A Retrospective on Semantics and Interoperability Research

Bernhard Haslhofer and Erich J. Neuhold

Department of Distributed and Multimedia Systems,
University of Vienna, Liebiggasse 4/3-4, 1010 Vienna, Austria
{firstname.lastname}@univie.ac.at

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 it can process them correctly. In this context, we often find the distinction between *data*, *information*, and *knowledge*, which has been subject of intensive discussions in the Information Science literature for years. For a more comprehensive and actual discussion on these term we refer to Rowley [49]. Here we simply 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 an 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 was 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 programs, 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, 46, 55, 57]). 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 syntax 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; now they decide 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 can assume 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 do in the subsequent sections.

The goal and major contribution of this paper 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.

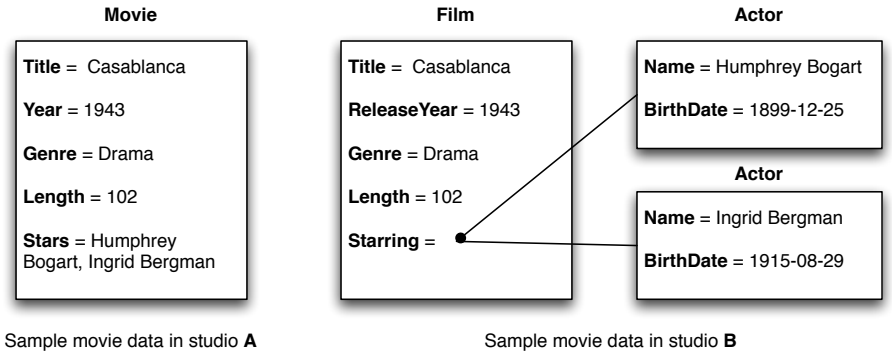


Fig. 1. Illustrative Example. Studio A records for each `Movie` its `title`, the `year` when it was first presented, the `genre`, its `length`, and the `stars` playing in the movie. Studio B records for each `Film` the `title`, the `releaseYear`, the `genre`, the `length`, and for each `starring` the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

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 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.

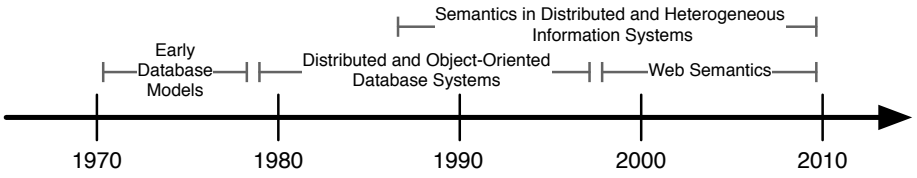


Fig. 2. Semantics and Interoperability Research in Computer Science.

As illustrated in Figure 2, we start our retrospective in the early 70s and present early database models in Section 2. Then, in Section 3, we move along to distributed databases and object-oriented database models, which allow application-oriented and context-dependent design of databases. In Section 4, we describe major models and languages for the representation of semantics in distributed and heterogeneous information systems. Then, in Section 5, we describe the Semantic Web and the ideas behind the currently ongoing Linked Data movement as a way to represent data semantics on the Web. Finally, we summarize our retrospective in Section 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 to allow for data-oriented interoperation of programs. At the same time this data independence brought semantics into play, in the sense that a data model reflected on the real world and allowed programmers and end users to better understand the meaning of those data and therefore to utilize them more effectively.

In this section, we will first focus on the Relational Model (Section 2.1), which, until today, builds the formal basis for modern database systems. Then we describe the Entity Relationship Model (Section 2.2) and other logical and conceptual data models (Section 2.3) from that period.

2.1 The Relational Model

In the 70s a large number of modeling approaches were proposed, quite a number of them still in use today. A seminal influence on this field had the Relational Model [17] since the simplicity of the table-oriented visualization allowed easy understanding and use of the data in a data storage independent way. With its keys and normal forms (2nd, 3rd, Boyce-Codd etc.) 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.

