (a more specific description) inherit properties of a superclass via the semantic concept of an is-a relationship including inheritance from multiple superclasses, e.g., as in case of the two superclasses SUV and Truck and the subclass SportTruck that has properties of both the SUV and Truck classes.
- *Object Instance*: some OODM allow that an object instance can populate all the superclasses it inherits properties from, others only allow the instance in the ultimate subclass where its most specific description is located. Missing information, later added, would change the class of an object whereas in the first case the object instance only would be added to the newly relevant subclass. The first case would also simplify the problem of interoperability in heterogeneous multi-OODBMSs. Of course it would still not solve the problem of object identity.

We believe that no single prototype or product fully supported all the possible features and also that no clear winner has ever been established in the OODBMS world. As it happens, object-oriented features were added by the relational database vendors as object-relational database management systems and today the “pure” OODBMS’s can only be found in niche application fields. A simple example of an OODBMS schema is given in Fig. 6.

To tackle the problem of heterogeneous distributed OODBMSs with their sometimes distinct formal semantics, more (formal) semantic flexibility was desirable. The VODAK Modeling Language (VML) [39] was an attempt to solve the problem by extending the two level models *Application Class* and *Instance* and the relationship *is-instance-of* with two additional levels, the *Meta Class (MC)* level and the

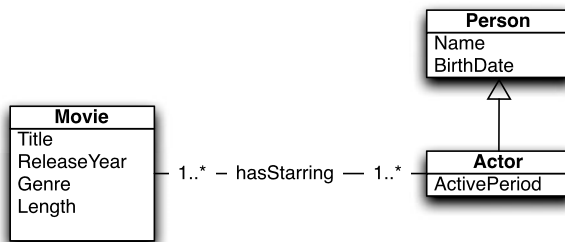


Fig. 6 Object-Oriented (UML) Model. The example shows the schema of *Studio B* in an object-oriented representation using the UML notation. To illustrate the inheritance feature of OO model, we introduced a superclass `Person` that defines all the attributes that would describe persons (not only actors) in the real world. The class `Actor` inherits all the properties from `Person` and introduces the additional attribute `ActivePeriod`

Meta-Meta Class (MMC) level (or Root Metaclass). The MC classes would specify the behavior of the specific *Class Model*, e.g. inheritance of all properties for all subclasses or only for specific properties or no inheritance at all could be specified. In case of heterogeneous OODBMSs, a global schema could then be used to integrate the individual different (formal) models and achieve interoperability between the databases. Today the idea of multi-level model architectures is reflected in the Object Management's Group (OMG) MOF model [48] and serves as formal basis for UML [49], which is now the de-facto standard for object-oriented application design.

However, as it soon turned out, even with the powerful object-oriented models, which allowed for the expression of many real-world semantic properties and behaviors, the expressive power needed in the growing world of multimedia and the World Wide Web was still missing. As a consequence, the OODBMSs never became *the* database concept envisioned in the late 1980s and early 1990s, despite the fact that some of their features can be found even today in multimedia, document, streaming, etc. data models.

4 Semantics in Distributed and Heterogeneous Information Systems

Distributed databases split data across several nodes and increased the performance and scalability in data management. The distinction between different types of schemas and the development of more application-oriented data models such as the Object-Oriented Data Model introduces novel ways of expressing data semantics. However, with the rapidly increasing size of local and wide area computer networks, those established database-oriented interoperability mechanisms turned out to be insufficient due to the technical heterogeneities of the involved network nodes.

In the late 1980s and early 90s *information integration* started to become an active research field having the goal to provide uniform access to data stored in distributed, heterogeneous, and autonomous systems. The Semistructured Data Model

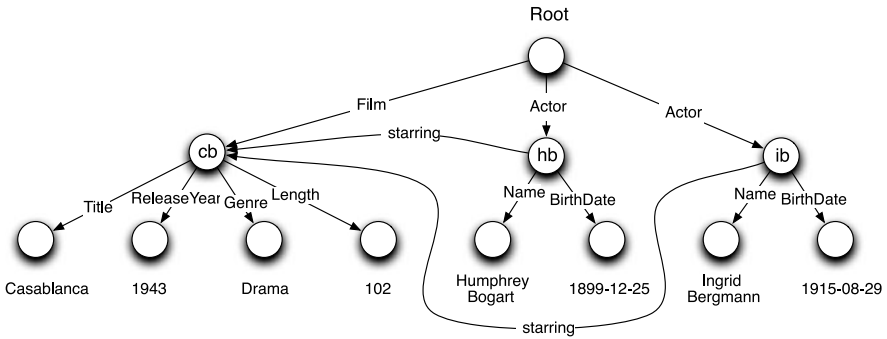


Fig. 7 Semistructured Model Example. A directed labeled graph represents the data of *Studio B*. The graph is self-describing because the data also carry schema information

(Sect. 4.1) plays a central role in this context. In parallel, research on Markup Languages (Sect. 4.2) evolved to a first agreed-upon standard (SGML), a derivative of which (XML) was later integrated with the Semistructured Data Model. Hyper-text and Hypermedia research (Sect. 4.3) focused not only on data and document representation, but also on navigation and access to documents in distributed environments. All these efforts had a direct impact on Multimedia Data and Document Models, which aimed at representing the semantics and behaviors of non-textual multimedia objects. As a representative for these developments we discuss MPEG-7 (Sect. 4.4) and briefly outline other metadata interoperability approaches (Sect. 4.5).

