On a Conceptual Data Modeling Tool via Data Abstraction Using the Entity Relationship Model

Japheth R. B.
Department of Mathematics/Computer Science
Faculty of Science
Niger Delta University
Wilberforce Island, Nigeria.
jbunakiye@gmail.com

Ogheneovo E. E.
Department of Computer Science
Faculty of Science
University of Port Harcourt
Port Harcourt, Nigeria.
edward_ogheneovo@yahoo.com

ABSTRACT

Data modeling is an important part of database design. The data model focuses on what data should be stored in the database, exploring the relation between data entities. The goal of the data model is to ensure that all data object required by the database are accurately represented. This paper presents data abstraction technique to ensure data is modeled and represented accurately using the entity relationship model. The ER modeling method provides a wide variety of possibilities in a database conceptually. There has to be a balance between possibilities to bring to bear any preferred representation, and a very good way to achieve this fit in database management is through data abstraction where all the levels of abstraction are applied for generalization, aggregation and also describes a subset of the data in the database for a particular set of users.

Key Words: Data Concurrency, Data Integrity, Entity Sets, Abstraction Levels

CISDI Journal Reference Format
Japhet, R.B & Ogheneovo, E.E. (2017): On a Conceptual Data Modeling Tool via Data Abstraction Using the Entity Relationship Model Computing, Information Systems, Development Informatics & Allied Research Journal. Vol 8 No 1. Pp 61-68
Available online at www.cisdijournal.net

1. INTRODUCTION

Databases are now ubiquitous in any areas where information has to be stored and processed. Their importance comes from the value of information they store [1]. The most important criteria for them can be described by two keywords reliability (covering many subareas from robustness to issues related to concurrency and security) and efficiency (covering not only data manipulation speed but also its flexibility towards new requirements or the degree to which database supports application software development). A Database management system (DBMS) consists of a collection of structured, interrelated and persistent data, and a set of application programs to access, update and manage the data. Thinking a bit about the way data is managed in a conventional DBMS, there are some identified problems that usually occur when trying to solve the problem without planning [12]. Some of the problems commonly known to be coped with in a DBMS include data redundancy and inconsistency, data integrity, data access, data isolation, concurrency, and security issues. Data redundancy and inconsistency means the same information should not appear at several places of the system, and also related data should be updated consistently. Data integrity issues bothers on making sure stored data fulfils certain prescribed constraints, adapt to the change of the constraints, and should be able to recover from crashes [3].

Data access concerns with how the database system can generate answers to a large class of queries, it also means the system should support efficient data retrieval (e.g. by indexing, hashing). Data isolation issues are related to the ability to handle different types and/or magnitudes of data elements like text documents, numerical data, photos, etc. Concurrency involves the capacity of the system to work with multiple users at the same time, the concurrent modifications should not disturb the consistency of the data, and as well, avoid deadlocks [19]. Security in a DBMS handle access rights for users by providing secure remote access to the database. This paper is looking at how the mentioned problems can be addressed using the data abstraction data base management technology. The technicalities involved are presented using the entity relationship model to model data conceptually so that the desired manipulations in a DBMS can be achieved [7]. Data abstraction is the most important tool to treat the above problems conceptually.

There are three levels of abstraction (sometimes also referred to as database schemes): physical level, conceptual level, and view level [5]. Physical level deals with the storage and retrieval of the data on the disk (data and index file structures). This level is usually hidden from the users by the DBMS. Conceptual level handles the questions of what data are stored and what are the relationships between these data are decided. This is the level of the database administrator [2]. View level describes a subset of the data in the database for a particular set of users. The same data can be viewed in many different ways for instance the lecturer wants list of the students subscribed for his course while the dean of studies is interested in the number of people attending the course.

2. BACKGROUND AND RELATED WORK

Data models are collections of conceptual tools to describe data, data relations, and data constraints and data semantics [18]. The entity-relationship (E-R) model, which is an example of a data model describe data on the conceptual and view levels with the provision of flexible structuring capabilities. An important E-R concept is the one of data independence, which expresses that change in the database scheme at a given level of abstraction, should not affect the schemes at higher levels [22]. Thus it is adopted as the data model on which data abstraction mechanisms are implemented. In the data series report focusing on Conceptual Model to DBMS, Sparx Systems [20] provided a visual modeling platform that supports comprehensive functionality for modeling database structures.

