# A METHODOLOGY FOR THE DESIGN OF SPECIFIC-PURPOSE DATA MODELS APPLIED TO MULTIMEDIA AND GEOGRAPHICAL INFORMATION SYSTEMS*

Jonas A. Montilva C.

University of Los Andes
Faculty of Engineering
Computer Science Department
Data & Knowledge Engineering Research Group
Merida, Venezuela
e-mail: jonas@ing.ula.ve

## ABSTRACT

The integration of spatial and multimedia databases is an essential requirement imposed by the new generation of geographical information systems. Integrating spatial and multimedia databases is a complex process that takes place at three different levels: conceptual, language and system levels. This paper is concerned with the integration of data models at the conceptual level. It describes a methodology for the design of specific-purpose data models. It illustrates the application of the methodology by presenting some aspects of the design of a *hypermap data model*, an object-oriented model that enhances the ability of a geographical information system to manipulate maps with multimedia capabilities. The methodology has three key benefits: (1) it organizes the effort required to design a specific-purpose data model; (2) it reduces the complexity of the conceptual integration process, when many models are merged to create a new one; and (3) it serves as a valuable research tool for investigating the software integration process at a conceptual level.

## KEYWORDS

Software integration; data model design; multimedia and spatial data models; geographical information systems.

## INTRODUCTION

The demand for new data models tailored to specific types of information systems is becoming more apparent. In non-traditional database application domains, such as spatial or geographical information systems (GIS), multimedia information systems, office information systems, and intelligent information systems, the need for specialized models that satisfy particular requirements of their applications has been recognized as an important research direction. These domains impose specific requirements on data models, most of which cannot be satisfied by general-purpose models, such as the relational model or object-oriented data models. Some of these requirements involve the integration of concepts coming from different domains.

For instance, the new generation of GIS demands the integration of spatial and multimedia databases. The methodology presented in this paper addresses this problem. It is part of a more comprehensive framework for the integration of software technologies, i.e., models, languages and software systems (Montilva and Roberts, 1992; Montilva, 1995). This framework is inspired in the methodologies for database schema integration (Batini, Lenzereni and Navathe, 1986).

The integration of spatial and multimedia database systems is a complex process that takes place at three different levels: conceptual level, language level and system level . Only the conceptual level is described in this paper. The notion of *hypermap* emerges from the integration of spatial and hypermedia databases at the conceptual level. A hypermap is a hyperdocument extended with geographical references, in which the location of an object on a map is used to retrieve its associated multimedia information. The notion of hypermap is relatively new. Although several definitions of this notion are found in the literature (Wallin, 1990; Laurini and Thompson, 1992; Boursier and Mainguenaud, 1992; Montilva, 1993), no data models have been proposed yet. The motivation behind the integration methodology presented here is therefore to design a hypermap data model. This model can be used for designing and implementing hypermap database systems, as well as for modelling hypermap applications in GIS. The methodology is organized into four phases: pre-integration,

* *Proceedings of INTERSYMP95 - 5th. International Symposium on Systems Research, Informatics and Cybernetics.* (Baden-Baden, Alemania, Agosto 16-20, 1995). InterSymp/ISAS. pp. 121-125.

conceptual analysis, conceptual comparison, and conceptual integration. Each of these phases are explained separately.

## THE INTEGRATION METHODOLOGY

The purpose of the integration methodology is to guide the process of designing a new specific-purpose data model by integrating two or more existing models. The phases of the methodology and the flow of information between phases are shown in Figure 1 using a data-flow diagram. The main input to the methodology is a set of database requirements imposed by the intended application domains for which an integrated model is designed. The output, on the other hand, is a design specification that describes the concepts supported by the integrated model and its main components: constructs, operations, and rules.