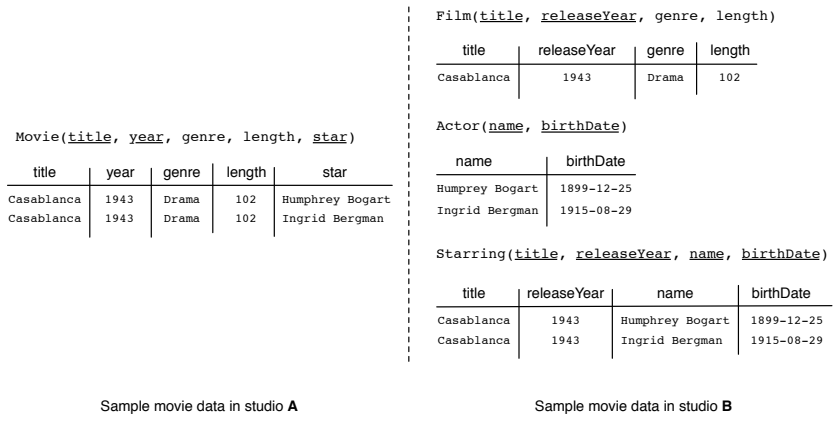


Fig. 3. Relational Model Sample. It shows how studio A and B could structure their data in relations. Studio A stores the information about movie and stars in a single relation, which can lead to data redundancies as well as update and deletion anomalies. Studio B decomposed its data into separate relations 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.

However, it soon became clear that the Relational Model was too restrictive to allow 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 being 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 J.R. 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 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 Section 5.1).

In the same conference, Bo Sungren introduced his thesis [52] 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 meta-data is information like quality of the data, changeability of the model, reliability of the information, the source of the data, e.t.c. Meta-data help the designer of the database to decide on the proper schema and the user when locating relevant information in the database.

2.2 The Entity Relationship Model

In 1975 Peter Chen published the Entity Relationship Model (see [15] and [16]). 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-multi valued, type) 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, for instance, not discuss is-a and inheritance. Figure 4 shows the Entity Relationship model for our illustrative example.

2.3 Other Models

Over the 70s but even later quite a number of additional models were proposed. The Object Role Model (ORM) originally proposed by Falkenberg [22] and Sjir Njissen as NIAM [42] was later adopted for the ORM modeling technique which in turn influenced (e.g., Halpin [30])) the data modeling part of the nowadays

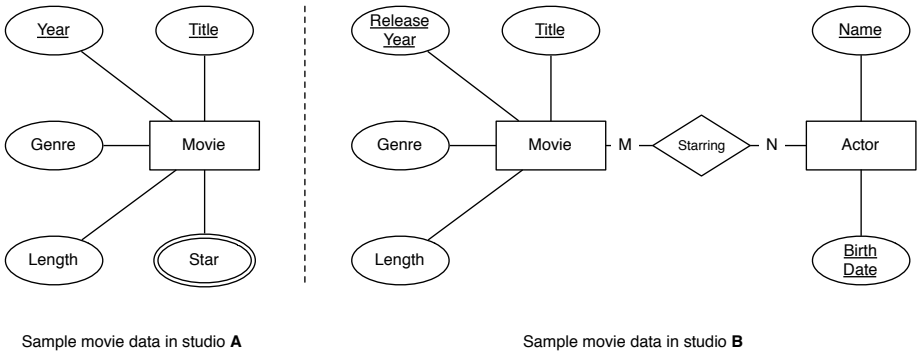


Fig. 4. Entity Relationship Model Sample. The example shows how studio A and B could model their data structures using the Entity Relationship Model (in original Chen notation). Studio A models the names of the movie stars as *multivalued* attributes (marked with double circles). Studio B models the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

predominant Unified Modeling Language (UML). Mostly all these models were developed to allow semantic-oriented design of databases and data independence respectively. 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: 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. This immediately led to a number of issues on how to map the different levels into each other without loss of essential information.

3 Distributed and Object-Oriented Database Systems

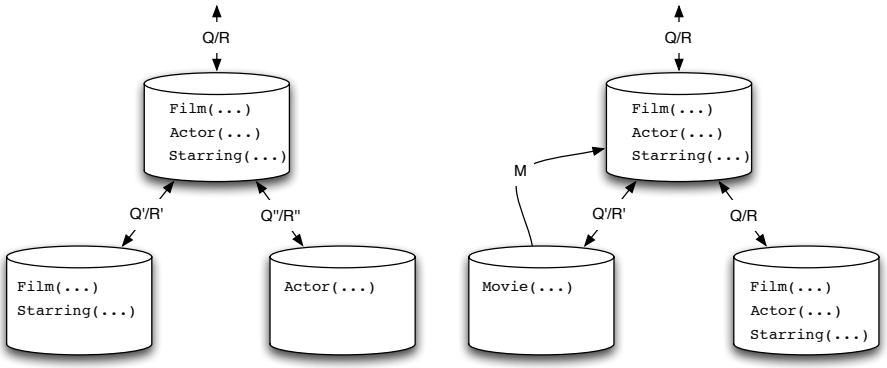
The powerful (relational) database systems developed in the 70s ensured data independence and interoperability of application programs. At the same time it was realized that more powerful data models were needed that reflect more of the semantics of these data and allow application-oriented and context-dependent design of databases. In the late 70s and early 80s 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 (Section 3.1). Then we introduce the central characteristics of object-oriented database models and systems (Section 3.2).