4.1 The Semistructured Data Model

In all models available so far (Relational Model, ER Model, OO Model), there has been a fixed schema describing the semantics of data. This leads to problems when data are exchanged across systems, because the underlying databases usually do not share the same schema even if they store similar data. This was the primary motivation for developing a more *flexible* data model, called the *Semistructured Data Model*.

The original model evolved from the LORE [3] and TSIMMIS [16] projects at Stanford University and was first described by Papakonstantinou et al. [51]. Unlike the other existing data models at that time, the semistructured model does not separate the schema from the data. It is *self-describing*, meaning that the data themselves carry their schema information. Data represented by the semistructured model takes the form of a directed labeled graph. The nodes in such a graph stand for objects or attribute values. An edge indicates the semantics of the relationship two nodes have with each other. Unlike previous models, an edge merges the notions of attributes and relationships into a single primitive. Figure 7 shows our illustrative example in a semistructured representation.

The semistructured data model provides the necessary flexibility for exchanging data across system boundaries. However, the price for this flexibility is the loss of efficiency in query processing. This is one of the reasons why most of today's data is still represented in the very efficient relational model and the technologies based on the semistructured model are primarily used for exchanging data. An architectural pattern combining the benefits of the static-schema and schema-less approaches is the mediator-wrapper architecture proposed by Wiederhold [63]. An extensive explanation of the Semistructured Data Model and its succeeding technologies is provided by Abitebul et al. [2].

4.2 Markup Languages

The motivation for the development of markup languages comes from the publishing industry and early works on electronic document management systems. Without any markup, documents are simply files containing a sequence of symbols. Applications processing these documents cannot anticipate, for instance, what are the section headings to be presented to the user or where in the character sequence the information about the authors is located. Therefore document exchange and consequently interoperability between applications and between vendors becomes very difficult. As a consequence, the goal of markup languages is to add explicit *semantics* to plain character sequences. Markers (tags) allow for the annotation of electronic documents in order to add data-, presentation-, and processing-semantics to character subsequences.

The IBM Generalized Markup Language (GML) [27] invented by Mosher, Lorie, and Goldfarb was the first technical realization of a markup language. Scribe [52] was the first language that introduced the separation of content and format and applied a grammar controlling the usage of descriptive markup elements. These works lead to the standardization of the Standard Generalized Markup Language (SGML) [35] in 1986. SGML is a *metalanguage* for describing markup languages and defines a common syntax for the markup or identification of structural textual units as well the grammar—the document type definition (DTD)—for defining the structure and allowed for tags in a document. Prominent derivatives of the SGML are HTML, which was developed in 1991, and XML, which was standardized in 1998.

HTML [7] is a presentation-oriented markup language that allows users to easily create Web sites without adhering to the strict formal requirements imposed by the SGML DTDs. In fact, it eliminates DTD's and only suggests structural and very few (formal) semantic features such as HTML META-Tags. Nevertheless, the extensibility and flexibility of HTML was one of the key factors for the success of the World Wide Web, with the result that today HTML is still the most widely used markup language.

While HTML mainly provides markup elements that define presentation semantics of document parts, XML [61] goes back to SGML, eliminates complex properties and streamlines DTD's. As a consequence, XML provides a simplified meta

```

<?xml version="1.0" encoding="UTF-8"?>
<movie>
  <title>Casablanca</title>
  <releaseYear>1946</releaseYear>
  <genre>Drama</genre>
  <length>102</length>
  <starring>
    <actor>
      <name>Humphrey Bogart</name>
      <birthDate>1899-12-25</birthDate>
    </actor>
    <actor>
      <name>Ingrid Bergman</name>
      <birthDate>1915-08-29</birthDate>
    </actor>
  </starring>
</movie>

```

Fig. 8 XML Document Example. It shows the movie data of *Studio B* represented in XML. The first line contains an XML processing instruction, the following lines the XML elements and values that describe the movie and its actors

markup language for defining documents that contain data to be communicated between applications. Since XML is backwards-compatible to SGML, DTDs can be applied for imposing element definitions and document structures on XML documents. The elements in XML documents indicate the semantics of contained data values. Nowadays, however, DTDs are superseded by the XML Schema, which offers the great advantage that not only data but also the schema information is represented in XML. Figure 8 shows our illustrative example represented in XML.

It was soon discovered that the freedom of the original HTML specification, as a presentation-oriented markup language, lead to semantic interoperability problems among web browsers and applications. Around the year 2000, XHTML was developed in order to bind the features of HTML to an XML format. The goal was to represent Web documents as well-formed XML documents, which promised greater interoperability but less freedom in the creation of Web sites. With the development of XHTML 2² and HTML 5³ a competition on the next generation markup language started. At the time of writing, HTML 5 seems to be the winner in the field of Web-markup languages because of its less strict, more evolutionary design approach. However, despite this development the expressibility of real-world semantics remained weak and led to the development of additional meta-languages such as RDF/S and OWL, which will be discussed in Sect. 5. HTML 5 now provides the possibility to include metadata content expressed in RDF in Web documents.

²<http://www.w3.org/TR/xhtml12/>.

³<http://www.whatwg.org/specs/web-apps/current-work/multipage/>.