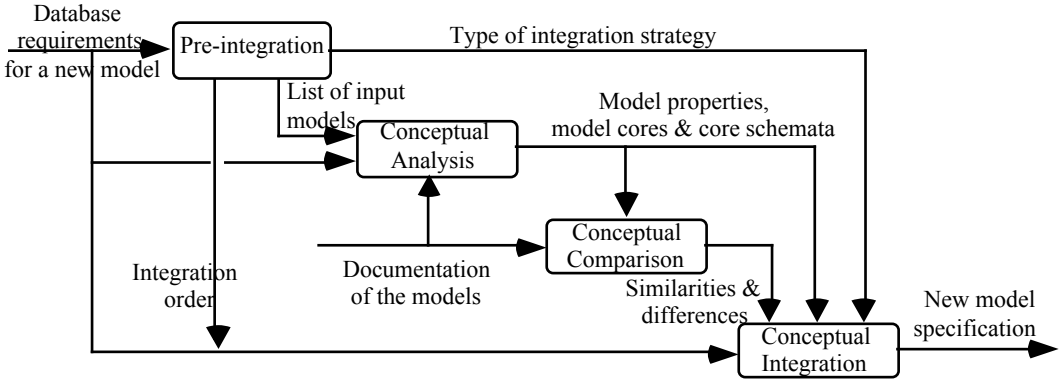


Figure 1. The Integration Methodology: Phases and information flow

The steps of each phase are enumerated in Table 1. Like most software process models, the integration methodology entails a great deal of iteration between its phases and steps. The integration methodology presupposes that a sequence of iterations of its activities may be required before the final integrated model is achieved.

| PHASES | STEPS |
|---|---|
| PRE-INTEGRATION | Selecting the models to be integrated |
| | Selecting the integration strategy |
| | Selecting the order of integration |
| CONCEPTUAL ANALYSIS | Identifying the model's properties |
| | Defining the model cores |
| | Modeling the cores |
| CONCEPTUAL COMPARISON | Comparing the views of the world |
| | Comparing the concepts |
| | Comparing the core schemata |
| CONCEPTUAL INTEGRATION | Solving the conceptual conflicts |
| | Identifying unification points |
| | Integrating the model cores |
| | Validating and refining the integrated model |

Table 1: Steps of the integration phases

## PRE-INTEGRATION

The *pre-integration phase* is concerned with the selection of: (1) the data models to be integrated, called here *input models*; (2) the integration strategy (or strategies) to be used; and (3) the order of integration to be applied, if more than two models are to be integrated.

The selection of models for integration is led by a set of database requirements. This set establishes the needs, in a given application area or realm, for a new specific-purpose data model. An analysis of the requirements may help to determine, first, the domains from which the models will be selected and, second, the specific models to be used in the integration. In the hypermap example, the database requirements were defined based on the work of (Frank, 1991; Laurini and Thompson, 1992 and Wallin, 1990). Two specific domains were identified: object-oriented spatial databases and hypermedia databases. From the first domain, we selected the object-oriented spatial data model of Roberts et al. (1991). For brevity, we refer here to this model as the *OO-GIS model*. In the second domain, we used a hypermedia reference model, called here *the HMR model*. (Montilva, 1993). It is based mainly on the Lange's data model (Lange, 1990) and is extended with some

features drawn from the Dexter model (Halasz and Schwartz, 1994).

Three types of strategies can be used for designing integrated models: extension, combination, and transference (Montilva 1995). Creating an integrated model by *extension* involves enhancing an existing model by adding concepts, constructs or operations from another one. *Combination* involves the merging of concepts, constructs and operations from two given models to create a new one. *Transference* implies the adaptation of a concept, from a well-known domain or discipline, and its incorporation into a given model. In the design of the hypermap data model, combination was used as the main strategy to integrate the OO-GIS data model and the HMR model. This strategy produces a more comprehensive integration than the other ones. The resulting model displays the properties of each input model, as well as the emergent properties that arise from the merging of concepts.

When more than two models are participating in the integration endeavor, deciding the order in which they will be integrated becomes imperative. Only two models are integrated at a time. The result of the first integration is then combined with the third model, and so on. Which models will be integrated first should be previously decided based on the requirements.