The presentation covers the core features for data modeling over the full lifecycle of an application, initially, the basic modeling process were discussed outlining the conceptual model and then working through the steps to form a concrete database schema. Ours focused on applying the data abstraction modalities on the entity relationship model. Avi, S. et al. [23] sees the data model as the underlying structure of a database, they posited that a data model is a collection of conceptual tools for describing the real world entities to be modeled in the database and the relationships among these entities.

Thus their focus was on the object based and record based logical models that can be logically organized and merged with object oriented data model to a new relational model for development. Shashi, S et al. [21] centered their discussion on the successful development of any geographic information system via conceptual and logical data-modeling. Their reasoning was based on the fact that conventional entity-relationship diagrams have limitations for conceptual data-modeling, since they get cluttered with numerous relationships. So they presented an extension to ER diagrams using pictograms for entities and as well as relationships. This approach complements ours in the sense that a grammar sequence was introduced to abstract the data and modify it to the logical models.

3. THE ENTITY-RELATIONSHIP MODEL

The ER model views the real world as the set of entities with attributes and relationships between them. An entity is an object with identity, i.e. an object which is distinguishable from other objects [15]. Entities are concrete or abstract objects, persons, things, documents, events, programs, etc., about which we intend to collect information. Entities are described by attributes which are, from the formal point of view, functions assigning to an entity an element from a domain of permitted values [4]. For example let the entity set be the students of the university; let us describe a student for the sake of simplicity only with the attributes: first-name, last-name, student-ID, and birth-date. The first-name and last-name attributes can take values from the nonempty strings of characters which will be the domain of these attributes.

The student-ID can be a seven-digit positive numbers and the birth-date can be a string of the format dd-mm-yyyy with the usual semantics. An entity (a particular student of the university) can be described then as first-name, John), (last-name, Joseph), (student-ID, 0145897), (birth-date, 21-09-1987). Of course, in any kind of information processing the entity appears only via its description by the set of attributes which are relevant from the point of view of the task to be carried out.

A relationship represents association between two or more entities. A relationship set is the set of relationships of the same type, for example let another entity set be the set of courses offered in the current semester at the university. Let the attributes describing a course be the course-ID, lecturer-ID, course-title. Let a course be described by (course-ID, 544367), (lecturer-ID, 265478), (course-title, Compiler Design). If the student of the other exercise takes this course, there is a relationship between the two entities (the student and the course). If we take the entity sets of all students and all the courses offered, the arising relationship set will describe which student takes what courses this semester. It can be easily recognized that a relationship can also be taken as an abstract entity, and hence a relationship set can be described just as an entity set [6]. The relationship set that associates students to the courses they take in a semester can be described via the entity set whose elements are described by the attributes student-ID and course-ID. The relationship between John Joseph and the Compiler Design course would then be described by {(student-ID, 0145897), (course-ID, 544367)}.

4. GRAPHICAL EXPRESSION OF DATABASE LOGICAL STRUCTURE

Entity Relationship diagrams are a graphical way to express the logical structure of a database. In the light of this paper, as shown in figure 1; some notations are used to describe the entities, attributes and the relationships in an E-R model showing the relationship between student members registered in the university library and the relevant book(s) borrowed. Entity sets are represented by labelled rectangles, for weak entity sets the rectangles have double border [8]. Relationship sets are represented by a solid line connecting two entity sets with the name of the relationship written beside the line. Attributes, when included, are listed inside the entity set rectangle. Attribute names should be singular nouns. Members of the primary key can be denoted by underlining. If the line ends in an arrow, the relationship cardinality is Many for the entity set at that end. Otherwise the cardinality is one. Existence dependency is denoted by a bar crossing the line of the relationship set at the dominant entity set.

