Chapter 5

Conceptual design

Chapter 3 clarified user requirements for 3D GIS for a municipality, and delineated the scope of the thesis regarding the type of real objects, mutual relationships and possible format of outcomes. Chapter 4 presented the approach for visualisation and data access via Internet. An extended discussion on possible methods to retrieve and edit data has clarified specific requirements related to the presented approach (referred to as visualisation requirements). Some of the user requirements are already accomplished by the concepts accepted by VRML and the principles of the client-server architecture proposed, e.g. remote access, real-time navigation and exploration, ability to represent highly realistic models, query and modification of spatial and non-spatial objects, a GUI familiar to the user. Yet, a number of user and visualisation requirements have to be resolved at the conceptual level. The co-ordination between all the types of data as well as the synchronisation of the "query-response" process according to the visualisation approach stimulate the introduction of specific parameters and influence data organisation.

This chapter focuses on structuring the data needed to represent the objects and their characteristics significant for the municipality governance. For this purpose, first the requirements derived in Chapters 3 and 4 are summarised. Second, a generalised definition of objects is proposed, which provides a framework for structuring the information collected per object, regardless of the type of object. The data per object are distinguished on the basis of their thematic and geometric origin. Under the assumption that the objects with spatial extent are the more sophisticated for organisation, further elaboration is provided only in the geometric domain. Three spatial topological models are assessed for their suitability to 1) represent spatial objects and spatial relations, and 2) ensure sufficient data for correct visualisation in a short period of time. Motivated by the advantages and disadvantages, a new spatial model is formulated in Section 5.5.

5.1 Summarised requirements for 3D GIS on the Web

The requirements outlined in the previous chapters refer to different aspects of a 3D GIS, i.e. modelling, analysis, visualisation and Web access. In accordance with the objectives of the research, i.e. an integrated conceptual model, the requirements have to be considered in their completeness. In other words, the organisation of data in the database (see Figure 4-7) must be appropriate for both 3D GIS analysis (thematic and spatial) and composition of documents (HTML and VRML). Recalling discussions from the previous chapters, we can summarise the requirements of the data and their structuring as follows:

- **Integration of spatial and non-spatial objects:** The user has to be able to visualise (query, explore) records with personal data or browse documentation, or perform

spatial analysis, or query spatial objects, i.e. request any information with a uniform, standard operation. For example, the department of social services may need to see the apartments of all the habitants older than 75. The query can be completed only if the personal records and the spatial information about buildings are linked.

- **Maintenance of radiometric characteristics:** Similar to the geometric characteristics, the radiometric characteristics are substantial for visualisation. VRML builds the scene utilising radiometric properties of objects, i.e. shading model, colour, material and texture (recall Figure 2-3). The investigation into realistic texturing has revealed a great interest in true representation of radiometric properties (i.e. texturing with photo images). In most cases, the user does not use them for particular activities, but relies on them for better orientation in the model.
- **Maintenance of 3D spatial relationships:** The scope of relationships derived from user requirements has exposed preferences for describing adjacency, belonging and inclusion. Since the topology is the most appropriate way to encode such relationships, the initial set-up aims at furnishing 3D topology.
- **Information about behaviour:** Virtual reality techniques permit description of complex movements, functions and dynamic interrelation among objects in virtual worlds. Typically "games-driven", the issue gains popularity among users as a tool for exploration and better perception of complex 3D worlds. Some dynamic behaviours might be so typical of real objects that their permanent description in a database can be encouraged. For example, the very natural behaviour of a door is to be able to open. Apart from interest to the user, the storage of behaviour is of particular importance for the visualisation approach presented in Chapter 4.
- **Fast traversal of the database:** The waiting time at the client station is a critical issue for data retrieval over Internet. Among the variety of factors (hardware, software, Internet communication lines) influencing the performance, this thesis focuses on the optimisation of the model for fast retrieval of data.
- **Ability to operate with composites of objects:** Most commonly, different users need different abstractions of an object or group of objects, e.g. a user may be interested in an object "building" while another user could be satisfied with an object "neighbourhood". The creation and maintenance of composite objects is a rather broad issue, which requires special attention. The next sections will present some initial ideas.
- **Ability to create different geometric representations and LOD:** LOD for fast visualisation may have a different meaning to the geometric representations required by the user. Objects remote from the viewer in the 3D scene do not need details and can be automatically substituted by the visualisation software with less detailed representations (see Chapter 2). The geometric representations meant by the user support the user's tasks and, commonly, are not related to the position of the viewer. For example, buildings represented by their outlines are sufficient for a telecommunications company but insufficient for a utility company. Such LOD, however, require generalisation techniques (see Peng 1997), which are outside the scope of this thesis. Here, the concentration is on LOD for visualisation.

5.2 Framework for object identification and information structuring

The categorisation of real objects given in Chapter 3 as well as the approach for visualisation presented in Chapter 4 make claim for an extended object definition, capable of describing various characteristics of real objects.

5.2.1 Objects: spatial and non-spatial

The object definitions in geo-sciences (see Chapter 2) focus on spatial objects, with their geometric, radiometric properties, semantics, spatial relationships and time. As shown, our study requires broader understanding dealing with spatial and non-spatial objects. Therefore, we will start with general concepts applied in business processes, and will later make specifications for a spatial object. Among the common object-oriented approaches to identifying objects, we have chosen the one proposed by Coad because of the common notations for objects, responsibilities and scenario. The initial statement in Coad's definition, "the object can be anything: feature, action, process, which is of interest for the user", can be successfully applied to the variety of real objects of interest already identified. The object responsibilities and time-related component (scenario) can be utilised to complete a broad characterisation of any real object. Chapter 3 has used the basic principles of this OO approach to clarify objects of interest. This chapter applies the same principles to derive an extended definition of an object capable of describing spatial and non-spatial objects (see also Zlatanova and Gruber 1998).

An object **O** can be represented by two components **OR** (object responsibilities) and **S** (scenario): **O (OR, S)**. The brackets here are notations for the expression *consist of*, i.e. the notation **O (OR, S)** has to be read *an object consists of object responsibilities and scenario*. Thus, considering the groups of real objects introduced in Chapter 3, we can distribute the information maintained in current information systems according to the meaning of *object responsibilities* and *scenario* (see Table 5-1).

Table 5-1 Object responsibilities and scenario of real objects

Groups of real objects	What the object knows about itself	Who the object knows	What the object does	Scenario
Juristic	Attributes	Interrelations	Functions, Operations	Archive
Topographic	Shape and/or size, location, attributes	Spatial relationships	Functions, Operations	Archive
Fictional	Shape and/or size, location, attributes	Spatial relationships	Functions, Operations	Archive
Abstract	Attributes	Interrelations	Functions, Operations	Archive

Coad's three questions refer to characteristics of objects, which here will be called *attributes*, *relationships* and *behaviours*. *Attributes* **(A)** comprise the characteristics, which identify the object on the basis of personal properties. *Relationships* **(R)** represent interactions (mostly static) of the object with other objects. *Behaviours* **(B)** refer to the dynamics (functions) of objects or dynamic interaction with other objects. *Scenario*

represents the dynamics of objects with respect to the absolute time (days, months, years, etc.). Thus we write the three components of **OR** as:

OR (A,R,B)

The substitution of the **OR** components in the notations for an object **O** will give us the full set of components describing an object, i.e. *attributes, relationships, behaviour* and *scenario*:

O (A,R,B,S)

The preliminary, still generalised, notion of an object provides the first classification of the information per object. For example, the existing records about a person, a building, a district and a document can be mapped into the four components as follows:

Table 5-2: An example for a classification of the data per object

Object	Components	Existing information	Extended information
Person	Attributes	PIN, name, address, marital status	
	Relationships	Living house, agricultural land	
	Behaviour	CheckIn, CheckOut	Lessons in music,
		Select, edit, delete, add data	Web page
	Scenario	Thee changes of the address	
Building	Attributes	ID, hotel, made of bricks,	
		Address, size, shape, position	
	Relationships	Attached to the building of	
		the theater, part of chain of hotels, owner	
	Behaviour	Select, edit, delete, add data, show owner	Sightseeing from
			the roof
	Scenario	Building is reconstructed four times,	
		last used as a hospital	
District	Attributes	ID, district (centre), position, size, shape	
	Relationships	Neighbour districts	
	Behaviour	Select, edit, delete, add	Provides
			statistic information
	Scenario	Boundary archives	
Tax document	Attributes	ID, building tax, car tax, dog tax	
	Relationships	PIN of the payer, address of the payer	
	Behaviour	TaxPaid	
		Select, Edit, delete, add	
	Scenario	Records of each year	

While acceptable for non-spatial objects, such classification of data is not sufficient to distinguish between semantics and geometry of spatial objects: 1) geometric and thematic characteristics are united behind attributes, e.g. the position of a building is together with the usage and 2) the spatial relationships are maintained together with thematic relationships. Therefore, we will further elaborate on components in the *geometric* **(GD)** and *thematic* **(TD)** *domains*:

O (GD, TD)

We introduce *attributes, relationships, behaviour* and *scenario* of spatial objects in the thematic and geometric domains:

GD (GA, GR, GB, GS)
TD (TA, TR, TB, TS)

where **GA, GR, GB, GS** - *geometric appearance* (will be discussed), *geometric relationships, geometric behaviour, geometric scenario;* **TA, TR, TB, TS** - *thematic attributes, thematic relationships, thematic behaviour, thematic scenario*

Thus the components of an object can be written as:

O ((GA, GR, GB, GS), (TA, TR, TB, TS))

While the thematic component is compulsory, the geometric one is optional per object. That is to say, if the geometric components do not exist, the object can be maintained only according to its thematic description. For example, documents or people commonly do not have geometric representations (see Chapter 4 for recent research). Similarly, not all the components within one domain are obligatory. For example, the geometric domain may be represented only by **GA** and **GR** or even only by **GA** (shape, size, position and colour of objects but not relationships). In general, the information that is maintained in current GISs corresponds to the information represented by the components **GA**, **GR** and **TA**, i.e. shape and position of spatial objects, spatial relationships and thematic attributes.

The components of objects with a clear differentiation between thematic and geometric information contribute to: 1) facilitation of the information structuring per object and 2) integration of spatial and non-spatial objects in one information system. The benefit for data organisation is threefold:

- Classifications can be introduced in any of the components (see Figure 5-1). Frequently, thematic and geometric properties are used to create classes or composites of objects. The common principles for building hierarchy, however, are different. While objects can be *associated* with a new *class* based on a thematic property, they can be only *aggregated* as *parts* to a new *object* if the geometry is focused (see Chapter 2). Pilouk 1996 proposed an object-oriented procedure for object creation, assuming predefined thematic and geometric hierarchy. The approach can be extended to combine the behaviour of objects.
- The components can be freely substituted with new representations or the hierarchies can be modified, and all this independent of the other components. For example, the user may wish to switch from one geometric representation (e.g. boundary representation) to another (e.g. voxel representation). In this case, the modifications in the integrated database will reflect only two components, i.e. **GA** and **GR**.
- Different associations between hierarchies permit a multi-resolution description per object to be organised. For example, an object called "building" can be represented by a "box" (i.e. $\mathbf{GA_1}$) in **GD** and can have the properties of an administrative

building as **TA** in **TD**. Another application, however, can require the same building with the same **TA** to be represented as a point (i.e. $\mathbf{GA_2}$).

Inside the thematic and geometric components, spatial and non-spatial objects can be organised in an integrated database. The attributes, relations and behaviour of non-spatial objects will be limited to the thematic domain until the user introduces geometric description.