3.1 Distributed databases

In the late 70s, the research field of distributed databases[1] grew rapidly in importance. Early papers on distributed databases were Distributed INGRES [51] and POREL [41], 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 respectively 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) homogenous* and *(ii) heterogeneous* distributed databases. Figure 5 shows how our illustrative example can be deployed in a distributed setting.



(a) Homogeneous (b) Heterogeneous

Fig. 5. Homogeneous and Heterogeneous Distributed Databases Sample. In (a), we assume that studio B distributes the relations of its schema on 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.

In the first case, a top down design is realized by integrating external schemas into a single global schema. Guided by application-oriented meta data the de-

[1] See Ceri et al. [13] for an overview of distributed databases

sign 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 hindsight difficulties did arise 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 [34] and R* of IBM [28].

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 integrated 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 [37] uses Daplex, a logical data specification language, for modeling the various schemas. Heterogeneous SIRIUS-DELTA [38] 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 full functionality 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 became 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 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 heterogenous distributed database systems the need for *data consistency* and *object identity* became apparent. 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 multivalued 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 authors 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 (see Section 3.2).

3.2 Object-Oriented database Models and Systems

The main stream of databases — the relational model based systems, central or distributed — even when enhanced with Entity Relationship type semantic

descriptions did not show enough flexibility to support, for instance, the interoperation of heterogeneous systems or the extensibility for new appearing data types like semi-structured 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 80s, 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 OO 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. [19] and also in Bukhres et al. [11].

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 of the storage.

- *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*: objects of the same kind (in real world terms), that is, having the same data types, object attributes, behavior and relationships to other objects, will be collected in to 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 where 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 and today the OODBMS's can only be found in niche application fields. A simple example of and OODBM schema is given in Figure 6.

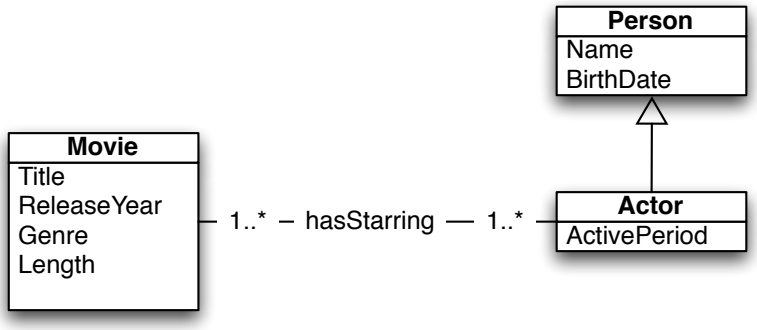


Fig. 6. Object-Oriented (UML) Model. The example shows the schema of studio B in an object-oriented representation using the UML notation. To illustrate the inheritance feature of OO model, we introduced a super-class `Person` that defines all the attributes that would describe persons (not only actors)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

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) [35] 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 *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 [44] and serves as formal basis for UML [45], 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 that allowed for the expression of many real world semantic properties and behaviors via concepts like meta classes, classes and inheritance, the expressive power needed in the opening-up world of multimedia and the World Wide Web. As a consequence the OODBMSs never became THE database concept envisioned in the late 80s and early 90s, 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 the semantics of data. With the rapidly increasing size of local and wide area computer networks, however, already established database-oriented interoperability mechanisms turned out to be insufficient due to the technical heterogeneities of the involved network nodes.

In the late 80s 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 (Section 4.1) plays a central role in this context. In parallel, research on Markup Languages (Section 4.2) evolved to a first agreed-upon standard (SGML), a derivative of which (XML) was later integrated with the Semistructured Data Model. Hypertext and Hypermedia research (Section 4.3) not only focused 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. Representative for these developments we discuss MPEG-7 (Section 4.4) and briefly outline other metadata interoperability approaches (Section 4.5).