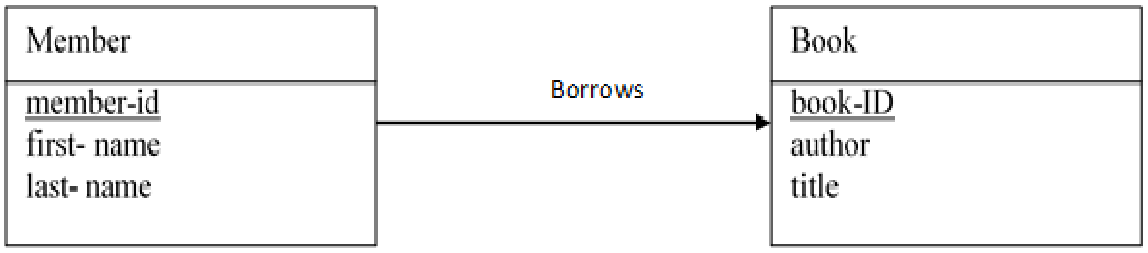


Figure 1. An E-R Diagram

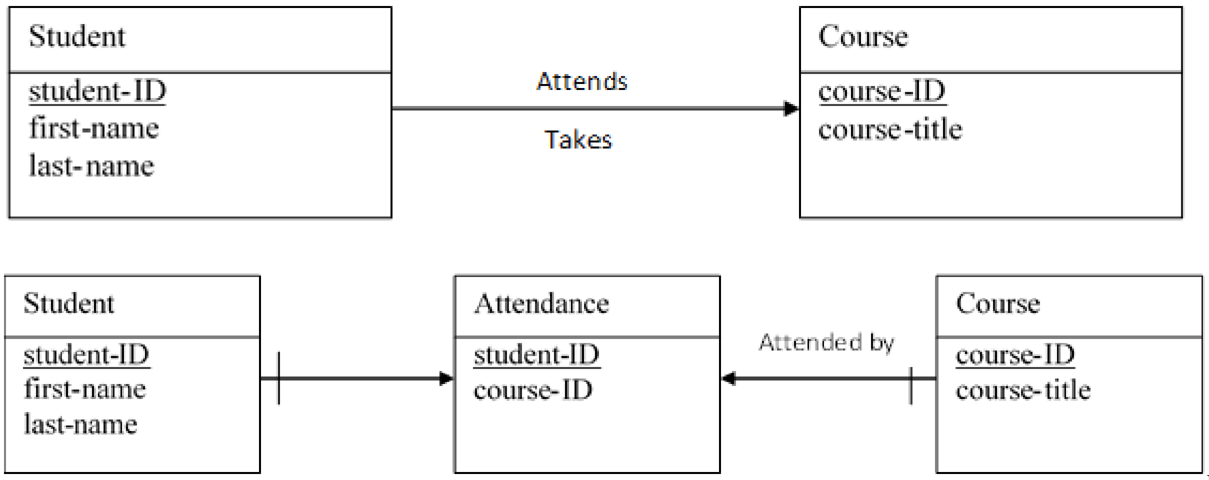
A database conforming to an E-R diagram as shown in table 1 can be represented by a collection of tables in tandem with relational databases. For each entity set corresponds a table with as many columns as many attributes the entity set is described with [11]. A row of the table will represent an entity, containing the corresponding attribute values. For weak entity sets we attach to the columns of the primary key of the strong entity set on which it depends. Relationship sets can also be described this way. Tables will contain only a subset of all possible rows: the ones which represent entities in the system we describe. As the system changes in time, we may add, delete or modify rows [17].

Table 1 Database Conforming to E-R Model

Area-code	Area-name	Phone-number	Owner-first-name	Owner-last-name	Area-code
0463	Lagos	23469188635	Marian	Benson	0123
0335	Kaduna	23418008234	Thompson	Millionaire	0732
0783	Yenagoa	23465908432	France	Stephen	0662
0442	Bonny	23486382377	Amaka	Okorie	0732
02993	Warri	23469005235	Perekimi	Ezekiel	0557
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.

5. RESOLVING RELATIONSHIPS

Data can be abstracted in such a way that many-to-many relationships cannot be taken over directly to the relational model, such relationships should be resolved to One-to-Many relationships as early in the design phase as possible. The method is to introduce a new entity type for the relation itself and associate the original entity types to this new one via One-to-Many relationships [16]. For example take the entity sets of students and courses and the Many-to-Many relationship set between them that associate students and the courses they attend. We introduce then a new entity set with name Attendance with two attributes: student-ID, course-ID, whose entities express the fact that a certain student attends a given course [13]. Then establish a relationship between a student and an entity in Attendance if the student-ID is the same, and we act analogously for courses and entities of Attendance. The E-R diagram to this abstraction is shown in figure 2. It is important to note; this abstraction requires that for each Attendance entity a student and a course entity exists.