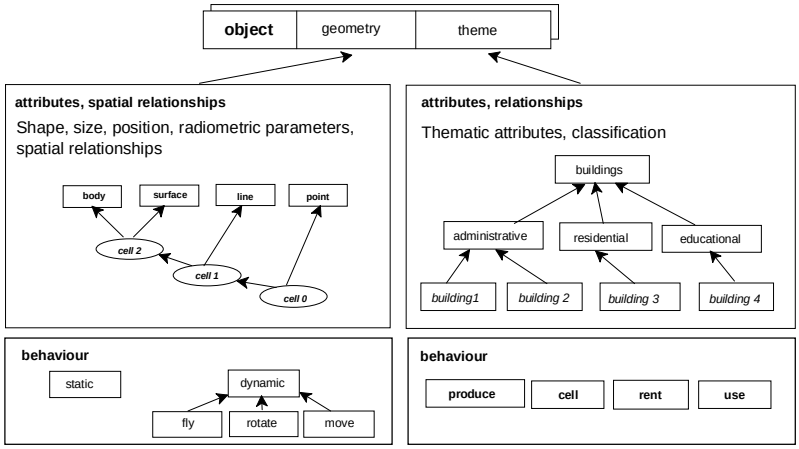

Figure 5-1: Hierarchy in different object components

5.2.2 Object components in the geometric domain

The components in thematic domain **(TD)**, are not further explored because of 1) their high dependence on particular user requirements, which were not investigated and 2) a variety of approaches and methods to structure semantic information (Norman, 1996). Consequently, we will preserve the detailed notation to the geometric domain, with an indication that, the thematic domain has to be considered as well, i.e.:

O((GA, GR, GB, GS), TD)

5.2.2.1 Geometric appearance (GA)

The component **GA** is referred to here as *geometric appearance* (not *geometric attributes*). The more complex meaning of the attributes in the geometric domain motivates the introduction of another term. The information about geometric characteristics (i.e. shape, position and size) and radiometric characteristics (i.e. reflectance) are intended to be represented by this component. As mentioned above, the 3D visualisation process needs these properties to create a 3D scene. Since they make feasible the *appearance* of the object in the scene, the term *geometric appearance* is used. The shape, size and position are implicit properties dependent on the manner of geometric description chosen, (e.g. vector representations, CSG, raster representations) and the abstraction principle applied. Variations can be quite significant. Colour, texture, material are determined by some

physical properties (e.g. material used for covering roofs) of real objects and are not influenced by the geometric representation, i.e. they are explicit properties. For example, the roof of a building might be represented by red colour regardless of the geometric representation, e.g. a cone or a set of triangles. Therefore we introduce two new components: *geometric description* **(GDsc)** and *geometric "attributes"* **(GAtt)** as part of **GA**, i.e.

GA (GDsc,GAtt) => O (((GDsc, GAtt), GR, GB, GS), TD)

The component **GDsc** addresses shape, size and position and the **GAtt** is responsible for radiometric properties. The notation geometric attribute is introduced to indicate that this is a "personal" characteristic of the object (i.e. belongs to the group attributes) in the geometric domain.

Despite the three dimensions of every object, the modelling process still requires certain abstractions of real objects to be built. The historical human experience with maps and 3D CAD models has contributed to the establishment of four abstraction types of objects, i.e. points, lines, surfaces and solids. We will use the terms *point, line, surface and body* and will give them the common notation *geometric objects* **(GO)**. The next distinction is between *geometric objects* and *constructive objects* **(CnsO)**. *Geometric objects* are elementary *nD* objects (*n = 0,1,2,3*), which can be associated with thematic meaning, while *constructive objects* are used to compose *geometric objects.* They represent either shape and position or size and position of **GO.** For example, a house represented as *body* **(GO)** can be built of many *cubes* **(CnsO)** with different sizes and positions in the space. The same house can be built of many *faces* **(CnsO)** with different shapes and positions in the space. Although many spatial models use different **CnsO**, the geometric objects in 3D space are usually four, e.g. 3D FDS, TEN and the cell model (see Section 5.3).

The component *geometric description* is a function of *constructive elements:*

GDsc(GO[CnsO])

Then the *geometric appearance* is represented by two components *geometric description* and *geometric attributes*, where the *geometric description* is expressed by *geometric objects* **(GO)**, which are function of *constructive objects* **(CnsO),** i.e.

GA(GO(CnsO),GAtt)

The notation of an object is extended with the components containing more detailed information about **GDsc**:

O(((GO[CnsO],GAtt), GR, GB, GS), TD)

5.2.2.2 Geometric relationships (GR)

The second component in the geometric domain deals with *geometric relationships* **(GR)** or spatial relationships. The manner of representing spatial relationships is closely related to

the method of description. If the representation in the **GDsc** component does not ensure the needed spatial relationships, some of them can be explicitly formulated, e.g. 3DFDS. **GR** and **GDsc** will be discussed in detail in Sections 5.3 and 5.5.

5.2.2.3 Geometric behaviour (GB)

The third component, denoted *geometric behaviour*, focuses on permanent repeatable behaviours of real objects (e.g. opening of a door), which are preserved in the virtual world. The way to represent such dynamics is very similar to the interactions between the objects in the real world. For example, to open a door someone has to push the handle down, i.e. someone takes the initiative to open it. The same interaction has to be simulated in the virtual world, i.e. some event has to "take the initiative" to open the door. Since the door is open, a new view appears, i.e. there is a response event. Behaviour contains parameters needed to simulate such dynamics. Furthermore some actions might be allowed to some users and forbidden to others. Identically, in the real world not everyone has the right to destroy a particular house. In this respect, a control on the permitted operations on the object during navigation and editing has to be ensured. In addition to these considerations, the access, retrieval and display of data over Internet gains from organisation of behaviour at database level (see Chapter 4).

In the light of the VRML concepts and the CGI scripting, we can distinguish the following types of *behaviour*:

Operations on geometry (OG): This type refers to permitted operations on an object such as the generic operations (see Chapter 2): 1) *deleting* (**OD**) an existing object or some of its components, 2) *updating* (**OU**) some values of components of an existing object and 3) *adding* (**OA**) a new object or a new component of an object. Operations on geometry can be presented as a set of three components **OG (OD, OU, OA)**. Further, we can specify which particular components are accessible to the user for modification. For example, we can forbid any changes in the components **GDsc** and allow changes only in **GAtt.**

Such behaviours, known as *methods*, are widely used in object-oriented programming to define different operations (see Chapter 2). The idea, here, is the organisation of behaviour at database level. Control of the operations on objects can be successfully used to protect the information on the GIS server. Since the tendency of our approach is to provide a broad range of users with access to the information, a strong security system against mistakes and unscrupulous actions has to be developed. Protection of data can be built up on two levels: server and database. The server level controls and restricts user rights to modify the data in general. The database level protects a particular object from a particular action, e.g. a building cannot be deleted by any user via Internet.

Reactions of objects to events (GE): This type of *behaviour* aims at a strategy to describe user's interactions with the object. In this context, we define two components: *initial event* (**EI**) and a corresponding *reaction* (**ER**) of an object. *Initial events*, i.e. the action that can be detected by the system and processes, which are supported by VRML, are: 1) user action (i.e. click with the mouse, drag and drop with the mouse, pass over object with the mouse); 2) absolute and relative time (i.e. some event can be initiated at a moment predefined in the VRML document, counted by an internal clock), and 3) events, caused by other applications (e.g. the display of a document, successful connection to the server, which

are detectable by a special *field values* in the syntax of VRML. The reaction can be either executing of existing HTML or VRML document on the GIS server, or running a script file (CGI, Java, etc.), or starting a predefined action (animation, rotation, shifting of object), which can be included the current VRML document. In the last case, the **ER** component needs to be refined for the parameters necessary to describe the action – for example, if we want to define: "after two clicks with the mouse start an animation showing rotating building". Some parameters, e.g. centre of rotation, can be computed from the data in the **GDsc** component, but others, e.g. speed of rotation, might be stored. The **GE** component is represented as **GE(EI,ER)**

Reactions to interactions with other objects (GI): This type of *behaviour* concerns the interaction between objects inside the model. For example, if someone has an object car and he starts to move with the car through the town, he might wish to specify what will happen if the car touches one or other building. Applying VRML, we can specify different reaction: the car could crash or pass through the object of interaction. To make possible this kind of behaviour, we define three components: *initiator*, i.e. the object causing the interaction (**IO**), *initial event* (**IE**) and *reaction* (**IR**). Then the short notation for this type of *behaviour* can be written as **GI(IO,IE,IR).**

Degree of immersion (GM): Last possible *behaviour* is with respect to detailed explorations of objects in the virtual world, e.g. entering a building or entering a room. The behaviour is essential for composite objects. For example, suppose a building is an aggregation of rooms, walls, stairs, etc. The information about the interior of the building is not necessary for a simple "walk through the town", therefore a VRML document with only the walls of the building can be created. If the user wants to enter the building, a new VRML document should be created and submitted to the client station. A possible way to display the interior of buildings is to use panoramic images and appropriate viewers. Useful information ordering the files with panoramic images can be organised in the **GM** component. Note, the component does not contain geometric or thematic information but a description (in a CGI or Java script) of how to extract the necessary data.

The complete set with all the components describing and structuring the *behaviour* of objects can be written as:

GB(OG, GE, GI, GM)
=>GB((OD, OU, OA), (EI, ER), (IO, IE, IR), GM)

In fact, the classification of *behaviour* listed above can be realised in the VRML document, applying different mechanisms that may result in combinations of some parameters at implementation level.

The last component of the **GD**, i.e. *geometric scenario* **GS**, pursues maintenance of information about geometric changes over time. For example, appropriate data and structuring can represent renovations and modifications in the shape of the building over a period of ten years, or the changes in the vegetation in a town in five years, or even what is the pollution propagation in an hour. However, the **GS** is far beyond the scope of the thesis and will not be further discussed.

Finally, all the components that participate in the description of an object (considering the elaboration in *geometric domain*) can be presented as follows:

O(((GO[CnsO],GAtt), GR, ((OD, OU, OA), (EI, ER), (IO, IE, IR), GM), GS), TD)

5.2.3 Composite objects

Research into the identity of composite objects, methods to create composites and the description of relationships between *parts* and *wholes* have been actual for GIS applications (Clementini et al 1995), object-oriented databases (Kim 1995), artificial intelligence domain (Brodie 1984) and linguistics (Cruse 1979). Issues relevant to composite objects have always been related to a variety of difficulties: 1) the new object has individual characteristics, i.e. the composite object cannot be associated with any of the composing objects, 2) the decomposition into composing pieces is sometimes impossible, 3) the principles underlying inheritance of parameters are difficult to describe. Apart from the problems, composite objects are usually maintained in graphics modellers due to:

- facilitation in dynamic modelling, e.g. to move composites relative to one other
- increase in storage economy by references to already known objects
- easy update propagation, i.e. modification of a “parent” object will be propagated to "children" object

Two basic techniques have been applied in computer graphics for creating composites of 3D cells (solids): spatial set operations (union, intersection and difference) and joining pieces along their boundaries. The first technique is more suitable for 3D objects represented as solids, while the second is more often used for surface representations. An advantage of the first method is the easy way of decomposition, and a disadvantage the impossibility to model separate faces. The second method does not usually support back partition into composing object. The complexity of the problem increases when thematic and geometric components of a composite object are considered.

With respect to the extended definition of an object presented above, a very general picture of a composite object will be drawn here. A *composite object* **(CO)** is defined as a set of *objects* **(O_i)** and *composing rules* **(Ru_i)** attached to the objects, as components in both the *geometric domain* **(GD)** and *thematic domain* **(TD)** are effected, i.e.