4.3 *Hypertext and Hypermedia*

Inspired by Vannevar Bush's vision of Memex [13], Ted Nelson and Douglas Engelbart started their research on hypertext and hypermedia systems in the late 1960s (cf., [24, 44]). The goal of hypertext was to extend the traditional notion of linear *flat* text files by allowing a more complex organization of the material. Hypertext systems should allow direct machine-supported references from one textual chunk to another. Via dedicated interfaces, the user should have the ability to interact with these chunks and to establish new relationships between them.

Hypertext was considered as a non-linear extension of traditional text organization. In its simplest form, hypertext consists of nodes and plain links, which are just connections between two nodes. They carry no explicit semantics but simply serve for the navigation between documents or document chunks. But links can also be used to connect a comment or annotation to a text. In such a case, the links that connect data with other data express semantics. When links have explicit types assigned, as described in Trigg et al. [59], they explicitly define the semantic relationship between nodes. There is a clear analogy between explicitly typed links in hypertext systems and the semistructured model described in Sect. 4.1: the underlying models are directed labeled graphs.

Hypermedia is an extension of hypertext that also includes non-textual multimedia objects such as audio, video, and images. A detailed survey on early hypertext research and existing hypertext systems is available in [20]. However, hypermedia inherits the properties of hypertext and has also only limited means to express the semantics of the involved media objects and the relationships between them.

For achieving interoperability and exchanging hypertext and hypermedia documents between applications, it soon became clear that a standardized exchange format is required in order to provide interoperability. *HyTime*, as an extension of SGML, is an example for such a standard (see Goldfarb [28]). Also the work in the *Dexter Group* focused on hypertext exchange formats and architectural models that should facilitate the exchange of hypertext [29, 32].

The World Wide Web is the most popular hypertext application in use today. One of the success factors of the Web was that several technologies were integrated into an easy-to-use technology stack: Uniform Resource Identifiers (URIs) and Uniform Resource Locators (URLs) for addressing documents in the Web, HTML (and its extensions) as a flexible markup language for creating hypertext documents, and HTTP as a protocol for the communication between clients and servers (see [36]).

4.4 *The MPEG-7 Metadata Interoperability Framework*

With the release of the MPEG-7 standard in February 2002, a powerful metadata system for describing multimedia content was introduced. The goal was to provide higher flexibility in data management and interoperability of data resources. The difference between MPEG-7 and other already existing MPEG standards is that

MPEG-7 does not specify any coded representation of audio-visual information but focuses on the standardization of a common interface for describing multimedia materials [43]. MPEG-7 aims to avoid being a single monolithic system for multimedia description but rather an extensible metadata framework for describing audiovisual information.

MPEG-7 standardizes an extensive set of content *Descriptors (D)* and *Description Schemas (DS)* and offers a mechanism to specify new Description Schemas, such as the *Description Definition Language (DDL)*. It is a description standard for all kind of media (audio, image, video, graphics, etc.) and creates a common basis for describing different media types by a single standard. It thereby eases interoperability problems between media types as well as applications.

MPEG-7 uses XML for encoding content descriptions into a machine-readable format. XML Schema serves as the basis for the DDL that is used for the syntactic definition of the MPEG-7 description tools and that allows for extensibility of the description tools. Further details on MPEG-7 are available in Kosch [40].

MPEG-7 was not developed with a restricted application domain in mind. With the ability to define media description schemas by means of the DDL, MPEG-7 is intended to be applicable to a wide range of multimedia applications ranging from home entertainment (e.g., personal multimedia collections) to cultural services (e.g., art galleries) and surveillance (e.g., traffic control). However, this wide application spectrum has resulted in an enormous complexity of that standard, which, in our opinion, is one of the reasons why the ambitious goals of MPEG-7 remain unreached.

4.5 Other Metadata Interoperability Approaches

The relevant characteristics of the 1990s are the emergence of the World Wide Web and an increasing need for interoperability among distributed applications. The availability of markup languages such as XML promoted the development of metadata interoperability standards that should allow the exchange of information objects across system boundaries. These standards ranged from rather generally-applicable schemas such as Dublin Core [22] to very domain-specific schemas such as ONIX [58], which provides standardization for the publishing industry.

This was also the period when global models covering the semantics of whole application domains emerged. Those models are supposed to define the common notions used in a domain and serve as a global schema for the integration of data in a heterogeneous distributed environment. The CIDOC CRM⁴ model, for instance, is such a model. It defines a conceptual model that aims at providing interoperability among information systems in cultural heritage institutions. This is architecturally similar to the idea of heterogeneous databases (cf., Sect. 3.1) where a global schema defines the model primitives for querying the underlying databases. The difference

⁴<http://cidoc.ics.forth.gr/>.

from the 1990s onwards is that global model interoperability approaches are being applied in the Web, which is an open-world environment. However, they inherit the problems distributed databases can only cope with in their much smaller closed-world environment. As in databases, one must always deal with semantic ambiguities in the interpretations of the involved schemas and provide adequate mappings to bridge the heterogeneities.

For a more detailed discussion on techniques for achieving metadata interoperability, we refer to a recent survey provided by Haslhofer and Klas [34].

5 The Semantic Web and Linked Data

The late 1990s were characterized by the success of the World Wide Web. A set of simple-to-use technologies (URI, HTTP, HTML) suddenly allowed also non-technical users to easily create and publish documents in a globally accessible information space. This was one of the reasons for the rapid spread of the Web. However, the information published on the Web was intended for human consumption and not for machine-interpretation. This motivated the development of the Semantic Web, which is an extension of the existing Web and has the goal to use the Web as

a universal medium for the exchange of data. The Web should become a place where data can be shared and processed by automated tools as well as by people.⁵

Section 5.1 focuses on early Semantic Web activities and briefly describe the major specifications in place. Section 5.2 summarizes current activities in the area of Linked Data.