Fig

Figure 2 Resolving Many-to-Many relationships

5.1 Complex Relationships of Degree Greater Than Two

Complex relationships of degree greater than two can be resolved to binary One to-Many relationships using the same technique for resolving Many-to-Many relationships. Also for a conceptually clearer presentation, this technique is used to describe relationships between relationships (see fig 3).

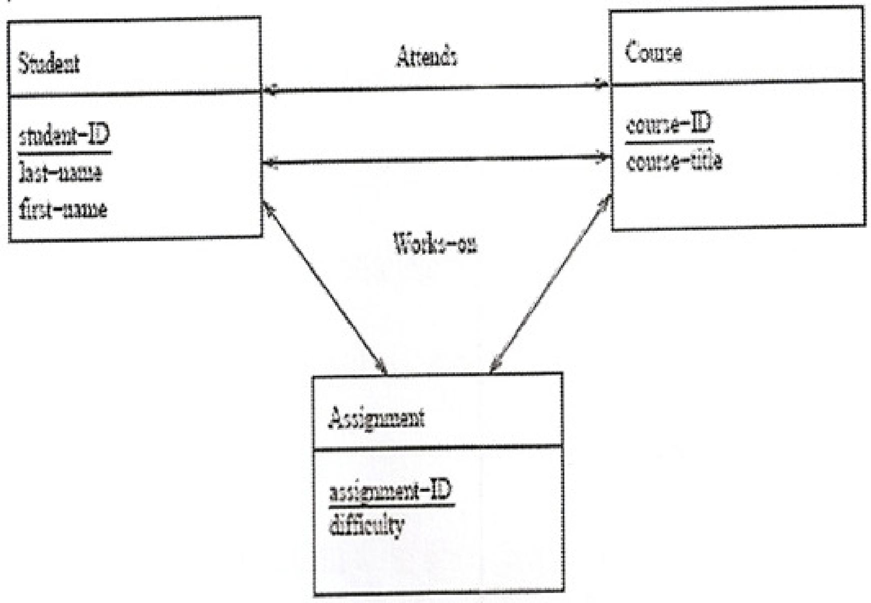


Figure 3 Relationships Resolved to binary One to-Many

The process of resolving relationships involves creating new entities for relationship sets, considering them on a higher abstraction level, and establish the relationships between them. As shown in figure 4 the entity sets are aggregated because they share common attributes; their generalization can be formed by taking only the common attributes, which can be considered as creating an abstraction type (entity set) that describes similarities of the original entity sets.