CO (O_i, Ru_i, TD, GD)

The *composing rules* are per object and refer to each component of the object, i.e. *attributes*, *relationships*, *behaviour* and *scenario*:

Ru (RuA, RuR, RuB, RuS)

where, **RuA** are *rules for composing attributes,* **RuR** - *rules for composing relationships,* **RuB** - *rules for composing behaviour and* **RuS** - *rules for composing scenario.*

Since the *composition rules* are different for the *geometri*c and *thematic domains*, the **Ru** component per object has to be written as:

Ru(RuGD, RuTD)

where **RuGD** and **RuTD** are *rules for composition in the geometric and thematic domains.*

=>Ru((RuGA, RuGR, RuGB, RuGS), (RuTA, RuTR, RuTB, RuTS))

Thus we can write the notation about composite object as:

$CO(O_i, Ru_i, GD, TD) => CO(O_i, (RuGD_i, RuTD_i), GD, TD) => CO (O_i, ((RuGA_i, RuGR_i, RuGB_i, RuGS_i),$
$(RuTA_i, RuTR_i, RuTB_i, RuTS_i)), GD, TD)$

Some simplification of the components can be achieved if the rules for composition are unified and the same rules are applied for all the components in a certain domain. For example, the **RuGR** component from the *geometric domain* **(GD)** can be dropped off the notations because in most of the cases the spatial relationships are related to the **GDsc**, defined on an object level.

VRML maintains composites of objects as the principles are *aggregation* (geometry, appearance and behaviour of objects), *inheritance* (transformations) and *encapsulation*. These principles are applied to create composite objects in the geometric domain (see Chapters 7 and 8). This thesis does not deal in detail with the formation of composite objects.

In summary, the object-oriented framework presented here contributes to several aspects of integration and maintenance of heterogeneous data in urban areas:

- **Integrity of information:** Large variety of objects (spatial and non-spatial) can be embedded in one conceptual model.
- **Extended analysis:** Relationships in the geometric and thematic domains usually kept separately can be integrated, and either spatial analysis or thematic analysis or combination of both can be performed in one information system.
- **Dynamics of objects:** Behaviour maintained in two domains opens room for maintenance of dynamics in the geometric domain (different from the usual time changes focused in GIS research), which is a step toward virtual GIS.
- **Classification:** Separate hierarchies in the geometric and thematic domains can be built without mutual disturbance and eventual violation of rules.
- **Geometric representations:** Geometric representation of a 3D model may be substituted with a new one, as the theme component may remain fully unchanged.
- **Interoperability:** The components defined in two domains provide a bridge for data exchange between different information systems.

The principles of the framework are successfully implemented to structure geometric and limited thematic data about two towns (see Chapters 7 and 8).

Yet the geometric description **(GDsc)** is not clarified. The type of **CnsO**, their mutual relations and the rules to construct **GO** are to be specified in the next section. The research in this area has resulted in many solutions as the terminology varies, e.g. data structures,

data schemas, spatial models, topological models (see Chapter 2). For the scope of this thesis, we will refer to the way of representing the geometry of objects by *spatial model*.

5.3 Spatial models

The major objective of this thesis is a unified spatial model that is capable of performing visualisation and spatial analysis in urban areas. As stated in Chapter 1, the strategy is the adaptation (or extension, or definition) of a spatial model, which maintains 3D topology, to perform visualisation analysis. Following this strategy, three models (two explicitly and one implicitly describing cells) will be discussed in detail. The terms introduced so far, i.e. *object*, *geometric object* and *constructive object*, will be used to unify the terms used for each model and exhibit differences, similarities, advantages and disadvantages. The comparison between the models, based on their conceptual and logical models, will be presented with respect to three aspects:

Modelling of urban data: The spatial model has to be able to resolve the geometric complexity in urban areas rather than complicate it with restrictive constructive rules. In this respect, the criterion for suitability for urban areas will be minimal partition of shapes, subdivision of the space and ability to maintain singular objects.

Visualisation: The spatial model has to provide data to create VRML documents (a list of co-ordinates, a list of faces and orientation of faces), to be able to prevent visualisation artefacts caused by rendering packages (see Chapter 4).

Performance: The spatial model has to have performance appropriate to client-server work over the Web. Some of the factors that depend on the data organisation are: the traversal of the database (operations to extract data and compose the VRML document), time for delivery of the document on the client station (size of the VRML document), time for parsing by the VR browser (types of faces: triangles, only convex or concave faces; size of the file, texture organisation), expected size of the database.

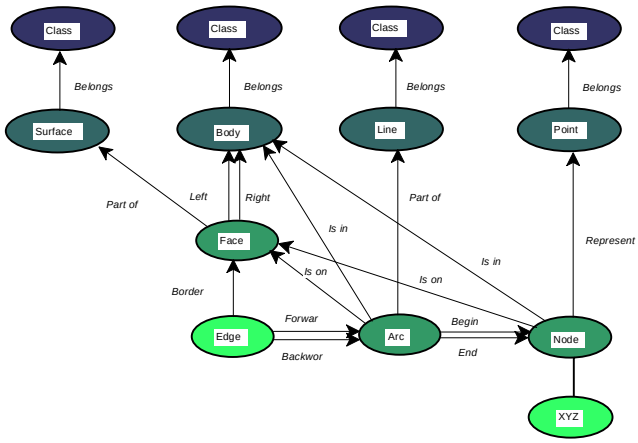

Figure 5-2: 3D FDS: conceptual model

5.3.1 3D FDS

FDS is represented by a conceptual model (see Figure 5-2) and 12 conventions (see Molenaar 1990). The model consists of three fundamental levels: *feature* (related to a thematic class), four *elementary objects* (*point*, *line*, *body* and *surface*) and four *primitives* (*node*, *arc*, *face* and *edge*). Considering the definition given in previous chapter the elementary objects correspond to **GO** and the primitives to **CnsO**. According to the conventions (6 and 8), arcs and faces cannot intersect, can an arc intersect a face (convention 9). A node and an arc must be created instead. Singularities are permitted in such a way that arcs and nodes can exist inside faces or bodies. The role of the edge is dual, i.e. to define the border of a face (relationship *face-arc*) and establish an orientation for a face, which is needed o specify left and right body. The number of arcs constituting an edge is not restricted. Arcs are straight lines (convention 4) and faces are planar (convention 7). The surface has one outer boundary and may have several non-nested boundaries, i.e. may have holes or islands (convention 12). The body has one outer surface without a boundary and can have several non-nested bodies or holes (convention 12).

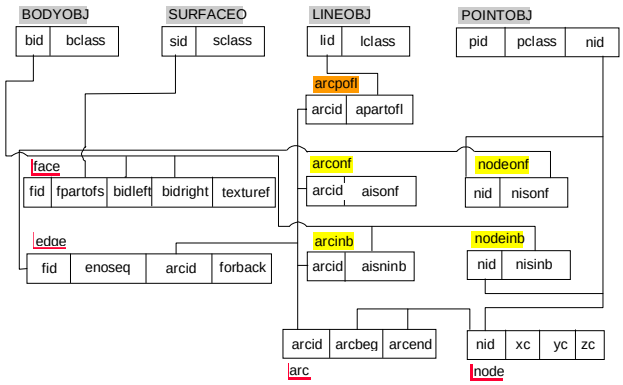

Figure 5-3: 3D FDS: logical model

The fundament of 3D FDS is the concept for a single-valued map, i.e. a **CnsO** (node, arc, face or edge) can appear in the description of only one **GO** of the same dimension (Molenaar 1989). The idea of the single-valued approach is to partition the space into non-overlapping objects (0,1,2,3 D), and thus ensuring 1:1 relationships between **GO** and **CnsO** of same dimensions, e.g. surfaces and face. **CnsO** of different dimensions can overlap, e.g. relationships *node-on-face*, *arc-on-face*, *node-in-body* and *arc-in-body* are explicitly stored.

The last basic concept is related to linking thematic class and geometry. Convention 2 imposes a thematic class to have instances **GO** of only one type, as the belonging to a class is compulsory (convention 1).

The model (see Figure 5-2) is mapped into a relational data model (Rikkers et al. 1993) and extended by Tempfli and Pilouk 1996 for texture storage. The mapping leads to 13 normalised tables (see Figure 5-3). The research reported currently has approved 3D FDS as appropriate for modelling and analysis of urban data. This section discusses the model with respect to the visualisation strategy presented in Chapter 3 (see also Zlatanova and Tempfli 1998).

In general, 3D FDS contains all the necessary data to visualise the geometry of objects. As mentioned, VRML (and the most of the rendering packages) operate with faces (triangles), which are represented by vertexes. The model also provides the orientation of faces, which is crucial for the correct rendering. Since the model has well defined objects, it can be extended with information about geometric attributes and behaviour. The disadvantages of the model focus on mainly performance issues. Since the performance is influenced by a particular implementation, the following analysis is based upon the relational mapping (see Figure 5-3).

The first concern raises from the lack of explicit relationship *face-part-of-body*, which has impact on 1) the response time and 2) the size of the database. One of the basic visualisation queries, i.e. "find all the faces composing a body" requires a double check, i.e. the fields *bodyleft* and *bodyright* has to be visited for each record. The size of the database may grow rapidly due to storage of repetitive information for some objects. For example, terrain data represented by TIN have an air body (0) to the left and an underground body (-1) to the right (see Table 5-3).

Table 5-3: An example of terrain data stored in the *FACE* table

Fid	Bodyleft	Bodyright	Fpartofs
…	…	…	…
1245	-1	0	5
1246	-1	0	5
1247	-1	0	5
…	…	…	…

The second concern is the organisation of texture. In general, one object (body or surface) can be textured with one or several images by texture mapping, and one image file can be used to drape several faces (see Chapter 4). 3D FDS is capable of keeping one or more textures per face. The textures can be single images or part of one image. Sithole 1997 proposes several different methods for storage of texture in order to facilitate dynamic loading of images from the database. One of the methods, i.e. the composition of textures needed for a surface in one image file, suits the VRML concepts (see Chapter 4). However, it is still impossible to wrap a surface or a body with one image file or to texture the face (or both its sides) with different textures. In many cases, draping with one image file is much more efficient, e.g. terrain. An indication as to which side of the face is textured might be necessary as well, e.g. two adjacent buildings with a common wall. This is a quite important issue for dynamic modelling: suppose the user wishes to see only body2, which has a common wall with body1, constituted of face1 and face2, then the wall of body2 has to have the appropriate texture (see Figure 5-4).

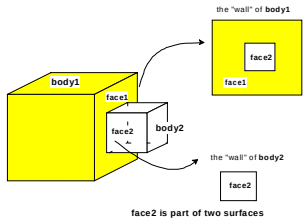

Figure 5-4: An example of a *face* needing two textures

The third concern is visualisation of lines and points. The line and point objects rendered in VR browsers have shown a low level of realism if simple lines and points are used. Tiny cylinders instead of lines, and small spheres instead of points, significantly improve the performance. Line modelling with predefined primitives (cylinders, spheres, cones, etc.), however, may often need beginning- and end-of-the-line objects to be established. For example, the visualisation of a lamppost will improve if the lamp is indicated with a small sphere. Cylinders with different diameters and cones also give better results but require indications also the direction of the decreasing diameter. The information in 3D FDS (only arc identifiers) is sufficient to construct the line object; however, the direction of the entire object is not known. Storage of the direction will speed up the process of extracting data for visualisation as well.