## CONCEPTUAL ANALYSIS

The purpose of this phase is to help the model´s designer to gain a good comprehension of each model being integrated. This is achieved by: (1) identifying the properties of each input model; (2) selecting a relevant subset or core of each input model; and (3) modeling or representing the cores using an appropriate modeling notation. The result of the representation process of each core is referred here to as *core schema*.

**Identifying the properties of each model**

A data model may be analyzed based on three properties: (1) the paradigm(s) used for its design and its associated view of the world (i.e., the way of looking at the world used by the designer and users of the model), (2) the concepts that it supports, and (3) its three components: constructs, operations, and modeling rules.

The OO-GIS model assumes an object-oriented view of the world. The HMR model, on the other hand, is based on a graph theory view. A hyperdocument is seen as a graph G(N, L), where N is a finite set of information nodes and E is a finite set of arcs called links. Each link connects two information nodes of N. Table 2 shows the meaning of the different constructs of both model with regard to an ontological view of the world proposed by M. Bunge (1977). In the OO-GIS, a domain entity, for example, is modeled using an object whose attributes capture the entity´s properties. Actions and events in the world being modeled are captured by the constructs methods and messages respectively. In the HMR model, for instance, each information node is used to represent information about entities or concepts of the application domain. By aggregating multimedia items (e.g., textual items, graphical items, images, audio clips and video clips) in a node, we can represent the properties of the entity or concept being modeled. Active items (i.e., a script or program that is executed when its associated button is activated) are used to represent actions to be performed when events arise.

| Element of the View | OO-GIS Construct | HMR Construct |
|---|---|---|
| Domain object: entity or concept | Object or Instance | Information node |
| Kind | Type | - |
| Property | Attribute | Item, Link |
| State of a thing | Attribute´s values | Item´s content |
| Action | Method | Active item |
| Event | Message | Button , Active link |

Table 2. The meaning of the OO-GIS and HMR constructs

Each model is based on a set of concepts. The OO-GIS model, for example, is founded on object-oriented concepts such as object identity, abstract data types with encapsulation, aggregation, generalization hierarchy with inheritance and late binding. The HMR model, on the other hand, is based on the notions of hypertext, hypermedia, aggregation, associative linking, organizational linking, active linking and navigation.

The OO-GIS model is an extension of an object-oriented data model. Besides the typical object-oriented constructs (i.e., object, class, attribute, method and message), this model provides a collection of specialized object classes for supporting GIS vector and raster-based applications. These classes are the following: Spatial_Object, Spatial_Representation, Map, Poligon_Overlay, Quatree_Overlay and Feature. Each class provides a set of operations for creating and manipulating their instances.

Similarly, the HMR model provides a set of constructs and operations for creating, manipulating, browsing and maintaining hypermedia databases. The constructs provided by this model are the following: hyperdocument, information node, item, link, button and anchor.

**Defining the core of each model**

A core should be defined for each input model. A *model core* is a manageable subset of a model that includes a subset of the concepts supported by the model and a subset of the components of the model including constructs, operations, and modelling rules. A model core is made of those properties of the model that can be used to satisfy the database requirements in the intended application domains of the integrated model. The set of requirements is used here to determine whether a concept, a construct, or an operation of a model should be included in its core or not.

The OO-GIS model core included only those object types required to support a vector-based representation of spatial data. Spatial_Object, Spatial_Representation, Map and Feature classes were chosen as the basic constructs needed to model hypermaps and network-based GIS applications. Except for the construct anchor, the HMR model core included all the constructs of the input model.

**Modelling the cores**

The comparison and integration of the model cores in the next two phases of the methodology are based on the schemata of the core models. A *core schema* is a meta-model or representation of a model core expressed in terms of a particular notation, such as a knowledge representation scheme, an object-based design notation, or a specification language.

For modelling the core of the OO-GIS and HMR models, we applied the Object Modeling Technique (Rumbaugh, 1991). It was chosen because of its ability to represent not only the structure of a construct, but also its associated operations. A meta-model or core schema, expressed in the OMT notation, was built for each model core. For space reasons, only a subset of each core schema is shown in Figures 2 and 3.