4.1 The Semistructured Data Model

In all models available so far (Relational Model, ER Model, OO Model) data had a fixed schema describing the semantics of data. This leads to problems when data are exchange across systems, because the underlying databases usually do not share the same schema even if they maintain 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 [14] projects at Stanford University and was first described by Papakonstantinou et al. [47]. Unlike the other existing data models at that time, the semistructured model doesn't separate the schema from the data. It is *self-describing*, meaning that the data themselves carries their schema information. Data represented by the semi-structured model takes the form of a directed labelled 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. Different from 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 semi-structured data model provides the necessary flexibility for exchanging data across system boundaries. However, the price for this flexibility is

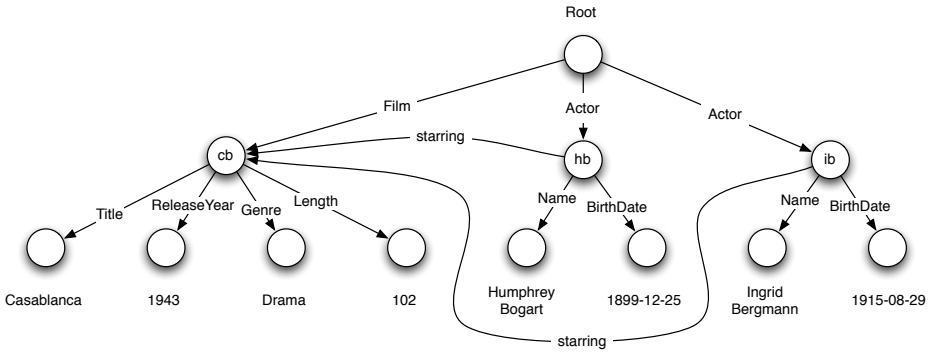


Fig. 7. Semistructured Model Example. A directed labelled graph represents the data of studio B. The graph is self-describing because the data also carry schema information.

the loss of efficiency in query processing. This is one of the reason why most of today's data are still represented in the very efficient relational model and the technologies based on the semi-structured 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 [58]. 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 markups, documents are simply files containing a sequence of characters. 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, the goal of markup languages is to add *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) invented by Mosher, Lorie, and Goldfarb (cf., [24]) was the first technical realization of a markup language. Scribe [48] 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) [32] 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 a grammar — the document type definition (DTD) — for defining the structure and allowed for tags in a document.

Prominent derivatives of the SGML are HTML, developed in 1991, and XML, 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. The extensibility and flexibility of HTML was one of the key factors for the success of the World Wide Web, with the result that until today HTML is the most widely used markup language.