The fourth concern is the explicit storage of the relationships *node-on-face* and *arc-on-face*. These relationships may easily create a false impression of "sinking" in or "flying" over the face during rendering. 3D FDS allows faces with an arbitrary number of arcs, but requires their planarity. Forcing the nodes to lie exactly on a plane can ensure the planarity. The approach, however, raises a number of questions about the computation of the approximate plane, the method of node projection on the plane, the preservation of the relationships, etc., which require more investigations. The simpler method is triangulation of the face and converting it into a surface. The triangulation can be executed prior to creating the VRML document, or left to the VR browsers. Now suppose a node or an arc lies inside the face and the face is independently triangulated, a variety of "arty-facts" (e.g. an arc flying over the face, an arc crossing the face, a node below or above the face) may be observed on the screen. Pitfalls (blinking and disappearing while navigating) are observed even if the face is strictly planar and the arc lies exactly on it. Therefore, existing arcs-on-face and nodes-on-face have to be incorporated in the triangulation of the face. Since this cannot be left to the browser, intermediate algorithms are required for triangulation and control of these relationships.

The last concern is the visualisation of holes in faces. Although permitted, holes do not have a special indication in 3D FDS. Holes are stored together with the parent *face,* as the *arcs* bordering a hole have an opposite to the *arcs* bordering the face direction. Clearly, the holes can be recognised, as the necessary order can be obtained by checking the arcend/arcbegin relationship per arc in the ARC table. This operation, however, has to be performed for each arc bordering a face, which requires more sophisticated and thus slower algorithms for data extraction.

In summary, 3D FDS supplies sufficient data for rendering, and can be easily extended to accommodate data about behaviour and geometric attributes. However, the time for creating VRML documents is expected to be rather long due to the following conceptual characteristics:

- lack of explicit boundary information per body object (it effects the time for database traversing)
- storage of co-boundary relationship per face, i.e. left/right body (it effects the time for database traversing)
- maintenance of texture per face (it effects the realism of scenes)

- explicit storage of relationships arc-on-face and node-on-face (it effects the realism of scenes)
- the implicit description of holes (it effects the time for the creation of VRML documents).

5.3.2 TEN

TEN was introduced by Pilouk (see Tempfli and Pilouk 1994 and Pilouk 1996) to overcome some difficulties of 3D FDS in modelling objects with indiscernible boundaries. According to the definitions, TEN has four constructive objects (*tetrahedron*, *triangle*, *edge*, *node*). The relationship *arc-node* is given by the ARC table; the TRIANGLE table contains the *tetrahedron-triangle-edge* link. A body object is composed of *tetrahedrons*, a surface object of *triangles*, a line object of *arcs* and a point object of *nodes*. The general rule for creating the model is based on the fact that each node is part of an arc, each arc is part of a triangle and each triangle is part of a tetrahedron (see Figure 5-5 and Figure 5-6). Singularities are not permitted. The model is appropriate for representing irregularities in the real world, such as terrain, soil, air, geological objects, etc. 3D man-made objects are embedded as 3D FDS features in TEN (see Pilouk 1996). Since the model uses the simplex-complex concept (see Egenhofer and Herring 1992), TEN can be expected to cover the scope of possible topological relations in 3D space. Pilouk 1996 reported series of positive results concerning the construction of the model.

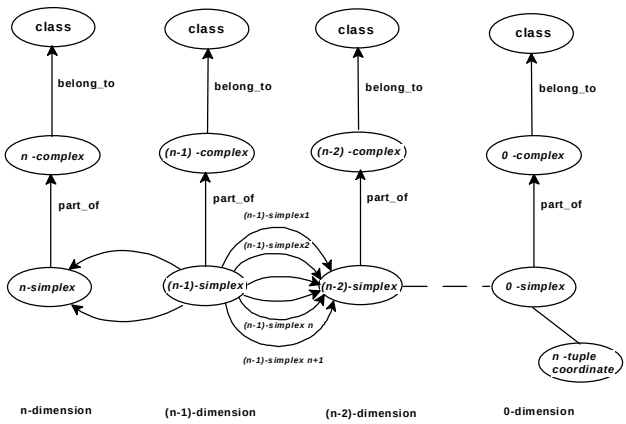

Figure 5-5: TEN: n-dimensional conceptual model

The first problem refers to the modelling stage. The complete tetrahedronization of urban models is theoretically possible and has to be applied in two steps: first, constrained triangulation of all the 2.5D objects (walls, roofs, floors, streets, parking lots, etc) and, second, constrained tetrahedronization of 3D objects (buildings, rooms, etc.). While algorithms for constrained triangulation are widely discussed in the literature, the construction of 3D constrained tetrahedrons is still under research.

The model furnishes the data needed for display of graphic information in the most appropriate way, i.e. triangles. In this respect, TEN is perhaps the optimal model for

visualisation of surfaces and irregular bodies. Maintenance of triangles solves a couple of visualisation problems mentioned regarding 3D FDS, i.e. holes and pitfalls due to explicit storage *arc-on-face* and *node-on-face*. Parsing of the VRML file must be faster due to the provision of only triangles for rendering, i.e. the VR browser does not need to perform a triangulation. Volume and area computations are facilitated as well.

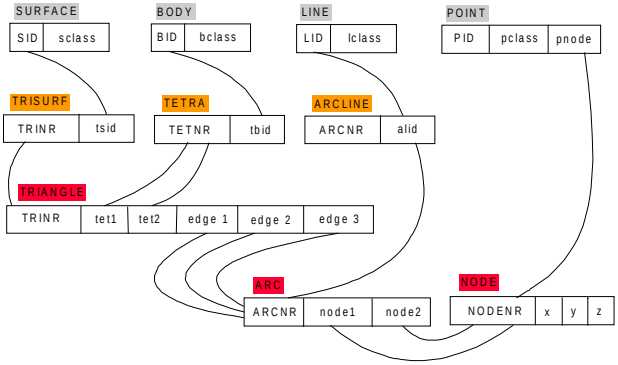

Figure 5-6: TEN: logical model for n=3

Considering the logical model, data to create a VRML document can be extracted in several steps, applying SQL statements and the host language. Suppose during the construction phase, edge1, edge2, and edge3 are oriented in an anti-clockwise direction (for surface triangles of the body), the order of the nodes still has to be derived. Therefore the steps to extract triangles for VRML visualisation might be:

```
SELECT TETNR FROM TETRA WHERE tbid=OBJECT
For each TETNR do {
    SELECT n11, n12 FROM TRIANGLE, ARCNR WHERE TETNR=tet1 and tet2=0 and
    edge1=ARCNR;
    SELECT n21, n22 FROM TRIANGLE, ARCNR WHERE TETNR=tet1 and tet2=0 and
    edge2=ARCNR;
    (Note, that a body has a number of invisible triangles that do not need to be visualised.
    Assuming that the "air" body has ID=0 and tet2 is the right body than the condition tet2=0 will
    extract only the visible triangle. Edge3 is not necessary because all the nodes are extracted.)
    Order the nodes:
        If  (n12=n21) the order is n11, n12 (n21), n22
        If  (n12=n22) the order is n11, n12 (n22), n21
        If  (n11=n21) the order is n12, n11 (n21), n22
        If  (n11=n22) the order is n12, n11 (n22), n21
    For each NODE do {
        SELECT x, y, z FROM NODE WHERE NODE=NODENR;
    }}
```

A number of undesirable side effects concerning the VRML creation may occur. First, the VRML document may become rather long due to more faces in the description section. The VRML node *IndexetFaceSet* may preserve the size of the co-ordinate list (the number of nodes can be the same for 3D FDS and TEN); however, the list of the triangles given in

IndexCoord will at least double. The increase depends on the shape of the real objects and more specifically on the roofs of the buildings. While buildings with complex roof construction (represented by many triangles in 3D FDS) will be affected only with respect to the walls, flat roofs with many corners will create a lot of triangles. This may slow down the delivery of data to the client station.

Second, since the space is completely subdivided into tetrahedrons, the interiors of objects (e.g. buildings), as well as the open space, are also decomposed into tetrahedrons. These tetrahedrons, however, disturb the scene and have to be omitted from the VRML document, which requires additional algorithms to be developed. The covering surface of a body (needed for visualisation) can be either stored in the database as an independent object, or created on the fly by an algorithm selecting the faces of a body, which are not interior. The first approach will lead to database size expansion, the second one will cause longer response time.

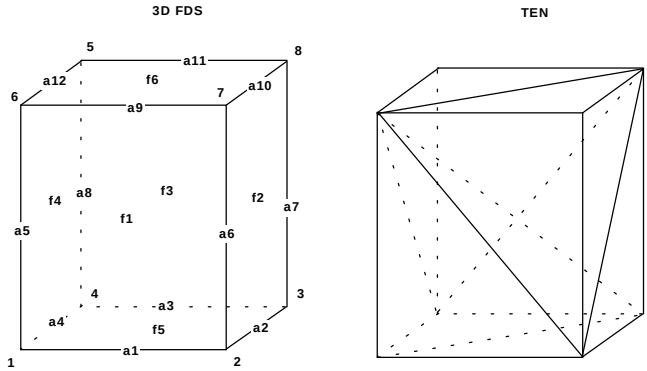

Figure 5-7: A simple box represented in 3D FDS and TEN

However, the most essential concern is the size of the database. The comparison between the logical model of TEN and 3D FDS for a simple box (see Figure 5-7) reveals about 25% increase in the size in a TEN representation. Table 5-4 contains the number of records, the length (in bytes) of a single record and the total size (in bytes) of the box. Analysis of the values, i.e. face+edge vs. triangle, shows that the triangle representation is "cheaper" considering the 2-cells (triangle and face). Although three times more than faces, triangles need less space because the relationship *triangle-arc* is constant (1:3) and explicitly stored in the TRIANGLE table. The EDGE table in 3D FDS has to maintain the *face-arc* relationship, which has cardinality 1:n, as well as the order of the arcs in a face. The order of the arcs in the TRIANGLE table is obtainable from the sequence in which edge1, edge2, edge3 are recorded.

TEN maintains two more tables, i.e. TETRA and TRISURF, which are necessary to compose the needed *complex*. The size (144 bytes for the example) is such to “compensate” 3D FDS for the more expensive *face-arc* relationship. That is to say, FACE+EDGE (3DFDS) needs less space (408 bytes) than TETRA+TRISURF+TRIANGLE (TEN) (576 bytes).

The significant in data is a result mostly of the large number of arcs and triangles obtained from the full triangulation. A subdivision of a face bordered by *n*-arc *(n>3)* leads to *(n-3)* additional faces and *(n-3)* new arc. For example, the ARC table in TEN contains more records than the one in 3D FDS. The increase is at least as much as the number of the rectangular faces plus at least one internal arc. The growth of information is even faster for faces with holes, as the rate depends on the number of holes. The image pieces used for texturing also have to be subdivided and we face again an increase of data: triangular pieces of texture require larger storage space.

Last, is still difficult to create an urban spatial model it in TEN, mostly because of a lack of efficient algorithms for constrained tetrahedronization.

Table 5-4: Size comparison of 3DFDS and TEN

3D FDS – BodyObj				TEN – Body			
	Number	b/r	Bytes		Number	b/r	Bytes
Bodyobj	1	24	24	Body	1	24	24
Surfobj	1	24	24	Surface	1	24	24
-	0	0	0	Tetra	6	8	48
-	0	0	0	Trisurf	12	8	96
				-	0	0	0
Face	6	16	96	Triangle	18	24	432
Edge	24	13	312	-	0	0	0
Arc	12	12	144	Arc	19	12	228
Node	8	16	128	Node	8	16	128
Total	50	57	680	Total	65	116	980

In conclusion, despite the facilitation for rendering, TEN creates a much larger database than that created by 3D FDS, and requires special processing of the tetrahedrons that are not needed for visualisation.