5.1 The Semantic Web

The term *Semantic Web* was coined by Tim Berners-Lee and popularized in an article published in Scientific American in 2001 [8]. There the Semantic Web is described as *a new form of Web content that is meaningful to computers and will unleash a revolution in new possibilities*. In the early Semantic Web vision *intelligent agents* should act on behalf of their users and automatically fulfill tasks in the Web (e.g., making a doctor's appointment). This of course requires that these agents understand the *semantics* of the information exposed on the Web.

Based on this vision, the *Semantic Web Activity* was started at the W3C and has lead to the specification of several standards that technically enable this described vision: RDF/S, OWL, OWL-S, SKOS, and SPARQL.

Since one of the major design principles was to build the Semantic Web upon the existing Web architecture, URIs form the basis for all these standards. Hence,

⁵<http://www.w3.org/2001/sw/Activity>.

all resources in the Semantic Web—including, but not limited to, those describing real-world objects—should have URIs assigned.

The Resource Description Framework (RDF) serves as data model for representing metadata *about* a certain resource. It allows us to formulate statements about resources, each *statement* consisting of a subject, a predicate, and an object. The subject and predicate in a statement must always be resources, the object can either be a resource or a literal node. A statement is represented as a *triple* and several statements form a *graph*. RDF data can be exchanged between applications by serializing graphs using one of the RDF serialization syntaxes (e.g., RDF/XML, N-Triples, Turtle). We will give an example of an RDF graph in Fig. 9 in Sect. 5.2.

The *RDF Vocabulary Description Language RDF Schema (RDFS)* and the *Web Ontology Language (OWL)* are means of describing the vocabulary terms used in an RDF model. RDFS provides the basic constructs for describing classes and properties and allows for their arrangement into simple subsumption hierarchies. Since the expressiveness of RDFS is limited and misses some fundamental modeling features often required to construct vocabularies, the *Web Ontology Language (OWL)* was created. It is based on RDFS and allows the distinction between attribute-like (`owl:DatatypeProperty`) and relationship-like (`owl:ObjectProperty`) properties and provides several other expressive modeling primitives (e.g., class union and intersections, cardinality restrictions on properties, etc.) that allow us to express more complex models, which are then called ontologies. With RDFS and OWL one has the possibility to define models that explicitly express the semantics and specify and process possible inferences of data. The formal grounding of OWL (Description Logics) allows applications to reason on RDF statements and infer logical consequences. The binding of semantics to a logical system reduces interpretation ambiguities and leads to greater semantic interoperability between applications.

OWL-S Semantic Markup for Web Services is a specific upper-level ontology for the description of services on the Web (Semantic Web Services). It should enable automatic Web service discovery and invocation by Web agents as well as automatic Web service composition and interoperation. Therefore, OWL-S can be considered as an attempt to establish interoperability between services in the Semantic Web.

The *Simple Knowledge Organization System (SKOS)* is a model for expressing the structure and constituents of concept schemas (thesauri, controlled vocabularies, taxonomies, etc.) in RDF so that they become machine-readable and exchangeable between applications. With SKOS one can attach multi-lingual labels to concepts and arrange them in two major kinds of semantic relationships: broader and narrower relationships for constructing concept hierarchies and associative relationships for linking semantically related concepts.

The *SPARQL Query Language for RDF* is an expressive language for formulating structured queries over RDF data sources. It defines a protocol for sending queries from clients to an SPARQL endpoint and for retrieving the retrieved results via the Web. Currently, the abstract protocol specification has bindings for HTTP and SOAP. This allows clients to execute a query against a given endpoint (e.g., <http://dbpedia.org/sparql>) and to retrieve the result set through common Web transport protocols. The underlying motivation for defining the SPARQL query language

is analogous to the motivation for defining SQL (see Sect. 2.1): to be able to access RDF stores via a uniform interface in order to achieve greater interoperability.

A core belief of the early Semantic Web was that intelligent agents should be able to reason and draw conclusions based on the available data. This is why, in the Semantic Web, the meaning of terminology used in Web documents, that is the *semantics* of data, is expressed in terms of ontologies. The term *ontology* has its technical origin in the Artificial Intelligence domain and is defined as a specification of a conceptualization (see e.g., [30]). In its core, an ontology is similar to a database schema: a model defining the structure and semantics of data. Noy and Klein [47] describe several features that distinguish ontologies from database schema, most importantly that ontologies are logical systems that define a set of axioms that enable automated reasoning over a set of given facts.

Although intensive research has been conducted in the Semantic Web domain over the last ten years, this early vision of the Semantic Web has not been implemented yet. The limitations of the Semantic Web lie in formal issues like decidability and computational complexity but also in its conceptual complexity. It is difficult to make it clear to the user (and the system designers) what the inferences implied by a given fact are.

5.2 *Linked Data*

In 2006 Tim Berners-Lee proposed the so-called *Linked Data principles* [6] as a guideline or recommended best practice to share structured data on the Web and to connect related data that were not linked before. These are:

1. Use URIs to identify things.
2. Use HTTP URIs so that people can look up those names.
3. When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL).
4. Include links to other URIs, so that one can discover more things.