While HTML mainly provides markup elements that define presentation semantics of document parts, XML [56] provides a simplified meta markup language for defining documents that contain data to be communicated between applications. Hence, the elements in XML documents indicate the semantics of contained data values. Since XML is backward-compatible to SGML, DTDs can be applied for imposing element definitions and document structures on XML documents. Nowadays, however, DTDs are superseded by 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.

```
<?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.

Soon it was realized that the freedom of the original HTML specification lead to interoperability problems among web browsers. 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 XHTML2[2] and HTML5[3] a competition on the

[2] http://www.w3.org/TR/xhtml2/

[3] http://www.whatwg.org/specs/web-apps/current-work/multipage/

next generation markup language started. At the time of this writing, HTML5 seems to be the winner because of its less strict, more evolutionary design approach.

4.3 Hypertext and Hypermedia

Inspired by Vannevar Bush's vision of Memex [12], Ted Nelson and Douglas Engelbart started their research on hypertext and hypermedia systems in the late 60s (cf., [40, 21]).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. Thus, hypertext was considered as a non-linear extension of traditional text organization. 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 [18].

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 hypertext nodes. But links can also be used to connect a comment or annotation to the text it writes about. 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. [54], 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 Section 4.1: the underlying models are directed labelled graphs.

For exchanging hypertext and hypermedia documents between applications, it soon became clear that a standardized exchange format is required in order to provide interoperability. *HyTime* is an example for such a standard (see Goldfarb [25]). Also the work in the *Dexter Group* focused on hypertext exchange formats and architectural models that should facilitate the exchange of hypertext [29, 26].

The *Synchronized Multimedia Integration Language (SMIL)* and *Scalable Vector Graphics (SVG)* specifications are based on hypertext and hypermedia research and find their application in the World Wide Web, 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) for addressing documents in the Web, HTML as a flexible markup language for creating hypertext documents, and HTTP as a protocol for the communication between clients and servers.

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 came up. 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 [39]. MPEG-7 should not be 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)*. MPEG-7 is still the most complete description standard for all kind of media (audio, image, video, graphics, etc.) and in that way it creates a common basis for describing different media types by a single standard.

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 [36].

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. Application domains range from home entertainment (e.g., personal multimedia collections) over cultural services (e.g., art galleries) to surveillance (e.g., traffic control). This wide application spectrum, however, resulted in an enormous complexity of that standard, which is considered as one of the reasons why the ambitious goals of MPEG-7 are yet unreached.

4.5 Other Metadata Interoperability Approaches

The 90s were characterized by 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 [20] to very domain-specific schemas such as ONIX [53], which provides standardization for the publishing industry.

Also the idea of 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 CRM[4] 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., Section 3.1) where a global schema defines the model primitives for querying the underlying databases. The difference is that now the global model interoperability

[4] `http://cidoc.ics.forth.gr/`

approaches are being applied in an open-world environment such as the Web. However, they inherit the problems distributed databases need to encounter 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 [31].

5 World Wide Web Semantics

In the late 90s the success of the World Wide Web became obvious. The combination of results from hypertext and markup-language research led to the specification of URI, HTML, and HTTP, which until today are the fundamental technologies the World Wide Web is built on (see Jacobs et al. [33]). The URI specification introduced a simple generic syntax for identifiers and unified previously existing identification mechanisms. HTML, as a simple to use markup language without formal schema binding, suddenly allowed also non-technical end users to easily create documents, which was one of the reasons for the rapid spreading of the Web. HTTP defines a simple protocol to access and manipulate resources and resource representations in a distributed environment. From a technical point of view, these technologies provided the necessary interoperability that allows Web users to access information objects via their Web browser.