5.3.3 The cell tuple model

The spatial model introduced by Brisson 1990 and extended by Pigot (see Pigot 1992) will be referred to as *the tuple model*. It defines cells and cell complexes upon the fundamental properties of a manifold. A *k*-cell (where *k* is the dimension of the cell) is defined as a bounded subset of a *k*-manifold and hence it is homeomorphic to a *k*-manifold with *(k-1)*-manifold boundary(s). The *k*-cell complex is the union of all the *k*-dimensional and lower cells. Some later extensions (see Mesgari et al 1998) of the model permit the existence of singular n-cells, e.g. 0-cell inside 2-cell, 2-cell inside 2-cell (holes), 3-cell inside 3-cell (tunnels). Under these circumstances, any spatial object is described as a set of tuples of 3-cell, 2-cell, 1-cell and 0-cell, i.e. the representation of cells is implicit. From construction point of view, the model permits cells with an arbitrary shape. For example, the box above does not require decomposition into tetrahedrons. Despite less partitioning, the model does not lead to fewer records. For example, face f1 (see Figure 5-7), part of a body b1, will be represented by 16 records (see Table 5-5) and body b1 is fully described by 96 records. According to the author's estimation, complex real objects may lead to enormously large representations (see Pigot 1995).

Table 5-5: An example of a face representation in the cell model

Records	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0-cell	1	1	1	1	2	2	2	2	7	7	7	7	6	6	6	6
1-cell	1	1	5	5	6	6	1	1	9	9	6	6	5	5	9	9
2-cell	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3-cell	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0

The model promises 1) the capacity to provide a large spectrum of topological relations between cells and complex cells, 2) easy implementation (a table structure), and 3) an easy maintenance, due to the claimed solid mathematical foundations. Reports on investigations of the model for a variety of applications are already available (see Mesgari et al 1998, Raza and Kainz 1998).

Since the construction rules are similar to the rules of 3D FDS, the model can be considered quite suitable for modelling in urban areas. In the visualisation respect, the extraction of faces and points (needed for VRML documents) seems to be a simple operation, due to the explicitly stored link between the cells. The data obtained from the tuple representation, however, lacks any indication regarding the order. Supplementary records are needed to establish the order (clockwise or anti-clockwise) of cells (note the cyclicity is ensured). This will result in further increase of the space for database storage and eventual complication of the algorithms for data extraction. The performance is difficult to evaluate without implementation. Assuming a relational implementation, the entire tuple information is available in one relational table, which has advantages and disadvantages. On one hand, there is no need to perform JOIN operations to select data for VRML document. On the other hand, the size of the table grows tremendously, which slows down the speed of SELECT operations. For example, the records for the box above occupy 1536 bytes storage space, which is double compared with 3D FDS (see Table 5-4).

It is apparent that TEN and 3D FDS permit more compact representations than the tuple model. Note also that an appropriate JOIN operation can create the tuple table from TEN. 3D FDS can be converted into a tuple table as well, if special operators process the explicit relationships *node-on-face*, *node-in-body*, *arc-on-face* and *arc-in-body*. The opposite conversion is possible with some additional operations, i.e. partitioning of the objects according to the construction rules of TEN, and creation of new tables for singular cases in 3D FDS.

Among the three spatial models, 3D FDS and the tuple model reveal advantages for 3D modelling: 1) the shape of the real objects is maximally preserved, 2) complete subdivision of the space is not required and 3) many singularities (e.g. holes, *node-on-face*, *arc-on-face*) are permitted. More elaborated discussion on space subdivisions and singularities is given in Section 5.5.2. A simple comparison of size representation indicates some advantages of 3D FDS. TEN is quite appropriate for visualisation with respect to avoiding visualisation artefacts, but requires large storage space and imposes undesirable partitioning of real objects from urban areas. The tuple model leads to the largest representation among the three models. Bearing in mind the expected amount of data in urban models, its utilisation might result in an unmanageable system.

Clearly, advantages of a model in one of the aspects occur as disadvantages in another aspect (see Table 5-6), which motivates the search for alternatives. Alternatives can be

found in altering both construction rules and objets. Although quite approximate, the estimation of the models reveals benefits in utilising 3D FDS. The weak aspect of 3D FDS is visualisation, which, however, can be improved if more strict construction rules are applied. Let us introduce the rule "all the faces are triangles" and analyse the new model denoted 3D FDS (triangulated). The partition of the objects will be indeed higher, all the surfaces have to be triangulated, however the space subdivision is unchanged. Singularities will be relatively reduced, i.e. the relationships *node-in-body* and *arc-in-body* will remain but *node-on-face* and *arc-on-face* will disappear. These changes will result in a new logical model. The fields of the EDGE table will become a constant number, which will reduce the size of the table significantly. Calculations of the database size indicate that the modified structure of the relational table will compensate for the increased number of triangles in 3D FDS (triangulated). In general, 3D FDS (triangulated) promises greater suitability for our system architecture than 3D FDS.

Table 5-6: A comparison between the three spatial models

Aspects	Criteria	3DFDS	TEN	The tupple Model	3DFDS (triangulated)
Modelling	Partition	1	3	2	2
	Subdivision of space	1	3	1	1
	Singularities	1	3	1	2
Visualisation	Faces and co-ordinates	1	1	1	1
	Orientation of faces	2	1	3	1
	Artefacts	2	1	2	1
Performance	Time for database traversal And VRML creation	3	2	3	1
	Time for delivery	1	3	1	2
	Time for parsing	1	2	1	1
	Database size	1	2	3	1
Total		17	21	18	16

1-good, 2-acceptable, 3-unsatisfactory

5.4 Arcs in spatial models

Spatial models in 2D GISs commonly maintain three constructive primitives, i.e. nodes (0-cell), arcs(1-cell) and faces (2-cell). The topology is defined by explicit or implicit neighbourhood information between all the **CnsO**. Spread over all the objects in the entire map (or layers of different maps), the topology allows spatial analysis of various complexities to be carried out. Usually, the arc is the basic constructive object, mainly because it provides a finite boundary and co-boundary information. The arc has two bounding nodes and two co-bounding faces that strictly define the neighbourhood of the arc. A constructive object with the same properties does not exist in 2D space. The node lacks bounding **CnsO**, and the face does not have a co-bounding **CnsO**. In addition, the relation *node-arc* and *face-arc* is *one-to-many*. Several spatial models have been built utilising the unique properties of the arcs. Early approaches such as the DIME model (see Corbet 1975) store the two nodes bounding an arc and its co-bounding polygons (faces). To speed up the retrieval of faces, later spatial models explicitly store the boundary of the face usually as a list of arcs. Examples are the *arc-node* model described by Aronoff in 1989 (see Aronoff 1995), FDS (see Molenaar 1989), ATKIS (Hesse and Leahy 1990), etc.

Early spatial models in computer graphics systems (CAD) make use of the same three **CnsO**. Since the topology in computer graphics is defined for the surface of a solid object, many of the spatial models are in a sense similar to those in GIS. Despite the different original mathematical reasoning, they can be classified as subdivisions of 2-manifolds (see Chapter 2). A study on existing modelling systems, which operate with solids presented by Baer et al in 1979 has revealed that the arc (edge) is the key constructive element of many systems. The very first structuring for graphic visualisation purposes is the *winged-edge* model proposed by Baumgart in 1974 (see Mäntylä 1988). The model maintains information about the left and right faces, and left, right, counter and clockwise arcs of an arc. Following this model, several graphic systems have been developed, e.g. *Geomed, Geomap* and *Build-2* (see Baer et al 1979). Several systems have a structure similar to the 2D GISs mentioned above, i.e. face is represented as a list of arcs and arc is represented as a list of nodes (e.g. *PADL, BUILD, COMPAC, PADL*). However, two modelling systems (among 11 explored) have been organised on the relation between only faces and nodes (i.e. the faces are represented by lists of nodes), one maintains only faces with neighbouring faces and two have expressed faces as lists of arcs and lists of faces. One of the systems missing arcs is *EUKLID*, developed at LIMSI, Orsay, France in 1976. Each body is represented by a list of faces with information about the number of the nodes in a face and the pointer to the first node in a face. All the descriptions of faces are stored in a file called "line". The nodes are in a separate file "vertices" as current positions of nodes in the file are used as ID to complete the description of faces. The system has been capable of operating with several primitives, e.g. box, wedge and polyhedron, as well lines and points. Although the study is from the time of vector graphics, it is evidence that graphics systems based on faces and nodes has been successfully realised in the past.

Currently, the status of arc in CG is more doubtful than ever. On one hand, the requirements of VR systems for real-time visualisation and effective management of large volumes of spatial data have directed the research in CG toward investigation of hierarchical data representations with minimal **CnsO** for storage (see Campagna et al 1999, Lindstrom et al 1996, Popovich and Hoppe 1997). On the other hand, the standard rendering engines, e.g. OpenGL (see Woo et al 1997) and Direct 3D, which are already widely hardware implemented, contain sets of procedures that require vertices ordered in a special manner. The structure of data in VRML lacks arcs too (see Chapter 4).

The function of arcs in 3D FDS is basically representation of the link between nodes and specification of the order (in contrast to the 2D variant). Further, the *arcs* are building elements for lines and edges, where the order is explicitly specified or can be specified. If arcs do not exist, the line object and edge CnsO can be represented by a sequence of nodes. The replacement of *sequence of arcs* by *sequence of nodes* in the LINE and EDGE tables will not increase the number of records drastically: it remains the same in the EDGE table and increases by one per line in the LINE table. Consequently, the global effect of this modification of the model will be the significant reduction of data. The ARC table is one of the largest tables. The results of experiments with triangulated surfaces shows that the ratio faces:arcs:nodes is 2:3:1. With the elimination of the ARC table, the relationships arc-in-body and arc-on-face are also superfluous, because node-in-body and node-on-face will represent the same spatial relationships.

On the basis of the discussion above, we have created the hypothesis that *arcs* can be safely omitted from the representation of the model. To prove the hypothesis a new spatial model will be defined in the next section. Since the model was inspired during the experiments with 3D FDS, some principles are quite similar. The concepts of the 3D FDS that are preserved in the new model will be explicitly mentioned.

5.5 The Simplified Spatial Model (SSM)

In this section, the definition of a new spatial model will be given. It will be referred to as Simplified Spatial Model because arcs are not used to construct objects. According to the proposed definition of an object, the geometry of each spatial object can be associated with four abstractions of geometric objects, i.e. *point*, *line*, *surface* and *body*. A *point* is a spatial object that does not have shape or size but position is the space. A *line* is a type of a spatial object that has length and position. A *surface* is an abstraction of spatial object that has position and area. A *body* is a type of spatial object that has a position and a volume. All the **GO** are built of smaller, simpler elements, i.e. constructive objects. The model consists of two **CnsO**, i.e. *node* and *face*.

The formal definition of the spatial model establishes the rules according to which an object can be composed, clarifies allowed configurations and specifies the *topological primitives* (*closure*, *boundary*, *interior* and *exterior*) needed for the later elaboration on neighbourhood relations. Furthermore, we assume that all the objects are embedded in *Euclidean 3D-space*, denoted by $\mathbb{R}^n$ where $0 \le n \le 3$. The formalism employs fundamental definitions, theorems and concepts of set theory (see Lipschultz 1964 and Willard 1970) and linear algebra (Anton 1994). The basic category utilised in the definitions is the one of *indexed sets.* The index gives a unique identification of any spatial object, which facilitates many stages of the implementation (see Chapter 7).

5.5.1 Definition of constructive objects

Let U be the universe and p any point in it.