These principles eclipse the reasoning part of the Semantic Web, accentuate the data-centric aspects of existing Semantic Web technologies and thereby demystify their application in real-world environments. A central point in the Linked Data principles is the application of HTTP URIs as an object (resource) identification mechanism. When an application dereferences such a URI it receives data expressed in RDF. Structured access to RDF data within data sources is provided by SPARQL. This, in fact, resembles the central features provided by traditional (relational) database systems. The goal of the fourth principle is to interlink semantically related resources on the Web. If, for instance, two studios maintain a data record about the same movie, they should be interlinked. The semantics of the link depends on the application scenario; existing Semantic Web languages provide a set of pre-defined properties (`rdfs:seeAlso`, `owl:sameAs`, `skos:closeMatch`, etc.) for defining the meaning of links. Figure 9 shows how our illustrative example is represented in the Web of Data.

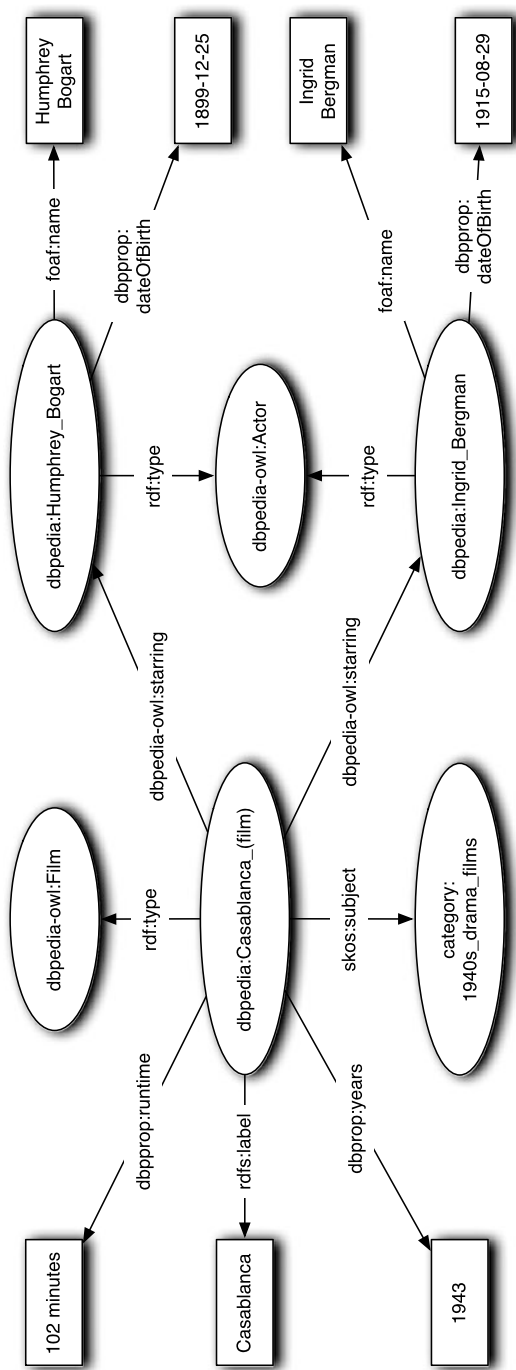


Fig. 9 Linked Data Example. The example how the data of Studio B could be exposed on the Web following the Linked Data guidelines. The prefixes expand as follows: dbpedia to <http://dbpedia.org/resource>, dbpprop to <http://dbpedia.org/property>, dbpedia-owl to <http://dbpedia.org/ontology/>, and category to <http://dbpedia.org/resource/Category>:

The Linked Data idea rapidly raised interest in various communities. Shortly after the formation of the W3C Linking Open Data Community project,⁶ DBpedia [11] was launched as the first large linked data set on the Web. It exposes all the information available in Wikipedia in a structured form and provides links to related information in other data sources such as the *Linked Movie Database*.⁷ As of November 2009, the DBpedia knowledge base describes more than 2.9 million things such as persons, music albums, or films in 91 different languages. It provides a user-generated knowledge organization system comprising of approximately 415 000 categories and millions of links to semantically related resources on the Web.

After DBpedia, many other data sources followed. Today this so-called Web of Data comprises an estimated number of 4.7 billion RDF triples and 142 million RDF links [10]. For data consumers this has the advantage that data as well as schema information is now available on the Web (see *The Best Practice Recipes for Publishing Vocabularies*⁸) and can easily be accessed via widely accepted Web technologies, such as URI and HTTP. RDF simply serves as a model for representing data on the Web. This pragmatic Web of Data principles also resembles the notion of *dataspaces* [26] that was coined in the database community.

However, as with all of the previously described interoperability attempts and technologies, Linked Data does not solve the complete stack of interoperability problems either. From the Semantic Web it inherits a set of technologies (RDF/S, OWL, etc.) that provide the necessary technical and structural interoperability, which in turn makes data easily accessible on the Web. But this does not solve the semantic interoperability problem. The data in the Linked Data Web are still heterogeneous because they use different vocabularies to describe the same real-world entities or the same vocabulary to describe different real-world entities. This leads to interpretation conflicts and usually requires manual interventions in terms of mappings. Although there exists a wealth of work in the area of semantic mediation and mapping (see e.g., [38]), the complexity of finding potential mappings between concepts grows with the size of the involved vocabularies. A fully automatic matching is considered to be an AI-complete problem, that is, as hard as reproducing human intelligence [9].