The Semantic Web is an extension to 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 et al. [8] in an article published in the Scientific American in 2001. 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 lead to the specification of four major standards that technically enable this described vision: RDF/S, OWL, SKOS, and SPARQL.

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

[5] `http://www.w3.org/2001/sw/Activity`

Hence, all resources representing real-world objects in the Semantic Web 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 identified by a URI, the object can either be a resource or a literal node. A statement is represented as a *triple* and several statements form a *graph*. We will give an example of an RDF graph in Figure 9 in Section 5.2.

The *RDF Vocabulary Description Language RDF Schema (RDFS)* and the *Web Ontology Language (OWL)* are means to describe the vocabulary terms used in an RDF model. RDFS provides the basic constructs for describing classes and properties and allows to arrange them in 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.

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. 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 query language for formulating query patterns over RDF graphs. Additionally it defines a protocol for sending queries from clients to a SPARQL endpoint and for retrieving the retrieved results via the Web. Currently, the abstract protocol specification has bindings for HTTP and SOAP. The important distinction between SPARQL and other languages such as SQL is that it operates entirely through the Web: a client executes a query against a given endpoint (e.g., `http://dbpedia.org/sparql`) and retrieves the result set through common Web transport protocols.

A central believe in the early Semantic Web was that intelligent agents must 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., [27]). In its core, an ontology is similar to a database schema: a model defining the structure and semantics of data. Noy and Klein [43] 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.

This aspect is also reflected in OWL, the language for expressing ontologies. It is formally grounded in Description Logics and allows to perform useful reasoning tasks on Web documents.

Although intensive research was conducted in the Semantic Web domain for over ten years, the early vision of the Semantic Web is not yet implemented. Probably also because of the computational but also conceptual complexity introduced by the automated reasoning requirement.

5.2 Linked Data

In 2006 Tim Berners-Lee postulated 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 accentuated the data-centric aspects of the existing Semantic Web technologies and thereby demystified 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 shall 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.

The Linked Data idea rapidly raised interest in various communities. Shortly after the formation of the W3C Linking Open Data Community project[6], DBpedia [10] was launched as 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.

[6] `http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData`

[7] http://www.imdb.com/

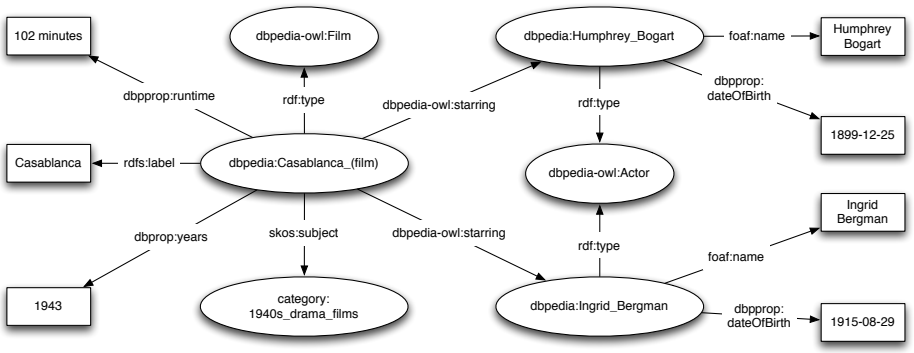


Fig. 9. Linked Data Example. The example how the data of studio B are exposed on the Web following the Linked Data guidelines. The prefixes expand as follows: `dbpedia` to `http://dbpedia.org/resource`, `dbpprop` to `http://dbpedia.org/property/`, `dbpedia-owl` to `http://dbpedia.org/ontology/`, and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

It provides a user-generated knowledge organization system comprising 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 [9]. 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 guidelines also resembles the notion of *dataspaces* [23] that was coined in the database community.

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.

In this article, 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 the 70s and ended with the currently ongoing activities in the Semantic Web 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 90s on, when research was driven by the evolving World Wide

[8] `http://www.w3.org/TR/swbp-vocab-pub/`

Web, the semi-structured 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 was 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 for that; it provides a uniform way for 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.

Similarly the currently ongoing Semantic Web / Linked Data activities do not solve the complete stack of interoperability problems. The proposed technologies (RDF/S, OWL, etc.) provide the necessary technical and structural interoperability but they do not solve the semantic interoperability problem. Different people still use different vocabularies to describe semantically related real-world concepts and even if one and the same vocabulary is used for a specific concepts, the interpretation of the terms still vary, which in turn leads to data heterogeneities.