Definition 1: The *node* denoted by N_i is an indexed set of one element p, i.e.:

$N_i = \{p\}$, where i is the *unique index* of a node,

with the following property:

a) Two nodes cannot have the same element, i.e. they are always *disjoint*:
If p is a point from $\mathbb{R}^n$ such $p \in N_i$ and $p \in N_j$, then $N_i = N_j$ and $i = j$

The *interior* of a node, denoted by $N°$, is the empty set. The *boundary* of the node, denoted by ∂N, is the node by itself. The *closure* of a node, denoted by $\overline{N}$, is the union of the boundary and the interior, i.e. $\overline{N} = \partial N \bigcup N°$. The *exterior* of a node, denoted by N^-, is the difference between the universe U and the closure of $\overline{N}$, *i.e.* $N^- = U - \overline{N}$.

A node embedded in $\mathbb{R}^3$ is represented by a point with co-ordinates (x,y,z). It is not necessary for all the points in $\mathbb{R}^3$ to be nodes. One can think about a building where only the footprints have known co-ordinates and thus only they are represented in the model as nodes. The geometric representation of the exterior of a node is everything but the node by

itself.

Next, we will define the family set of all the nodes *ND*, which belong to the same topological space. ND is a subset of all the points in the universe, i.e. $ND \subset U$ **.**

Definition 2: If Λ is the topological space then the set of all the *n* nodes $N_i \in \Lambda$ is denoted by *ND*, i.e.

$$ND = \{N_i\} = \bigcup_{i=1}^{n} N_i$$

with the following properties:

a) The intersection of all the nodes is the empty set, i.e. $\bigcup_{i=1}^{n} \bigcup_{j=i+1}^{n} N_i \cap N_j = \varnothing$, i.e. the subsets N_i are a *partition* of *ND.*

b) Two nodes N_i and N_j in $\mathbb{R}^3$ are *connected* iff there is a straight line linking them, otherwise they are disconnected. The straight line connecting two nodes will be referred to as *link* in the following text.

Definition 3: A *face* denoted by *F* is an indexed set of x ordered nodes $N_i \subset ND$, where $3 \le x \le n$ (if $x = 3$ the *face* is called *triangle*), i.e.

$$F_j = \{N_i^f\} = \bigcup_{f=1}^{x} N_i^f \ ,$$

where *f* is the *face index* of a node specifying the current order in a face F_j,
with the following properties:

(for simplicity, the unique index *i* is omitted)

a) F_j is a *connected set* for each ordered pair $(N^f, N^{f+1}) \in F_j$, $1 \le f \le x$.

b) In the set of nodes cannot exist two equal nodes, i.e. the intersection of the nodes $\{N^f\} = F_j$ is the empty set, i.e. $\bigcup_{f=1}^{x} \bigcup_{ff=f+1}^{x} N^f \cap N^{ff} = \varnothing$.

c) Each triple of nodes N^f, N^g, N^h ,where $N^f \in F_j, N^g \in F_j, N^h \in F_j$ and $f \ne g \ne h$ fulfils only one planar equation $ax + by + cz + d = 0$ in $\mathbb{R}^3$, i.e. the face is planar .

d) There is at least one ordered triple of nodes $(N^f, N^g, N^h) \in F_j$, which does not fulfil the line equation $ax + by + cz = f$ in $\mathbb{R}^3$.

e) All the nodes $\{N^f\} = F_j$, $1 \le f \le x$ are *anti-clockwise oriented* in $\mathbb{R}^3$, e.g.. for each ordered triple $(N^f, N^g, N^h) \in F_j$, $f < g < h$ which have a planar equation $ax + by + cz + d = 0$ and a normal vector *n=(a,b,c),* the constant $c < 0$.

f) F is *convex* in $\mathbb{R}^3$, i.e. if n_i, n_j, n_k are three points such that $n_i \in N^f, n_j \in N^g, n_k \in N^h$ then for each ordered triple $(N^f, N^g, N^h) \in F_j$ the *angle* $\alpha = (u, v)$, $u = \overrightarrow{n_j, n_i}$ and $v = \overrightarrow{n_j, n_k}$ is $0 < \alpha \leq 2\pi$.

g) The set of nodes $\{N_f\} = F_j$ is a subset of the boundary of the face, i.e. $\{N_1, \ldots, N_f\} \subset \partial F$.

The *interior* of *F*, denoted by *F*°, is the area closed by the set of nodes. The *boundary* of *F*, denoted by ∂*F* is the union of all the connected nodes $\{N_i\} = F$, i.e.

$$\partial F = \bigcup_{f=1}^{x} N^f$$

The *closure* of *F*, denoted by $\overline{F}$, is the union of the boundary ∂*F* and the interior *F*°, i.e. $\overline{F} = \partial F \bigcup F^\circ$. The *exterior* of *F*, denoted by F^- is the set difference of the universe *U* and the closure $\overline{F}$, i.e.

$$F^- = U - \overline{F}$$

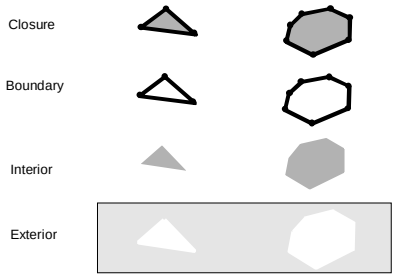

Figure 5-8: Closure, boundary, interior and exterior of a face

Definition 4: The family set of all the *m* faces $F_j \in \Lambda$ is denoted by *FC*, i.e.

$FC = \{F_j\} = \bigcup_{j=1}^{m} F_j$, where *j* is an *unique index* of a face,

with the following property:

a) The intersection of all the faces is the empty set, i.e. $\bigcup_{i=1}^{m} \bigcup_{j=i+1}^{m} F_i \cap F_j = \varnothing$, i.e. the subsets F_j are a *partition* of *FC*. This means that there are not two *equal* faces.

The two constructive objects may interact with each other while representing real objects. To resolve some of the obstacles discussed in the previous section, the interactions between the **CnsO** are specified in the following statements:

a) A node $N_x \in ND$ and a face $F_y \in FC$ are *disjoint*, if the intersection of the node with

the nodes $N^{f} \in F_y$ is the empty set, i.e. $\bigcup_{f=1}^{k} N_x \cap N^{f} = \varnothing$. Otherwise the node N_x is *part of* the face F_y.

b) Two faces are *disjoint* if for each pair of nodes N^{f1}, N^{f2}, where $N^{f1} \in F_{y1}$ and $N^{f2} \in F_{y2}$ their intersection is the empty set, i.e. $\bigcup_{f1=1}^{x1} \bigcup_{f2=1}^{x2} N^{f1} \cap N^{f2} = \varnothing$.

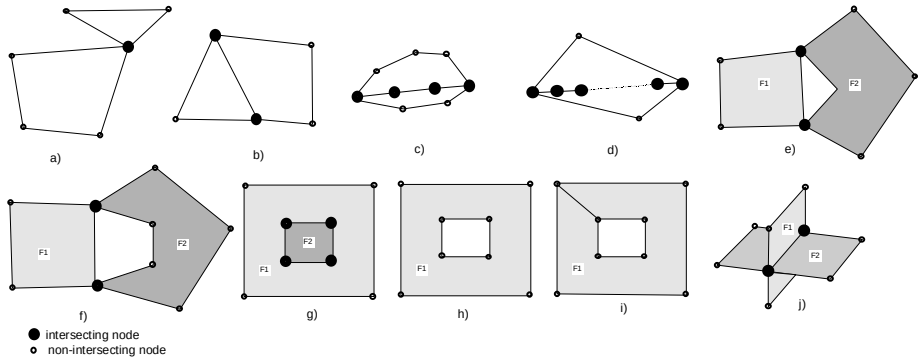

Figure 5-9: Intersections of faces: a)-d) possible; e)-j) impossible

c) If for a tuple of ordered nodes $(N^{f1}, \ldots, N^{f2}) \in F_i$, there are $(F_{y1}, \ldots, F_{y2}) \in FC$, such that the set intersection of all the faces results in the set of nodes, i.e. $(N^{f1}, \ldots, N^{f2}) = \bigcup_{i=y1}^{y} \bigcup_{j=y3}^{y} F_i \cap F_j$, where $y3 = (y2 - y1 + 1)$, $2 \le y \le m-2$ and $2 \le (x2 - x1) \le n-2$, then we say that the faces $F_{y1}, \ldots, F_{y2}$ *strongly meet* at nodes $N_{x1}, \ldots, N_{x2}$ (see Figure 5-9b, c, d). If the set of nodes has only one element N_x, i.e. $x2 - x1 = 0$, then the faces $F_{y1}, \ldots, F_{y2}$ *weakly meet* at node N_x (see Figure 5-9a). If the nodes are only two, i.e. $x2 - x1 = 1$, then they appear as a pair of ordered nodes in each face of intersection.

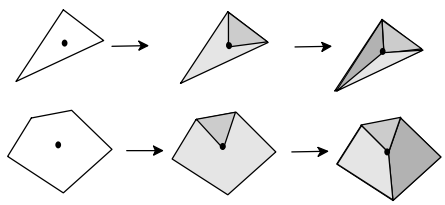

Figure 5-10: Subdivisions of a face with a node

d) $ND - FC \bigcap ND \neq \varnothing$, i.e. there are nodes that are not part of a face. This case occurs when a node is part of a line or a point object or falls inside a body (see Definitions 3 and 4).

The definitions above and their properties specify the shape and the allowed relations between **CnsO** in the model. A face can be constructed of more than three nodes but it must be planar and convex. The interior of *F* does not contain any nodes; a node cannot appear in the interior of a face. A node can be part of the boundary of the face or can be disjoint from the face. This implies that the face has to be subdivided appropriately (see Figure 5-10), when a new node falls inside its interior. The faces provide an orientation given by the order of the nodes. Holes are not allowed. The faces can meet only along the boundaries (i.e. the nodes composing the face) as only three possible configurations are allowed (see Figure 5-9 a,b,c). The Definition 2 (properties g,h) implies that faces that meet at more than two nodes must meet at the straight line connecting these nodes (see Figure 5-9 c,d). Definition 2f implies that two intersecting faces have at least two nodes that are not common (see Figure 5-9 d). For example, the intersections shown in Figure 5-9 (e-f) are not permitted because they contradict to the definitions, as follows:

Configuration e) to Definition 2b, i.e. the link between the nodes is not a straight line

Configuration f) to Definition 3f and 4c, i.e. face f2 is not convex and the intersecting nodes are not in sequential order for f2

Configuration g) to Definition 3g, i.e. there are nodes in the interior of face f1

Configuration h) to Definition 3a, i.e. the boundary of face f1 is disconnected

Configuration i) to Definition 3f, i.e. face f1 in not convex

Configuration j) to Definition 4c, i.e. the order of the intersecting nodes is not sequential for either of the faces.

The definitions of **CnsO** introduced above differ from those given for a face in 3D FDS. The additional restrictions to the shape of the faces (convex and without holes), and the elimination of the relation *node-on-face* by subdividing the face, ensure correct visualisation in any rendering package or VR browser.

5.5.2 Definition of geometric objects

The two constructive objects are now used to compose the four geometric objects, i.e. *point*, *line*, *surface* and *body*. The geometric objects consist of only one type of **CnsO**, as follows: points and lines consist of nodes and surfaces and bodies are built of faces. This section gives formal definitions and specifies the properties of GO. The possible relations between **GO** and **CnsO** and the rules to construct topology, which can be derived from the definitions, are discussed.

Definition 5: A *point* denoted by P_k is an indexed set of *p* nodes N_i, where $p = 1$ i.e.:

$P_k = \{N_i\} = \bigcup_{p=1}^{1} N_i^p$, where *p* is the *point index* of the node.