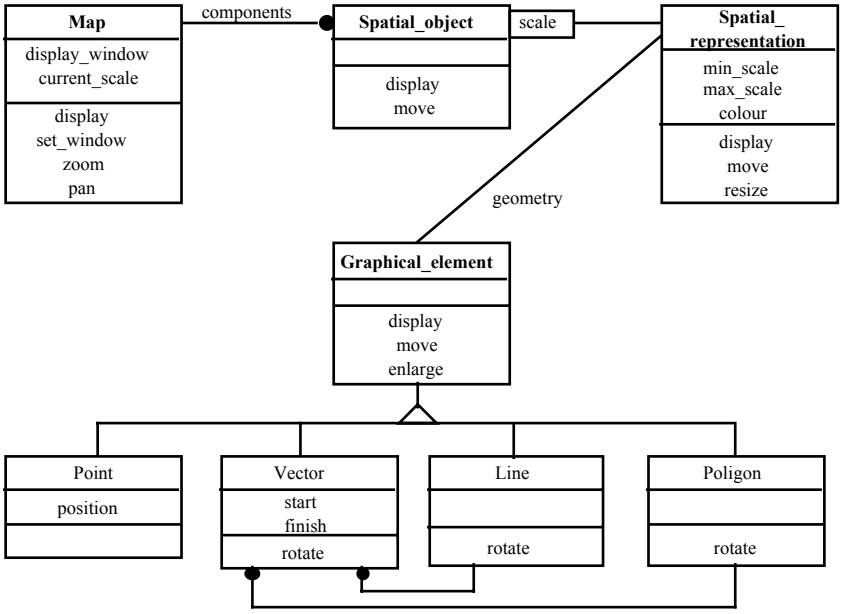


Figure 2. The schema of the OO-GIS model core

## CONCEPTUAL COMPARISON

The *conceptual comparison phase* involves the identification of similarities and differences between the concepts and components of the model cores. The differences could lead to conceptual conflicts that need be solved during the conceptual integration phase. The conceptual comparison should make a good account of these conflicts by comparing the different elements of the cores. The similarities are used in the next phase as unification points for integrating the model cores.

**Comparing the views of the world**

The similarities and differences between the views of the world assumed by the input models should be determined firstly. This comparison can be more comprehensive if a unique and global view of the world is used as a framework of reference. That is, instead of comparing each view with the other, the comparison is performed against the global view. To compare the OO-GIS and HMR models we used the ontological view described in the previous section (see Table 2). This comparison favours the first model because its object-oriented paradigm and view of the world are closer to the global view.

**Comparing the concepts**

The concepts supported by these models are compared next. Only those concepts encompassed by the cores should be compared. Similar concepts should be identified. Their differences should be determined based on their definitions and the way they are supported by each model.

**Comparing the core schemata**

Finally, the core schemata should be compared. Particularly important for the integration of core schemata in the next phase is to determine the semantic and structural equivalencies, similarities, and dissimilarities between the constructs of the cores to be integrated. Another source of commonalities and differences between constructs is due to their names. If the same name is used to designate two dissimilar constructs, the names are referred to as homonyms. Whereas if two different names are used to designate two equivalent or similar constructs, these names are called synonyms.

The semantic equivalence between two constructs is concerned with the meaning of these constructs. It is determined by: (1) finding the *denotational* relationship of each construct with regard to a common view of the world; and (2) comparing these relationships. The semantic equivalence between the constructs of the OO-GIS and HMR models is shown in Table 2. Instances and information nodes, for example, are semantically equivalent, since both are used to represent domain objects. However, they cannot be regarded as structurally equivalent or equivalent constructs, because it is not possible to find a 1:1 correspondence between their space states, as shown in Figures 2 and 3.