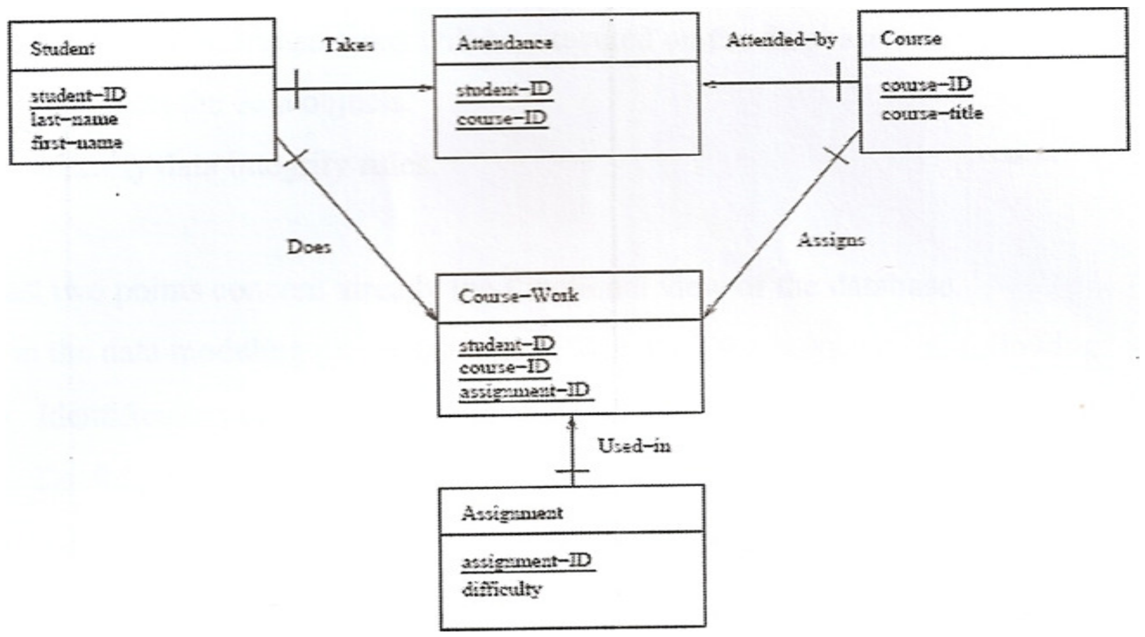


Figure 4 Common Attributes Aggregation

The generalized entity set, which is the super type original entity set is at a higher level of abstraction, and at the lower abstraction level, are the subtypes. Viewing the construction from top-down, the subtypes inherit the attributes of the super type [14]. Subtypes can be mutually exclusive (an entity can be a member only on one of the subtypes) or overlapping (otherwise). This process can be applied repeatedly in both 'directions, resulting in a generalization hierarchy. Generalization can be expressed by an element of the E-R diagram via which the super type is connected to the subtypes.

5.2 Classification of Relationships

The degree of a relationship is the number of entity sets it associates to each other. Higher degree relationships can be decomposed into lower ones, thus these are the most important occurrences (see fig 5). The (mapping) cardinality of a relationship expresses the number of entities that can be associated via it. For example, a typical One-to-One relationship would be the one between the students and the ID cards; every student can have at most one ID card and every ID card can belong to at most one student, which means if the student lost the card, it should be immediately invalidated but it might be produced at a later date. A typical One-to-Many relationship would be the one between members and books of a library; a member can borrow many books but a book, which means that an instance can be borrowed, by only one person at a time [9].

A typical Many-to-Many relationship would be the one between students and courses; a student can take many courses and a course can be attended by many students. The existence dependency of a binary relationship expresses that the existence of an entity is dependent on another, related entity. If in a relationship between entities p and q, the existence of p depends on the existence of q, we call q the dominant and p the subordinate entity. Taking the relationship between students and courses, for example, the first one has existence dependency with lecturers being the dominant entities and courses the subordinate ones. That is, for each course there must be someone to lecture it. There is no existence dependency between students and courses, which mean that certain courses are not obligatory for any students, thus there might be nobody taking them in a given semester.