6 Summary and Future Research Directions

Interoperability is a qualitative property of computing infrastructures. It enables a receiving system to properly interpret the information objects received from a sender and vice versa. Since this is not given by default, the representation of semantics has been an active research topic for four decades.

⁶<http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData>.

⁷<http://www.imdb.com/>.

⁸<http://www.w3.org/TR/swbp-vocab-pub/>.

In this chapter, we gave a retrospective on semantics and interoperability research as applied in major areas of computer science. We started with the Relational Model developed in the 1970s and ended with the currently on-going activities in the Semantic Web and Linked Data community. The technical outcome of all these activities were *models* that allow for the expression of data semantics and system architectures for the integration of data from several (heterogeneous) sources. From the late 1990s on, when research was driven by the evolving World Wide Web, the semistructured data model gained importance. Different from previous models, it is self-describing, meaning that data itself carries schema information.

In essence, all presented models and system architectures enable the representation of data and the description of the semantics of these data. If one and the same model were used for exchanging information objects, interoperability would be established at least on a technical level and to some extent also on a syntactic and structural level. The Web is a good example of that; it provides a uniform way of identifying resources, a common exchange protocol, and a simple standardized markup language.

If the involved parties also agree on the semantics of terms, as it is the goal of the various metadata standardization attempts, interoperability can also be established on a semantic level. In practice, however, such an agreement is hard to achieve, especially when multiple parties from a broad range of application domains are involved. We can observe numerous attempts of defining general (ontology) models for a complete domain (e.g., MPEG-7 for multimedia metadata, CIDOC CRM for the cultural heritage domain); although they provide a very detailed domain description, they hardly found their implementation in practice. As long as people are the designers of models, different conceptions and interpretations will always exist, even for superficially homogeneous domains and application contexts. We therefore believe that research in the area of semantic interoperability should take this situation into account and find solutions that deal with a multitude of models and allow for their semi-automatic or manual reconciliation.

We believe that the World Wide Web will continue to be the predominant area for semantics and interoperability research. Applications that were available on the Desktop before (e.g., Email, Calendar, Office Suites, etc.) are now on the Web. A more Web-centric solution for data management is, in our opinion, a logical consequence. The Linked Data movement is definitely an important starting point in this direction. However, it will require further research on the integration of existing data sources and the development of scalable graph-based data stores. Additionally, since data are exposed on the Web and they should in the end, also be consumable by machines, further research must be conducted in the areas of data quality, changeability of models, reliability of information, and data provenance. In fact, these research topics were already identified in the early years of database research. Now, however, the open, distributed, and uncontrolled nature of the Web calls for a review of these approaches and possibly their adaption to a Web-based environment.

The evolution of schemas and ontologies in decentralized semantic structures such as the World Wide Web also calls for further research. Aberer et al. [1] coined the term *Emergent Semantics*, which denotes a research field focusing on the under-

standing of semantics by investigating the relationships between syntactic structures using social networking concepts for the necessary human interpretations.

References

1. Aberer, K., Catarci, T., Cudré-Mauroux, P., Dillon, T., Grimm, S., Hacid, M.-S., Illarramendi, A., Jarrar, M., Kashyap, V., Mecella, M., Mena, E., Scannapieco, M., Saltor, F., Santis, L.D., Spaccapietra, S., Staab, S., Studer, R., Troyer, O.D.: Emergent semantics systems. In: In International Conference on Semantics of a Networked World (ICSNW), pp. 14–43 (2004)
2. Abiteboul, S., Buneman, P., Suciu, D.: Data on the Web: From Relations to Semistructured Data and XML. Morgan Kaufmann, San Mateo (1999).
3. Abiteboul, S., Quass, D., McHugh, J., Widom, J., Wiener, J.L.: The lorel query language for semistructured data. *Int. J. Digit. Libr.* **1**(1), 68–88 (1997)
4. Abrial, J.-R.: Data semantics. In: Klimbie, J.W., Koffeman, K.L. (eds.) *Data Base Management*, pp. 1–60. North-Holland, Amsterdam (1974)
5. ANSI/X3/SPARC Study Group on Data Base Management Systems: Interim report. *FDT—Bulletin of ACM SIGMOD* **7**(2), 1–140 (1975)
6. Berners-Lee, T.: Linked Data. World Wide Web Consortium, (2006). World Wide Web Consortium. Available at <http://www.w3.org/DesignIssues/LinkedData.html>
7. Berners-Lee, T., Conolly, D.: RFC 1866—Hypertext Markup Language—2.0. Network Working Group (1995)
8. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. *Scientific American* (May 2001)
9. Bernstein, P.A., Melnik, S., Petropoulos, M., Quix, C.: Industrial-strength schema matching. *SIGMOD Rec.* **33**(4), 38–43 (2004)
10. Bizer, C., Heath, T., Berners-Lee, T.: Linked data—the story so far. *Int. J. Semant. Web Inf. Systems (IJSWIS)* **5**(3) (2009)
11. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia—a crystallization point for the web of data. *J. Web Semant.* **7**(3), 154–165 (2009)
12. Bukhres, O.A., Elmagarmid, A.K. (eds.): *Object-Oriented Multidatabase Systems: A Solution for Advanced Applications*. Prentice Hall, New York (1996)
13. Bush, V.: As we may think. *Atlantic Monthly* **176**(1), 101–108 (1945)
14. Ceri, S., Pelagatti, G.: *Distributed Databases: Principles and Systems*. McGraw-Hill, New York (1984)
15. Chamberlin, D.D., Boyce, R.F.: Sequel: a structured English query language. In: *SIGFIDET '74: Proceedings of the 1974 ACM SIGFIDET (now SIGMOD) Workshop on Data Description, Access and Control*, pp. 249–264. ACM, New York (1974). doi:[10.1145/800296.811515](https://doi.org/10.1145/800296.811515)
16. Chawathe, S., Garcia-Molina, H., Hammer, J., Ireland, K., Papakonstantinou, Y., Ullman, J.D., Widom, J.: The TSIMMIS project: integration of heterogeneous information sources. In: *16th Meeting of the Information Processing Society of Japan, Tokyo, Japan*, pp. 7–18 (1994)
17. Chen, P.P.: The entity-relationship model: toward a unified view of data. In: Kerr, D.S. (ed.) *VLDB*, p. 173. ACM, New York (1975)
18. Chen, P.P.: The entity-relationship model—toward a unified view of data. *ACM Trans. Database Syst.* **1**(1), 9–36 (1976)
19. Codd, E.F.: A relational model of data for large shared data banks. *Commun. ACM* **13**(6), 377–387 (1970)
20. Conklin, J.: Hypertext: an introduction and survey. *Computer* **20**(9), 17–41 (1987). doi:[10.1109/MC.1987.1663693](https://doi.org/10.1109/MC.1987.1663693)
21. Dogac, A., Özsu, M.T., Biliris, A., Sellis, T.K. (eds.): *Advances in Object-Oriented Database Systems, Proceedings of the NATO Advanced Study Institute on Object-Oriented Database Systems, Held in Izmir, Kusadasi, Turkey, August 6–16, 1993*. NATO ASI Series F: Computing and Systems Sciences, vol. 130 (1994)