In the foreseeable future, it seems that standardization and global model attempts can hardly solve the semantic interoperability problem. As long as people are the designers of models there will always exist different conceptions and interpretations, even for superficially homogenous domains and application contexts. We therefore believe that computer science research should take this situation into account and find solutions that deal with a multitude of models and allow for their reconciliation. The establishment of mappings between existing models is such an approach. Rather than imposing a single "agreed-upon" mapping mechanisms we should accept the variety in existing models and establish semantic relationships between the components of these models. Since at the end it is important to implement interoperability on a technical level such semantic relationships must be tightly bound to technical mapping specifications defining the necessary transformations of data representations.

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 will is a logical consequence. The Linked Data movement is definitely an important starting point in this direction. However, it will require further research on performance

and scalability of the underlying (graph-based) data stores. Additionally, since data exposed on the Web, should at the end also be consumable my 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 call for a review and possible also their adaption to a novel setting.

Also the evolution of schemas and ontologies in decentralized semantic structures such as the World Wide Web calls for further research. Aberer et al. [1] coined the term *Emergent Semantics*, which denotes a research field focusing on the understanding of semantics by investigating the relationships between syntactic structures using social networking concepts for the necessary human interpretations.

References

1. Karl Aberer, Tiziana Catarci, Philippe Cudré-Mauroux, Tharam Dillon, Stephan Grimm, Mohand-Said Hacid, Arantza Illarramendi, Mustafa Jarrar, Vipul Kashyap, Massimo Mecella, Eduardo Mena, Monica Scannapieco, Fèlix Saltor, Luca De Santis, Stefano Spaccapietra, Steffen Staab, Rudi Studer, and Olga De Troyer. Emergent semantics systems. In *In International Conference on Semantics of a Networked World (ICSNW*, pages 14–43, 2004.
2. Serge Abiteboul, Peter Buneman, and Dan Suciu. *Data on the Web: From Relations to Semistructured Data and XML*. Morgan Kaufmann, 1999.
3. Serge Abiteboul, Dallan Quass, Jason McHugh, Jennifer Widom, and Janet L. Wiener. The lorel query language for semistructured data. *Int. J. on Digital Libraries*, 1(1):68–88, 1997.
4. Jean-Raymond Abrial. Data semantics. pages 1–60. North-Holland, 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. Tim Berners-Lee. *Linked Data*. World Wide Web Consortium, 2006. Available at `http://www.w3.org/DesignIssues/LinkedData.html`.
7. Tim Berners-Lee and Dan Conolly. *RFC 1866 — Hypertext Markup Language - 2.0*. Network Working Group, 1995.
8. Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientifc American*, May 2001.
9. Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data - the story so far. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 5(3), 2009.
10. Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. DBpedia - A Crystallization Point for the Web of Data. *J. Web Sem.*, 7(3):154–165, 2009.
11. Omran A. Bukhres and Ahmed K. Elmagarmid, editors. *Object-Oriented Multidatabase Systems: A Solution for Advanced Applications*. Prentice-Hall, 1996.
12. Vannevar Bush. As we may think. *The Atlantic Monthly*, 176(1):101–108, 1945.
13. Stefano Ceri and Giuseppe Pelagatti. *Distributed Databases: Principles and Systems*. McGraw-Hill Book Company, 1984.

14. Sudarshan Chawathe, Hector Garcia-Molina, Joachim Hammer, Kelly Ireland, Yannis Papakonstantinou, Jeffrey D. Ullman, and Jennifer Widom. The TSIMMIS project: Integration of heterogeneous information sources. In *16th Meeting of the Information Processing Society of Japan*, pages 7–18, Tokyo, Japan, 1994.
15. Peter P. Chen. The entity-relationship model: Toward a unified view of data. In Douglas S. Kerr, editor, *VLDB*, page 173. ACM, 1975.
16. Peter P. Chen. The entity-relationship model - toward a unified view of data. *ACM Trans. Database Syst.*, 1(1):9–36, 1976.
17. E. F. Codd. A relational model of data for large shared data banks. *Commun. ACM*, 13(6):377–387, 1970.
18. J. Conklin. Hypertext: An introduction and survey. *Computer*, 20(9):17 –41, Sept. 1987.
19. Asuman Dogac, M. Tamer Özsu, Alexandros Biliris, and Timos K. Sellis, editors. *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*, volume 130 of *NATO ASI Series F: Computing and Systems Sciences*, 1994.
20. Dublin Core Metadata Initiative. *Dublin Core Metadata Element Set, Version 1.1*, December 2006. Available at: `http://dublincore.org/documents/dces/`.
21. D. C. Engelbart. *Augmenting Human Intellect: A Conceptual Framework*. Stanford Research Institute, Menlo Park, CA, 1962.
22. Eckhard D. Falkenberg. Concepts for modelling information. pages 95–109, Freudenstadt, Germany, 1976. North-Holland.
23. Michael Franklin, Alon Halevy, and David Maier. From databases to dataspaces: a new abstraction for information management. *SIGMOD Rec.*, 34(4):27–33, 2005.