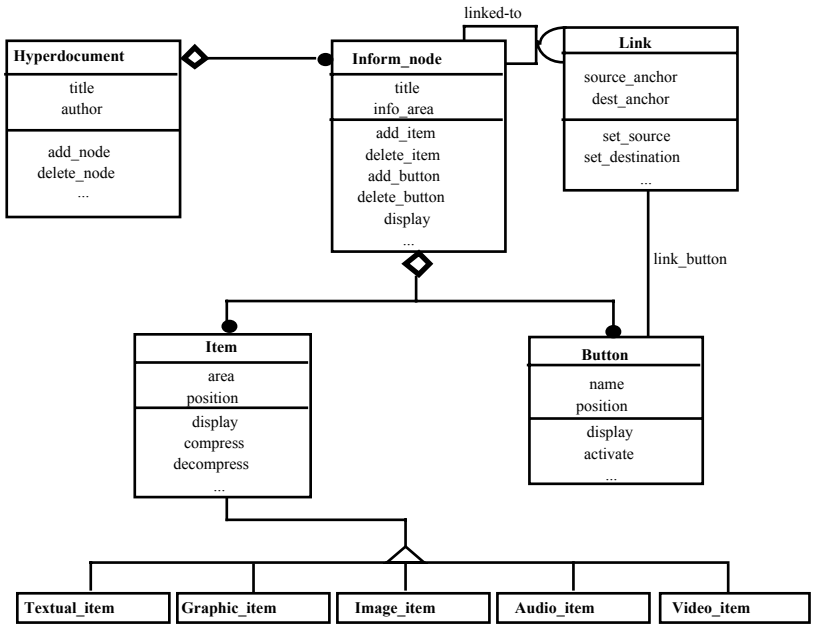


Figure 3. The schema of the HMR model core

## CONCEPTUAL INTEGRATION

In this final phase, the model cores are merged following the selected strategy. Before integrating the core schemata, conceptual conflicts betwwen model cores are conciliated. A set of unification points, elements for unifying two core schemata, are defined. The resulting integrated model is finally validated against the given set of requirements and refined by iterating the process until an appropriate solution is found.

**Solving conceptual conflicts**

The conflicts between the model cores that were found during the conceptual comparison should be resolved here. Possible sources of conflict are the differences between the paradigms and view of the world supported by the input models. Some conflicts may be caused by the way in which a concept common to two or more model cores is supported or implemented in the models. Other possible conflicts are (a) naming conflicts (synonyms or homonyms between two concepts, constructs and operations); type conflicts (e.g., two different constructs are used to represent the same domain object); and semantic conflicts (e.g., a common construct or operation has a different semantic in each model).

**Identifying unification points**

The next step in the conceptual integration phase is to find the ways of unifying the constructs. The complexity

of the integration of core schemata can be simplified if different ways of unifying the constructs are previously determined. We refer to them as unification points. A unification point is a concept or construct that can be used as the linking element for integrating the constructs of two core schemata.

The unification points are identified by determining how the concepts and constructs of one of the core schemata can be incorporated into the other one. In the design of the hypermap model, we proceeded as follows. First, we used object orientation as a global unification point to solve the differences between the paradigms, as indicated in the previous section. Second, we had to choose between two different ways of unifying the OO-GIS and HMR schemata. The first alternative involved to incorporate into the construct Map, as defined by the OO-GIS schema, a set of buttons for associating multimedia items or information nodes to the different spatial objects that compose a map. A second possibility was to consider a hypermap as a hyperdocument in which maps are defined and manipulated as a multimedia item. We chose the latter one because it is more comprehensive and embraces the first one. Two specific unification points were used for integrating both core schemata, as shown in Figure 8. The first one was to use the construct *Item* as the glue to merge maps and information nodes. In this way, a map may be seen as a new type of item, which can be used in the composition of information nodes. The second point was to associate new types of buttons to the spatial objects of a map item. The name or geometry of spatial object can be defined as the area or shape of a button.

**Integrating the core schemata**

The three components of the model cores -- constructs, operators, and rules -- should be integrated next following the course of action indicated by the selected strategy. The unification points are used here as the basis for integration. Since the conceptual conflicts have already been solved, the features of the cores being integrated in this phase are assumed to be compatible and mergeable. Using the core schemata, the process of integration could be performed in a similar way to that used in database view integration to merge DB schemata. The detailed step to be applied for integrating core schemata depends on the notation applied for building the schemata. If the Entity-Relationship Model is used, then the merging rules described by Navathe and Elmasri (1996) can be applied to integrate the constructs. The integration of operations and rules are considered by this notation. They should be integrated separately. The rules described by Qutaishat, et al. (1992) can be used for the integration of constructs and operations when OMT is used.