22. Dublin Core Metadata Initiative. Dublin Core Metadata Element Set, version 1.1. Available at: <http://dublincore.org/documents/dces/> (December 2006)
23. Elmasri, R., Weeldreyer, J., Hevner, A.: The category concept: an extension to the entity-relationship model. *Data Knowl. Eng.* **1**(1), 75–116 (1985). doi:[10.1016/0169-023X\(85\)90027-8](https://doi.org/10.1016/0169-023X(85)90027-8)
24. Engelbart, D.C.: Augmenting Human Intellect: A Conceptual Framework. Stanford Research Institute, Menlo Park (1962)
25. Falkenberg, E.D.: Concepts for modelling information. In: Nijssen, G.M. (ed.) *IFIP Working Conference on Modelling in Data Base Management Systems*, Freudenstadt, Germany, pp. 95–109. North-Holland, Amsterdam (1976)
26. Franklin, M., Halevy, A., Maier, D.: From databases to dataspace: a new abstraction for information management. *SIGMOD Rec.* **34**(4), 27–33 (2005). doi:[10.1145/1107499.1107502](https://doi.org/10.1145/1107499.1107502)
27. Goldfarb, C.F.: A generalized approach to document markup. In: *Proceedings of the ACM SIGPLAN SIGOA Symposium on Text Manipulation*, pp. 68–73. ACM, New York (1981). doi:[10.1145/800209.806456](https://doi.org/10.1145/800209.806456)
28. Goldfarb, C.F.: Standards-HyTime: a standard for structured hypermedia interchange. *Computer* **24**(8), 81–84 (1991). doi:[10.1109/2.84880](https://doi.org/10.1109/2.84880)
29. Grønbaek, K., Trigg, R.H.: Hypermedia system design applying the dexter model. *Commun. ACM* **37**(2), 26–29 (1994). doi:[10.1145/175235.175236](https://doi.org/10.1145/175235.175236)
30. Gruber, T.: A translation approach to portable ontology specifications. *Knowl. Acquis.* **5**, 199–220 (1993)
31. Haas, L.M., Selinger, P.G., Bertino, E., Daniels, D., Lindsay, B.G., Lohman, G.M., Masunaga, Y., Mohan, C., Ng, P., Wilms, P.F., Yost, R.A.: R*: A research project on distributed relational DBMS. *IEEE Database Eng. Bull.* **5**(4), 28–32 (1982)
32. Halasz, F., Schwartz, M.: The dexter hypertext reference model. *Commun. ACM* **37**(2), 30–39 (1994). doi:[10.1145/175235.175237](https://doi.org/10.1145/175235.175237)
33. Halpin, T.: Object-role modeling (ORM/NIAM). In: *Handbook on Architectures of Information Systems*, pp. 81–102. Springer, Berlin (1998)
34. Haslhofer, B., Klas, W.: A survey of techniques for achieving metadata interoperability. *ACM Comput. Surv.* **42**(2) (2010)
35. ISO JTC1 SC34. ISO 8879:1986 Information Processing—Text and Office Systems—Standard Generalized Markup Language (SGML) (1986)
36. Jacobs, I., Walsh, N.: Architecture of the World Wide Web, Volume One. Available at: <http://www.w3.org/TR/webarch/> (December 2004)
37. Rothnie, J.B. Jr., Bernstein, P.A., Fox, S., Goodman, N., Hammer, M., Landers, T.A., Reeve, C.L., Shipman, D.W., Wong, E.: Introduction to a system for distributed databases (sdd-1). *ACM Trans. Database Syst.* **5**(1), 1–17 (1980)
38. Kalfoglou, Y., Schorlemmer, M.: Ontology mapping: the state of the art. *Knowl. Eng. Rev.* **18**(1), 1–31 (2003). doi:[10.1017/S0269888903000651](https://doi.org/10.1017/S0269888903000651)
39. Klas, W., Aberer, K., Neuhold, E.J.: Object-oriented modeling for hypermedia systems using the VODAK model language. In: *NATO ASI OODBS*, pp. 389–433 (1993)
40. Kosch, H.: *Distributed Multimedia Database Technologies Supported MPEG-7 and by MPEG-21*. CRC Press LLC, Boca Raton (2003)
41. Landers, T.A., Rosenberg, R.: An overview of multibase. In: *DDB*, pp. 153–184 (1982)
42. Litwin, W., Boudenant, J., Esculier, C., Ferrier, A., Glorieux, A.M., Chimia, J.L., Kabbaj, K., Moulinoux, C., Rolin, P., Stangret, C.: Sirius system for distributed data management. In: *DDB*, pp. 311–366 (1982)
43. Nack, F., Lindsay, A.T.: Everything you wanted to know about MPEG-7: Part 1. *IEEE Multi-Media* **6**(3), 65–77 (1999)
44. Nelson, T.H.: Complex information processing: a file structure for the complex, the changing and the indeterminate. In: *Proceedings of the 1965 20th National Conference*, pp. 84–100. ACM, New York (1965). doi:[10.1145/800197.806036](https://doi.org/10.1145/800197.806036)
45. Neuhold, E.J., Biller, H.: Porel: A distributed data base on an inhomogeneous computer network. In: *VLDB*, pp. 380–395. IEEE Computer Society, Los Alamitos (1977)