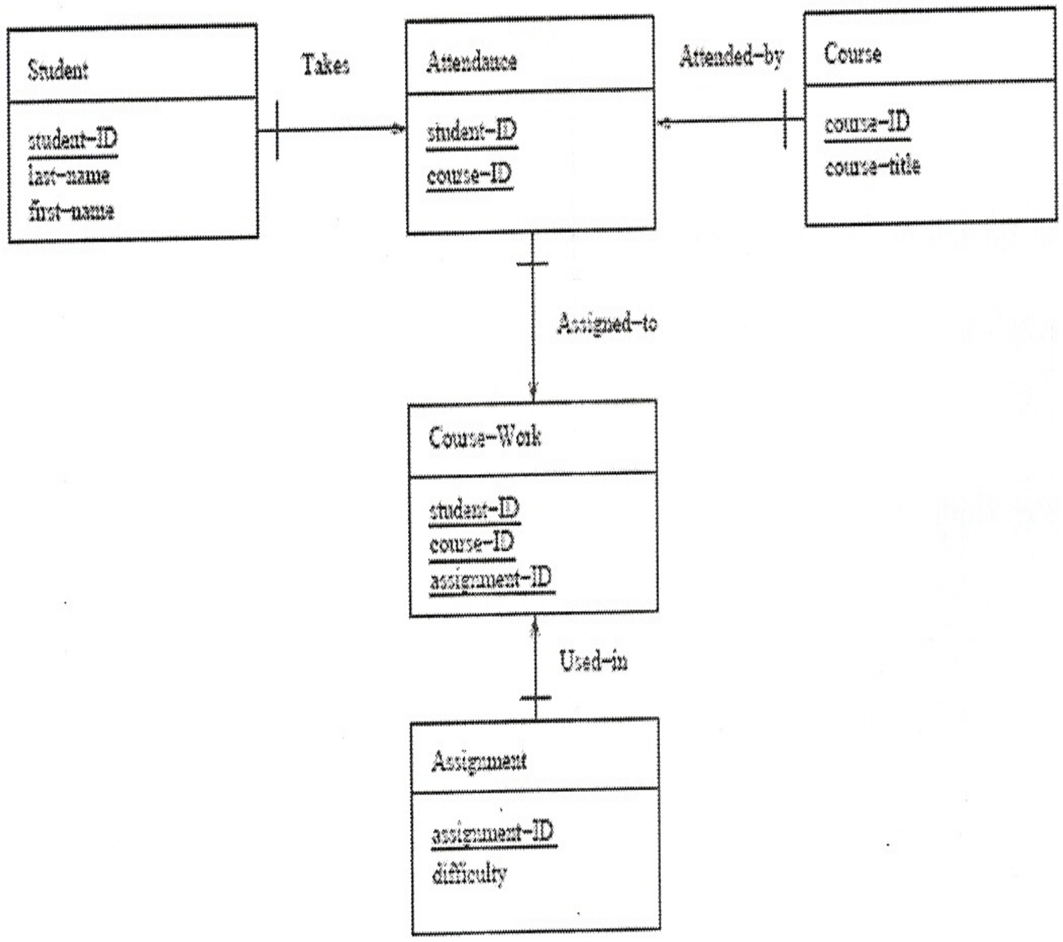


Figure 5: Decomposing Higher Degree Relationships

6. DISTINGUISHABLE ENTITIES

Since entities must be distinguishable, we must be able to choose a set of attributes for each entity set, so that there is not two entities in the set described by the same set of attribute values. Usually the set of attributes is more than enough to distinguish entities, and what we try to do is to find small subsets of attributes that still have this property [10]. A super key is a nonempty set of attributes which allows us to uniquely identify an entity of the entity set. A super key which does not have a proper subset which is also a super key is called a candidate key. A primary key is a candidate key which is chosen by the database administrator to identify entities in an entity set. There might be more than one candidate keys since set inclusion is only a partial order. Which candidate key becomes a primary key is apparently a matter of choice. An entity set that does not possess sufficient attributes to form a primary key is called a weak entity set. One that does have a primary key is called a strong entity set. A weak entity set must be part of a One-to-Many relationship in a subordinate role while the entity set with the dominant role must be a strong one so that it allows distinction between the entities of the weak set via the relation.

A discriminator is a set of attributes in the weak entity set that allows the distinction of the entities via a given relation with a strong entity set on which the weak one existence depends. A primary key for a weak entity set is the primary key of the strong entity set with which it is related together with a discriminator with respect to this relation [12]. For example take the entity set of all the telephones in a country, where an entity is described with the attribute phone-number (without any country or area codes). This is a weak entity set because many phones in different areas can have the same number (although the entities, that are the phones, are different). To make the system work we have to introduce the entity set of phone areas, where an entity is described by the attribute area-code.

The One-to-Many relation associates to an area each phone in it. Since the areas cover the country completely, no phones exist without an area, thus there is an existence dependency in which the areas are the dominant and the phones are the subordinate entities [5]. The area-code is a super key, a candidate key, and (after we declare it so) a primary key of the entity set of phone areas. The phone-number is a discriminator of the entity set of telephones with respect to the primary key we have chosen for the phone areas, via the relation we established between areas and phones. And the area-code with the phone number forms a primary key for the entity set of telephones in the country.

7. CONCLUSION

The ER modeling method provides a wide variety of possibilities and there is no standardize way to achieve optimal designs. Using the E-R model; data abstraction is one important tool to providing a wide variety of possibilities in a database conceptually. One has to balance between possibilities like to describe a class of objects by an entity set or via a relationship set, whether to allow weak entity sets or prefer strong ones, whether to apply generalization or aggregation or omit them to keep the ER model simpler, and so on. The levels of abstraction describe storage and retrieval of the data, handles the questions of what data are stored and what are the relationships, and also describes a subset of the data in the database for a particular set of users.