24. C. F. Goldfarb. A generalized approach to document markup. In *Proceedings of the ACM SIGPLAN SIGOA symposium on Text manipulation*, pages 68–73, New York, NY, USA, 1981. ACM.
25. C.F. Goldfarb. Standards-hytime: a standard for structured hypermedia interchange. *Computer*, 24(8):81 –84, Aug 1991.
26. Kaj Grønbæk and Randall H. Trigg. Hypermedia system design applying the dexter model. *Commun. ACM*, 37(2):26–29, 1994.
27. Tom Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisitions*, 5:199–220, 1993.
28. Laura M. Haas, Patricia G. Selinger, Elisa Bertino, Dean Daniels, Bruce G. Lindsay, Guy M. Lohman, Yoshifumi Masunaga, C. Mohan, Pui Ng, Paul F. Wilms, and Robert A. Yost. R*: A research project on distributed relational dbms. *IEEE Database Eng. Bull.*, 5(4):28–32, 1982.
29. Frank Halasz and Mayer Schwartz. The dexter hypertext reference model. *Commun. ACM*, 37(2):30–39, 1994.
30. Terry Halpin. Object-role modeling (orm/niam). In *Handbook on Architectures of Information Systems*, pages 81–102. Springer-Verlag, 1998.
31. Bernhard Haslhofer and Wolfgang Klas. A survey of techniques for achieving metadata interoperability. *ACM Comput. Surv.*, 42(2), 2010.
32. ISO JTC1 SC34. *ISO 8879:1986 Information processing – Text and office systems – Standard Generalized Markup Language (SGML)*, 196.
33. Ian Jacobs and Norman Walsh. Architecture of the world wide web, volume one, December 2004. Available at: `http://www.w3.org/TR/webarch/`.
34. James B. Rothnie Jr., Philip A. Bernstein, Stephen Fox, Nathan Goodman, Michael Hammer, Terry A. Landers, Christopher L. Reeve, David W. Shipman, and Eugene

Wong. Introduction to a system for distributed databases (sdd-1). *ACM Trans. Database Syst.*, 5(1):1–17, 1980.
35. Wolfgang Klas, Karl Aberer, and Erich J. Neuhold. Object-oriented modeling for hypermedia systems using the vodak model language. In *NATO ASI OODBS*, pages 389–433, 1993.
36. Harald Kosch. *Distributed Multimedia Database Technologies Supported MPEG-7 and by MPEG-21*. CRC Press LLC, Boca Raton, Florida, 2003.
37. Terry A. Landers and Ronni Rosenberg. An overview of multibase. In *DDB*, pages 153–184, 1982.
38. Witold Litwin, J. Boudenant, Christian Esculier, Arlette Ferrier, A. M. Glorieux, J. La Chimia, K. Kabbaj, Catherine Moulinoux, P. Rolin, and Christine Stangret. Sirius system for distributed data management. In *DDB*, pages 311–366, 1982.
39. Frank Nack and Adam T. Lindsay. Everything you wanted to know about MPEG-7: Part 1. *IEEE MultiMedia*, 6(3):65–77, July–September 1999.
40. T. H. Nelson. Complex information processing: a file structure for the complex, the changing and the indeterminate. In *Proceedings of the 1965 20th national conference*, pages 84–100, New York, NY, USA, 1965. ACM.
41. Erich J. Neuhold and Horst Biller. Porel: A distributed data base on an inhomogeneous computer network. In *VLDB*, pages 380–395. IEEE Computer Society, 1977.
42. G.M. Nijssen. Current issues in conceptual schema concepts. Nice, France, 1977. North-Holland.
43. Natalya F. Noy and Michel Klein. Ontology evolution: Not the same as schema evolution. *Knowl. Inf. Syst.*, 6(4):428–440, 2004.
44. Object Management Group (OMG). *Meta Object Facility (MOF) Core Specification - Version 2.0*, January 2006. Available at: `http://www.omg.org/cgi-bin/apps/doc?formal/06-01-01.pdf`.
45. Object Management Group (OMG). *Unified Modelling Language (UML)*, 2007. Available at: `http://www.uml.org/`.
46. A. M. Ouksel and A. Sheth. Semantic interoperability in global information systems. *SIGMOD Rec.*, 28(1):5–12, 1999.
47. Yannis Papakonstantinou, Hector Garcia-Molina, and Jennifer Widom. Object exchange across heterogeneous information sources. pages 251–260, 1995.
48. Brian K. Reid. A high-level approach to computer document formatting. In *POPL '80: Proceedings of the 7th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 24–31, New York, NY, USA, 1980. ACM.
49. Jennifer Rowley. The wisdom hierarchy: representations of the DIKW hierarchy. *Journal of Information Science*, 33(2):163–180, 2007.
50. Amit P. Sheth and James A. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Comput. Surv.*, 22(3):183–236, 1990.
51. Michael Stonebraker and Erich J. Neuhold. A distributed database version of ingres. In *Berkeley Workshop*, pages 19–36, 1977.
52. Bo Sundgren. *An Infological Approach to Data Bases*. PhD thesis, University of Stockholm, 1973.
53. The EDItEUR Group. *Online Information Exchange (ONIX)*, 2007. Available at: `http://www.editeur.org/onix.html`.
54. Randall H. Trigg and Mark Weiser. Textnet: a network-based approach to text handling. *ACM Trans. Inf. Syst.*, 4(1):1–23, 1986.

55. Pepjijn R. S. Visser, Dean M. Jones, T. J. M. Bench-Capon, and M. J. R. Shave. An analysis of ontological mismatches: Heterogeneity versus interoperability. In *AAAI 1997 Spring Symposium on Ontological Engineering*, Stanford, USA, 1997. Stanford University.
56. W3C XML Activity. *Extensible Markup Language (XML) 1.0*. W3C, 1998. Available at: `http://www.w3.org/TR/1998/REC-xml-19980210`.
57. Holger Wache. *Semantische Mediation für heterogene Informationsquellen*. PhD thesis, University of Bremen, 2003.
58. Gio Wiederhold. Mediators in the architecture of future information systems. *Computer*, 25(3):38–49, 1992.