Figure 4 illustrates the outcome of the OO-GIS and HMR core schemata integration. Attributes, operations and low-level subclasses were omitted from the figure for brevity. According to the resulting schema, a hypermap database is defined as a collection of spatial information nodes relates by links. Each spatial information node is a composed object whose components are maps, multimedia items, and buttons. A map is a type of item that preserves all the spatial structure and behaviour of the map objects, as in the OO-GIS data model. This feature is the major distinction between a hypermap and some hyperdocuments that display maps as simple images. The hypermap allows the user to manipulate interactively the spatial objects associated with its maps. The spatial analysis operations that characterize a geographical information system are provided by the maps that conform a hypermap. Two new types of buttons emerges from combining the OO-GIS and HMR core schemata. The first type of button, called Sp_Name_Button, uses the spatial object's name to associate the object with another information node. The second one is associated with the spatial representation of a spatial object. The shape of the button is the geometry of the object. The button is activated by clicking on the geometry of the object.

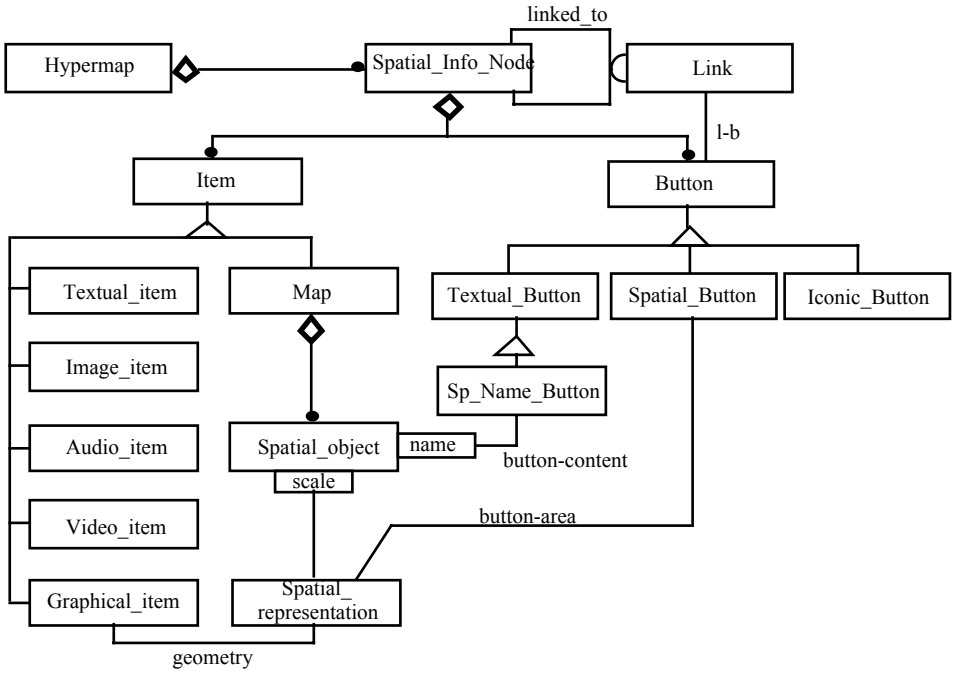


Figure 4. The schema of the hypermap data model

**Validating and refining the integrated model**

Finally, the integrated model should be validated using the database requirements given as input to the integration process. On the basis of this validation, the model should be refined or enhanced with other concepts or components. This refinement allows the designer to cope with those requirements that were not satisfied by the integration effort.