46. Nijssen, G.M.: Current issues in conceptual schema concepts. In: Nijssen, G.M. (ed.) Proc. 1977 IFIP Working Conf. on Modelling in Data Base Management Systems, Nice, France, pp. 31–66. North-Holland, Amsterdam (1977)
47. Noy, N.F., Klein, M.: Ontology evolution: Not the same as schema evolution. *Knowl. Inf. Syst.* **6**(4), 428–440 (2004). doi:[10.1007/s10115-003-0137-2](https://doi.org/10.1007/s10115-003-0137-2)
48. Object Management Group (OMG). Meta Object Facility (MOF) core specification—version 2.0. Available at: <http://www.omg.org/spec/MOF/2.0/PDF/> (January 2006)
49. Object Management Group (OMG). Unified Modelling Language (UML). Available at: <http://www.uml.org/> (2007)
50. Ouksel, A.M., Sheth, A.: Semantic interoperability in global information systems. *SIGMOD Rec.* **28**(1), 5–12 (1999). doi:[10.1145/309844.309849](https://doi.org/10.1145/309844.309849)
51. Papakonstantinou, Y., Garcia-Molina, H., Widom, J.: Object exchange across heterogeneous information sources. In: Eleventh International Conference on Data Engineering (ICDE 1995), pp. 251–260 (1995)
52. Reid, B.K.: A high-level approach to computer document formatting. In: POPL '80: Proceedings of the 7th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, pp. 24–31. ACM, New York (1980). doi:[10.1145/567446.567449](https://doi.org/10.1145/567446.567449)
53. Rowley, J.: The wisdom hierarchy: representations of the DIKW hierarchy. *J. Inf. Sci.* **33**(2), 163–180 (2007). doi:[10.1177/0165551506070706](https://doi.org/10.1177/0165551506070706)
54. Sheth, A.P., Larson, J.A.: Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Comput. Surv.* **22**(3), 183–236 (1990). doi:[10.1145/96602.96604](https://doi.org/10.1145/96602.96604)
55. Stonebraker, M., Held, G., Wong, E., Kreps, P.: The design and implementation of INGRES. *ACM Trans. Database Syst.* **1**(3), 189–222 (1976). doi:[10.1145/320473.320476](https://doi.org/10.1145/320473.320476)
56. Stonebraker, M., Neuhold, E.J.: A distributed database version of INGRES. In: Berkeley Workshop, pp. 19–36 (1977)
57. Sundgren, B.: An infological approach to data bases. PhD thesis, University of Stockholm (1973)
58. The EDItEUR Group: Online Information Exchange (ONIX). Available at: <http://www.editeur.org/onix.html> (2007)
59. Trigg, R.H., Weiser, M.: Textnet: a network-based approach to text handling. *ACM Trans. Inf. Syst.* **4**(1), 1–23 (1986). doi:[10.1145/5401.5402](https://doi.org/10.1145/5401.5402)
60. Visser, P.R.S., Jones, D.M., Bench-Capon, T.J.M., Shave, M.J.R.: An analysis of ontological mismatches: Heterogeneity versus interoperability. In: AAAI 1997 Spring Symposium on Ontological Engineering, Stanford University, Stanford (1997)
61. W3C XML Activity. Extensible Markup Language (XML) 1.0. W3C. Available at: <http://www.w3.org/TR/1998/REC-xml-19980210> (1998)
62. Wache, H.: Semantische Mediation für heterogene Informationsquellen. PhD thesis, University of Bremen (2003)
63. Wiederhold, G.: Mediators in the architecture of future information systems. *Computer* **25**(3), 38–49 (1992). doi:[10.1109/2.121508](https://doi.org/10.1109/2.121508)

Foundations for the Web of Information and Services

A Review of 20 Years of Semantic Web Research

Fensel, D. (Ed.)

2011, XXIV, 341 p., Hardcover

ISBN: 978-3-642-19796-3