The topological primitives of a point are:

- The *interior* of *P*, denoted by *P*°, is the interior of the set of nodes, i.e. $P_k{}^\circ = N_i^\circ = \varnothing$

- The *boundary* of the point, denoted by ∂P, is the boundary of the set of nodes, i.e. $\partial P = \bigcup_{i=1}^{1} \partial N_i$
- The *closure* of a node, denoted by $\overline{P}$, is the union of the boundary and the interior, i.e. $\overline{P} = \partial P \bigcup P^{\circ}$.
- The *exterior* of the node, denoted by P^{-}, is the set difference between the universe U and the closure of *P, i.e.* $P^{-} = U - \overline{P}$.

Definition 6: If Λ is the topological space, then the family set of all the *pn* points $\{P_i\} \subset \Lambda$ denoted by *PT* is:

$$PT = \{P_{k}\} = \bigcup_{k=1}^{pn} P_k$$

and has the following properties:

a) The intersection of all the points is not equal to the empty set, i.e. $\bigcup_{i=1}^{pn} \bigcup_{j=i+1}^{pn} P_i \cap P_j \neq \varnothing$, i.e. the subsets P_k are not a *partition* of *PT*. This means that the existence of two *equal* points is possible.

b) Two points P_k and P_l are *disjoint* if the intersection of the nodes composing the points $\{N_i\} \subset P_k$ and $\{N_j\} \subset P_l$ is the empty set, i.e. $N_i \bigcap N_j = \varnothing$.

c) For some $N_i \in ND$ a set of points $\{P_k, \ldots, P_l\}$ can exist such that $N_i \in P_k, .., N_i \in P_l$. This means that the points $P_n, \ldots, P_m$ are *equal*.

d) $ND - PT \bigcap ND \neq 0$, i.e. there are nodes, which does not constitute a point.

e) For some point $P_k \in PT$ a node $N_i \in P_k$ can exist such that $N_i \in F_j$, $F_j \in FC$. This means that the point P_k *meets* the face F_j. Otherwise the point P_k and the face F_j are *disjoint*.

f) a point P_k can be a subset of the boundary of a face F_j, i.e. $P_k \subset \partial F_j$.

Proof. Let there is a node $\{N_i\} \subset P_k$, such that $\{N_i\} \subset F_m$. According to Definition 2, N_i is a subset of the boundary of a face F_m, i.e. $\{N_i\} \subset \partial F_m$. Since $\{N_i\} = \{P_k\}$ (Definition 5), then $\{P_k\} \subset \partial F_m$.

g) $ND - PT \bigcap ND \neq \varnothing$, i.e. there are nodes, which does not constitute a *point*.

According to the definitions above, several points can have a common node and thus they are equal. Points can coincide with the boundary of a face but not with the interior. Similar to the case *node-in-face*, the face must be subdivided. Compare with 3D FDS the difference is the multi-valued concept, i.e. one node can constitute more than one point object.

Definition 7: A line denoted by L_k is an indexed set of x nodes N_l, $2 \le x \le n$, i.e.

$L_k = \{N_i^l\} = \bigcup_{l=1}^{x} N_i^l$, where l is the *line index* of a node,

with the following properties:

a) There is one pair of sets denoted by first node $N^{ft} \in L_k$ and last node $N^{lt} \in L_k$, which are disconnected.

b) The line is *closed* iff $N^{ft} = N^{lt}$.

c) L is a connected set for each $(N^l, N^m) \in L_k$, where $\{N^l, N^m\} \neq \{N^{ft}, N^{lt}\}$.

d) The intersection of each pair of nodes $N^l \in L$, $N^m \in L$ is the empty set, i.e. $\bigcup_{l=1}^{x} \bigcup_{m=l+1}^{x} N^l \cap N^m = \varnothing$.

e) Lines are homeomorphic to a 1-manifold, i.e. they are *simple* lines.

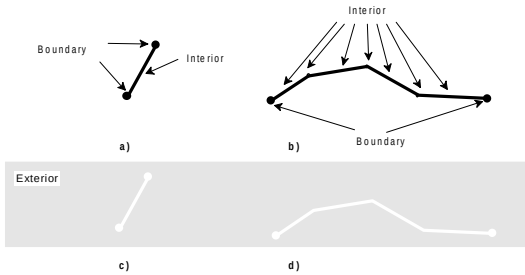

Figure 5-11: Interior, boundary and exterior of a line

f) The topological primitives of a line are:

- The *closure* of L_k, denoted by $\overline{L}_k$, is the set union of closure of all the x nodes $\{N_i^l\} = L_k$, i.e. $\overline{L}_k = \bigcup_{l=1}^{x} \overline{N_i^l}$.
- The *interior* of a line, denoted by $L_k°$, is the set difference of closure $\overline{N}$ of the x nodes constituting L_k and $\overline{N}^{ft}$ and $\overline{N}^{lt}$ i.e.
- $L_k^{\circ} = \bigcup_{l=1}^{x} \overline{N}_i^l - \overline{N}^{ft} \bigcup \overline{N}^{lt}$. If the line has only two points, i.e. $\overline{N}^{ft}$ and $\overline{N}^{lt}$, then the interior is the link between the two *nodes* and, hence, the set interior is the empty set.
- The *boundary* of a line, denoted by ∂L, is the set difference between the closure $\overline{L}_k$ and the interior $L_k°$ i.e. $\partial L_k = \overline{L}_k - L_k^{\circ}$ or $\partial L_k = \overline{N}^{ft} \bigcup \overline{N}^{lt}$.
- The *exterior* of a line, denoted by L_k^-, is the set difference between the universe U and the closure $\overline{L}_k$, i.e. $L_k^- = U - \overline{L}_k$.

A line embedded in $\mathbb{R}^3$ is represented as a finite enumerated sequence of linked points. The first and last points of the line are the boundaries of the line (see Figure 5-11). If the line is composed of only two points then the interior is the link between the two pints (see

Figure 5-11 a). A line must have at least two points that are disconnected.

Definition 8: If Λ is the topological space, then the family set of all the *ln* lines $\{L_k\} \subset \Lambda$ denoted by *LN* is:

$LN = \{L_k\} = \bigcup_{k=1}^{\ln} L_k$,where *k* is the *unique index* of a *line*,

and has the following properties:

a) The intersection of all the *lines* is not equal to the empty set, i.e. $\bigcup_{i=1}^{\ln} \bigcup_{j=i+1}^{\ln} L_i \cap L_j \neq \varnothing$,

i.e. the subsets L_k are not a *partition* of *LN*. This means that the existence of two *equal nodes* in a *line* is possible.

b) The line L_k and the node N_i are *disjoint* iff the intersections of the node with the nodes of the line $\{N^1\} = L_k$ is the empty set , i.e. $\bigcup_{l=1}^{x} N_i \cap N^l = \varnothing$, where $2 \le x \le \ln$,

otherwise the *node* is part *of* the *line*.

c) The line $\{L_k\} \subset LN$ is *disjoint* from the face $\{F_j\} \subset FC$ iff the intersections of each pair of nodes N^l, N^f , where $\{N^l\} \subset L_k$ and $\{N^f\} \subset F_j$ is the empty set , i.e.

$\bigcup_{l=1}^{x1} \bigcup_{f=1}^{x2} N^l \cap N^f = 0$, where $2 \le x1 \le \ln, 3 \le x2 \le m$.

d) The line L_k *strongly meets* the face F_j iff for some line $L_k \in LN$ there is a set of nodes $\{N_i, \ldots, N_y\} \subset ND, \{N_i^l, \ldots, N_y^l\} \subset L_k$ such that $\{N_i^f, \ldots, N_y^f\} \subset F_j$, $\{F_j\} \subset FC$. If the set intersection of *nodes* contains only one element N_x , then the line L_k *weakly meets* the face at N_x .

e) A line L_k cannot be a subset of the interior of a face F_m , i.e. the nodes $\{N_i, \ldots, N_y\} \subset L_k$ constituting the line can be part of the boundary of the face $\{N_i, \ldots, N_y\} \subset F_j$. The property follows from the definition of a line and the property g) in Definition 2 (the *Proof* is similar to Definition 3.e). The face has to be subdivided into smaller faces to incorporate the line (see Figure 5-12).

f) $ND - LN \bigcap ND \neq 0$, i.e. there are nodes, which does not constitute a line.

The line in the model is restricted to laying only on the boundary of a face or to existing freely in the space. If a line has to cross a face, then the face must be subdivided (or triangulated) in such a way as to include the line (see Figure 5-12).

Faces compose the next two geometric objects. Since the faces are set of nodes, some statements will refer to the nodes of the faces. To distinguish between any nodes and nodes *part-of-face*, the corresponding unique and face index of nodes will be utilised.

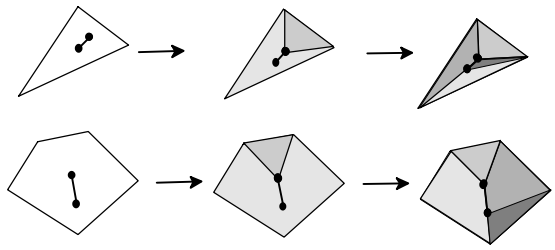

Figure 5-12: Subdivisions of a *face* with a *line*

Definition 9: A *surface* denoted by S_k is an indexed set of x faces F_j, $1 \le x \le m$, i.e. $S_k = \{F_j^s\} = \bigcup_{s=1}^{x} F_j^s$, where s is the *surface index* of a *face*

with the property that the nodes in a surface S_k, i.e. $S_k = \bigcup_{s=1}^{x1} F_j^s = \bigcup_{s=1}^{x1}\bigcup_{f=1}^{x2} N_i^{sf}$ belong to either one or two faces, part of the surface, i.e.

a) If the intersection of a pair of nodes $(N^i, N^{i+1}\} \subset \{F_1, \ldots, F_f\} = S_k$ and all the other pairs of nodes is the empty set, i.e. $\bigcup_{s=1}^{x1}\bigcup_{f=1}^{x2}(N^i, N^{i+1}) \cap (N^{s,f}, N^{s,f+1}) = \varnothing$, where *x1* is the number of faces in a surface and x2 is the number of nodes in a face, then the pair of nodes (N^i, N^{i+1}) belong to only one face. If the set of nodes is equal to the empty set , i.e. $\{N_i, \ldots, N_x\} = \varnothing$.

b) If the intersection of a pair of nodes $(N^i, N^{i+1}\} \subset \{F_1, \ldots, F_f\} = S_k$ and all the other pairs of nodes differ the empty set, i.e. $\bigcup_{s=1}^{x1}\bigcup_{f=1}^{x2}(N^i, N^{i+1}) \cap (N^{s,f}, N^{s,f+1}) \neq \varnothing$, then the pair of nodes (N^i, N^{i+1}) belong to two faces. If all the pairs of nodes belong to two faces then the surface is a *closed* surface.

c) Surfaces are homeomorphic to 2-manifold, i.e. they are *simple* surfaces.

d) The topological primitives of a surface are:

- The *closure* of S_k, denoted by $\overline{S_k}$, is the set closure of all the f faces $\{F_j^s\} = S_k$, T.e. $\overline{S}_k = \bigcup_{s=1}^{x} \overline{F}_j^s$
- The *boundary* of a surface, denoted by ∂S_k, is the union of all the boundaries of nodes $N_i \in S_k$ elements of pairs, which belong to only one face, i.e. $\partial S_k = \bigcup_{i=1}^{p} (\partial N^i, \partial N^{i+1})$, where p is the number of the pairs, which fulfils the condition

$\bigcup_{s=1}^{x1}\bigcup_{f=1}^{x2}(N^{i}, N^{i+1}) \cap (N^{s,f}, N^{s,f+1}) = \varnothing$. The boundary of a closed *surface* is the empty set. The boundary of a *surface* composed of only one face is the boundary of the *face*.