REFERENCES

[1] R. Angles and C. Gutierrez Levene, M. and Poulovassilis, A. 1990. The Hypernode model and its associated query language. In Proceedings of the 5th Jerusalem Conference on Information technology. IEEE Computer Society Press, 520–530.
[2] Levene, M. and Poulovassilis, A. 1991. An object-oriented data model formalized through hypergraphs. Data Knowl. Eng. 6, 3, 205–224.
[3] Mainguenaud, M. 1995. Modelling the network component of geographical information systems. Int. J. Geog. Inform. Syst. 9, 6, 575–593.
[4] Mannino, M.V. and Shapiro, L. D. 1990. Extensions to query languages for graph traversal problems. IEEE Trans. Knowl. Data Eng. 2, 3, 353–363.
[5] Mcguinness, D. L. and Van Harmelen, F. 2004. OWL Web ontology language overview, W3C recommendation 10 (February). http://www.w3.org/TR/2004/REC-owl-features-20040210/.
[6] Navat H E, S. B. 1992. Evolution of data modeling for databases. Communications of the ACM 35, 9, 112–123.
[7] Nejdl, W., Siberski, W., and Sintek, M. 2003. Design issues and challenges for RDF- and schema-based peer-to-peer systems. SIGMOD Record 32, 3, 41–46.
[8] Newman, M. E. J. 2003. The structure and function of complex networks. SIAM Rev. 45, 2, 167–256.
[9] Olken, F. 2003. Tutorial on graph data management for biology.IEEE Computer Society Bioinformatics Conference (CSB).
[10] Papakonstantinou, Y., Garcia-Molina, H., and Widom, J. 1995. Object exchange across heterogeneous information sources. In Proceedings of the 11th International Conference on Data Engineering (ICDE). IEEE Computer Society, 251–260.
[11] Pepper, s. and Moore, G. 2001. XML topic maps (XTM) 1.0—TopicMaps.Org Specification. http://www.topicmaps.org/xtm/1.0/xtm1-20010806.html.
[12] Poulovassilis, A. and Hild, S. G. 2001. Hyperlog: A graph-based system for database browsing, querying, and update. IEEE Trans. Knowl. Data Eng. 13 , 2, 316–333.
[13] Hommeaux, E. AND S Eaborne, A. 2005. SPARQL Query Language for RDF, W3C Working Draft 21 July. http://www.w3.org/TR/2005/WD-rdf-sparql-query-20050721/.
[14] Shasha , D., Wang ,J.T.L., and Giugno , R. 2002. Algorithmics and applications of tree and graph searching.In Proceedings of the 21th Symposium on Principles of Database Systems (PODS) . ACM Press, 39–52.
[15] Sheth, A., Leman, M., Eza , B., Anyanwu , K., and Kochut, K. 2005. Semantic association identification and knowledge discovery for national security applications. J. Database Manag. 16 , 1 (Jan-March), 33–53.
[16] Tsvetovat, M., Diesner, J., and Carley, K. 2004. NetIntel: A database for manipulation of rich social network data. Tech. Rep. CMU-ISRI-04-135, Carnegie Mellon University, School of Computer Science, Institute for Software Research International.
[17] Vianu, V. 2003. A Web odyssey: From Codd to XML. SIGMOD Record 32 , 2, 68–77.
[18] Yannakakis, M. 1990. Graph-theoretic methods in database theory. In Proceedings of the 9th Symposium on Principles of Database Systems (PODS). ACM Press, 230–242.

[19] Zicari, R. 1991. A framework for schema updates in an object-oriented database system. In Proceedings of the 7th International Conference on Data Engineering (ICDE). IEEE Computer Society, 2–13. Received November 2005; revised October 2006, April 2007; accepted May 2007ACM Computing Surveys, Vol. 40, No. 1, Article 1, Publication date: February 2008.

[20] Sparx Systems Series: Data Modeling From Conceptual Model to DBMS Enterprise Architect Visual Modeling Platform http://www.sparxsystems.com All material (c) Sparx Systemshttp://www.sparxsystems.com ©Sparx Systems 2011 Page:1

[21] Shashi Shekhar1, Ranga Raju Vatsavai1Spatial Pictogram Enhanced Conceptual Data Models and Their Translation to Logical Data Models shekharjvatsavaijchawla]@cs.umn.edu http://www.cs.umn.edu/research/shashi-group/

[22] Hugh W. Calkins Entity-Relationship Modeling Of Spatial Data For Geographic Information SystemsDepartment of Geography andNational Center for Geographic Information and Analysis, State University of New York at Buffalo Amherst, NY 14261-0023

[23] Avi Silberschatz Henry F_ Korth S_ Sudarshan Sikha Bagui: “Achievements and Weaknesses of Object-Oriented Databases”, in Journal of Object Technology, vol. 2, no. 4, July-August 2003, pp. 29-41. http://www.jot.fm/issues/issue_2003_07/column2