## CONCLUSIONS

An integration methodology for the design of specific-purpose data models was proposed and used to illustrate the design of a hypermap data model. This methodology assumes that the design of a new model should be based on the conceptual integration of existing models. The methodology can also be applicable to the integration of databases, knowledge bases and programming languages. It is particularly useful in domains such as intelligent databases, expert database systems, knowledge base management systems, semantic database modelling, hypermedia systems, object-oriented databases, and deductive databases.

Besides the design of a hypermap data model, the methodology has proven to be useful in two other cases. It was used for designing an object-oriented data/knowledge model for an intelligent database system (Montilva, 1994) and a spatial data model for the representation and manipulation of graph-based objects in a 2D space. In both cases, the methodology demonstrated to be valuable in reducing the complexity involved in the integration of different models. It introduced a systematic order to the activities required to integrate the models. It proved to be very useful in determining what, how and when to integrate. We cannot claim that the methodology, as described here, is complete and fully proven. More work is still required to deliver the methodology as a tested instrument for the design of special-purpose and integrated models. However, it is a valuable research tool for investigating the process of integration of software technologies at a conceptual level.

## REFERENCES

Batini, C., Lenzereni, M., and Navathe, S.B. (1986); A Comparative Analysis of Methodologies for Database Schema Integration. *ACM Computing Surveys*, Vol.18, No.4 (pp. 323-364)

Boursier, P. and Mainguenaud, M. (1992); Spatial Query Languages: Extended SQL vs. Visual Lanuguages vs. Hypermaps. 5th Int. Symp. on Spatial Data Handling, Vol. 1.

Bunge, M. (1977);*Treatise on Basic Philosophy, Vol. 3: Ontology I: The Furniture of the World*. Reidel.

Frank, A.U. (1991); Properties of Geographic Data: Requirements for Spatial Access Methods. In Gunther, O. and Schek, H.-J. (Eds), *Advances in Spatial Databases*, Springer-Verlag (pp..225-234)

Halasz, F and Schwartz, M. (1994); The Dexter Hypertext Reference Model. Comm. of the ACM. Vol. 37, N.2.

Lange, D.B. (1990) A Formal Model of Hypertext. In Moline, J., Benigni, D., and Baronas, J. (Eds.) *Proceedings of the Hypertext Standarization Workshop*, National Institute of Standards and Technology,.

Laurini, R. and Thompson, D. (1992); *Fundamentals of Spatial Information Systems*. Academic Press, London.

Montilva, J.A. and Roberts, S.A. (1992); *A Methodologyological Framework for the Integration of Data, Information, and Knowledge Management*.. Research Report No. 92-20, University of Leeds, School of Computer Studies, Leeds, UK, September.

Montilva, J.A. (1993);*An integration methodology applied to the design of a data/knowledge model for multi-media and spatial applications*. Ph.D. Thesis. University of Leeds, School of Computer Studies, Leeds, UK, January.

Montilva, J.A. (1994); A formal definition of an object-oriented data/knowledge model. *Proc. of the IV Ibero-american Conference on Artificial Intelligence*. Caracas. October.

Montilva, J.A. (1995); A methodological framework for the integration of software tools in intelligent information systems. *5th Int. Symp. on Systems Research, Informatics and Cybernetics.* Baden-Baden, Germany, August, 1995.

Navathe, S., Elmasri, R., and Larson, J. (1986); Integrating User Views in Database Design. *IEEE Computer*, Vol.19, No.1, January (pp.50-62)

Qutaishat, M.A., Fiddian, N.J. and Gray, W.A. (1992); Association Merging in a Schema Meta-Integration System for a Heterogeneous Object-Oriented Database Environment. *Proc. of the BDCOD*. UK. Abeedeen. July (pp. 210-226)

Roberts, S.A., Gahegan, M.N., Hogg, J., and Hoyle, B. (1991); Application of object-oriented databases to geographic information systems. *Information and Software Technology*, Vol.33, No.1 (pp.38-46)

Rumbaugh, J. et al. (1991); Object-Oriented Modeling and Design. Prentice-Hall, NY.

Wallin, E. (1990); The Map as Hypertext. Proc. of the 1st European Geographical Information Systems Conf. Amsterdam. April (pp. 125-134).