- The *interior* of a surface, denoted by $S_k°$, is the set difference between the closure $\overline{S_k}$ and the boundary ∂S_k i.e. $S_k = \overline{S}_k - \partial S_k$. The interior of a surface composed of one *face* is the interior of the *face*, i.e. the empty set
- The *exterior* of a surface, denoted by S_k^-, is the set difference between the universe U and the closure $\overline{S_k}$, *i.e.* $S_k^- = U - \overline{S}_k$.

Some examples of surfaces and their topological primitives are given in Figure 5-13. The surface can have disconnected boundaries (i.e. holes) but must not intersect itself.

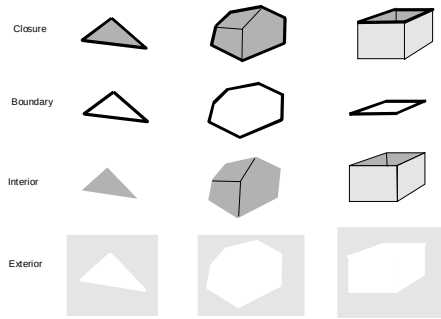

Figure 5-13: Closure, boundary, interior and exterior of a surface

Definition 10: If Λ is the topological space, the family set of all the *sn* surfaces $\{S_i\} \subset \Lambda$ denoted by SF is:

$SF = \{F_k\} = \bigcup_{k=1}^{sn} F_k$, where *k* is the *unique index* of a *surface*.

a) The intersection of all the *surfaces* is not equal to the empty set, i.e. $\bigcup_{i=1}^{sn}\bigcup_{j=i+1}^{sn} S_i \cap S_j \neq \varnothing$, i.e. the subsets S_k are not a *partition* of *SF*. This means that the existence of *equal faces* parts of different *surfaces* is possible.

b) The surface $\{S_k\} \subset SF$ is *disjoint* from the *face* $\{F_j\} \subset FC$ iff the intersections of each face $\{F^s\} = S_k$ and the *face* is the empty set, i.e. $\bigcup_{s=1}^{x} F_j \cap F^s = \varnothing$, where $3 \le x \le sn$. Otherwise the *face* is *part of* the *surface*.

c) $ND - SF \bigcap ND \neq 0$, i.e. there are *nodes*, which does not constitute a *surface*.

d) $FN - SF \bigcap FN \neq 0$, i.e. there are *faces*, which does not constitute a *surface*.

e) A *simple* surface has a connected boundary ∂S_k, which differs from the empty set, i.e.

$\partial S_k \neq \varnothing$.

Definition 11: A body denoted by B_k is an indexed set of x faces F_j, $4 \leq x \leq m$ (if $x=4$ the *body* is called a *tetrahedron*) i.e.

$B_k = \{F_j^b\} = \bigcup_{b=1}^{x} F_j^b$, where b is the *body index* of a *face*,

with the properties:

a) The nodes in a body B_k, i.e. $B_k = \bigcup_{b=1}^{x1} F_j^b = \bigcup_{b=1}^{x1} \bigcup_{f=1}^{x2} N_i^{b,f}$ belong to either three or more *faces*, part of the *body*, i.e.

- For each ordered pair of nodes $(N^i, N^{i+1}) \in B_k$ there are exactly two faces $F_y \in B_k$ and $F_z \in B_k$, such that $(N^i, N^{i+1}) \in F_y$ and $(N^{i+1}, N^i) \in F_z$
- For each node $N^i \in B_k$ there are at least three faces $F_x \in B_k$, $F_y \in B_k$ and $F_z \in B_k$, such that $N^i \in F_x$, $N^i \in F_y$ and $N^i \in F_z$.

b) A body cannot contain other **GO**, but can contain **CnsO**, i.e. *face* and *node*.

c) Bodies are homeomorphic to a 3-manifold, they are *simple* bodies.

d) The topological primitives of a *body* are:

- The *interior* of B_k, denoted by $B_k°$, is the space enclosed by set all the f faces $\{F_j^b\} = B_k$
- The *boundary* of B_k, denoted by ∂B_k, is the closure of all the faces $\{F_j^b\} = B_k$, i.e. $\partial B_k = \bigcup_{b=1}^{x} \overline{F}_j^b$
- The *closure* of a body, denoted by $\overline{B}_k$, is the set union of the interior $B_k°$ and the boundary ∂B_k i.e. $\overline{B}_k = B_k^{\circ} \bigcup \partial B_k$
- the *exterior* of a body, denoted by B_k^{-}, is the set difference between the universe U and the closure of $\overline{B}_k$, *i.e.* $B_k^{-} = U - \overline{B}_k$

A body embedded in $\mathbb{R}^3$ is represented as space bordered by faces. Restrictions on the content of the body interior are imposed concerning only geometric objects but not constructive objects. Nor are requirements for the subdivision of the interior formulated. The motivation for the proposed solution will be given after a short comparison between the three spatial models discussed before.

According to 3D FDS, bodies are constructed on the basis of the existing faces, i.e. each face is part of two bodies. If a face closes part of space in an existing body, then a new body is created. The weakness of the approach is basically in the modelling and maintenance of bodies. For example, a building constructed of two floors (represented as a box with a face in the middle) needs two bodies for the space subdivided by the floors. Thus the building, which most probably has to be maintained as one object, has to be separated into two geometric objects (i.e. two new ID of bodies). A number of operations on geometric and thematic information will be needed to hide the modification from the user.

TEN suggests a more elaborate solution, i.e. each body, regardless of the shape, is subdivided into tetrahedrons. The benefit is that the geometric object remains one object, independent of the geometric changes. The example of a building with floors modelled in TEN will require new tetrahedrons. The ID of the object, however, will be preserved. As discussed before, the disadvantages of TEN are related to the partition of the open space and the database size.

A third approach, i.e. maintenance of singularities, is reported for the cell model (see Mesgari et al 1998). The 3-cell (i.e. body) can contain 3-cell, 2-cell, 1-cell and 0-cell, i.e. no construction rules for the subdivision of the body are imposed. The modelling of the interior of the body is the responsibility of the user. It is well known that singularities permitted in the model allow users to preserve the variety of isolated objects of the real world, e.g. a room in a building, a counter in a building, a desk in a room, a lamp on a desk, a decoration wall in a building. However, the approach reveals disadvantages in several directions: 1) the duplication of data is inevitable, 2) the retrieval and updating require more sophisticated operations and 3) the consistency check is more complicated.

In short, singularities are convenient for the user and the modelling process but complicate the retrieval and control of data, and vice-versa, i.e. the partitioning eases the maintenance of data and imposes constructive rules (mostly undesirable) on the reconstruction. Apparently, some balance between advantages and disadvantages has to be established with respect to the application of the spatial model. The spatial model introduced here aims at visualisation and spatial analysis in urban areas. The definitions presented so far forbid singularities for 1D and 2D objects, i.e. node or point on line, node or point on face, line on face. Thus the first part of the requirements if fulfilled: the models ensure correct visualisation, i.e. provision of data that cannot cause artefacts and rendering problems. Consequently, 0D, 1D and 2D objects have to obey strict constructive rules. The subdividing rules of the body will not further affect the visualisation because the body has an optimal representation for rendering, i.e. a set of faces. Thus the partition of 3D objects is significant for the re-construction and completion of spatial analysis. The discussion above motivates a few singularities, i.e. *face-in-body* and *node-in-body*, to favour the modelling process and spatial analysis. Since the objects inside the body cannot be detected from the definition of body, the relations will be explicitly stored in the model.

Definition 12: If Λ is the topological space the family set of all the *bn* bodies $\{B_k\} \subset \Lambda$ denoted by BD is:

$BD = \{B_k\} = \bigcup_{k=1}^{bn} B_k$,where *k* is unique *body index*

and has the properties:

a) The intersection of all the *bodies* is not equal to the empty set, i.e. $\bigcup_{i=1}^{bn} \bigcup_{j=i+1}^{bn} B_i \cap B_j \neq \varnothing$, i.e. the subsets B_k are not a *partition* of *BD*. This means that the existence of *equal faces* parts of different *bodies* is possible.

b) For some F_j , there is exactly one pair of bodies B_n, B_m such that

$\{F_j\} \subset B_n, \{F_j\} \subset B_m$.

c) The body $\{B_k\} \subset BF$ is *disjoint* from the *face* $\{F_j\} \subset FC$ iff the intersections of each face $\{F^b\} \subset B_k$ and the *face* is the empty set , i.e. $\bigcup_{b=1}^{x} F_j \cap F^b = \varnothing$, where $3 \le x \le bn$. Otherwise the *face* is part *of* the *body*.

d) For some F_j there is exactly one body B_m such that $\{F_j\} \subset B_m^{\circ}$, i.e. the *face* is *inside* the *body*.

e) For some N_i there is exactly one body B_m such that $\{N_i\} \subset B_m^{\circ}$, i.e. the *node* is *inside* the *body*.

f) $ND - BD \bigcap ND \neq 0$, i.e. there are nodes, which do not constitute a body.

g) $FC - BD \bigcap FC \neq 0$, i.e. there are faces, which do not constitute a body.

Definition 13: If Λ is the topological space the family set of all the *fbn* faces inside body $\{F_j\} \subset FC$ denoted by *FIB* and all the *nbn* nodes in body $\{N_i\} \subset ND$ denoted by *NIB* are:

$$FIB = \{F_j^{fb}\} = \bigcup_{fb=1}^{fbn} F_j^{fb}$$,where *fb* is a *face-in- body index* of *node*

$$NIB = \{N_i^{nb}\} = \bigcup_{nb=1}^{nbn} N_i^{nb}$$,where *nb* is *node-in-body index* of *node.*

The 13 definitions given above complete the description of the Simplified Spatial Model (SSM). The model consists of two constructive objects, (*nodes* and *faces*) and four geometric objects (*point, line, surface* and *body*). Nodes constitute points and lines and faces constitute surfaces and bodies. The definitions specify the permitted shape of the constructive and geometric objects, as well as establish rules to compose **GO** from **CnsO**. Each geometric object has specified *topological primitives* as well. The topological primitives will be used to verify the scope of topological relations among the objects in Chapter 6. Visualisation and spatial analysis requirements guide the imposed restriction on the shape and allowed intersections. Since the rendering engines operate with faces and vertexes, most of the restrictions focus on faces and interactions with faces. The consequence of the restrictions is a subdivision of surfaces (and inside lines or points) into oriented, convex, planar, faces. The partition of the 3D space is not as strict as 2D space in the model. Singularities in terms of **CnsO**-in-body are allowed. Due to spatial analysis requirements, i.e. maintenance of relations with bodies, two explicit relations (i.e. *face-in-body* and *node-in-body*) are recorded in the model.

All the statements are presented by notation of set theory under the assumption that the objects are embedded in Euclidean space. The definitions are quite detailed and aim at easy implementation. For example, operators for the reconstruction of the model can be readily derived from the definitions. Apart from some metric computations to ensure a permitted shape, all the operations needed are the basic set theory operations *union, intersection* and

difference.

The core set theory notations utilised in the SSM are indexed sets and the corresponding families of sets. Thus eight families of sets are specified, i.e. *ND* (nodes), *FS* (faces), *PT* (points), *LN* (lines), *SF* (surfaces), *BD* (bodies), *FIB* (face in body) and *NIB* (node in body). The index of the objects agrees with the concept of a unique identification of objects. Being specified at conceptual level, the ID of the objects can be easily transformed from one implementation to another, e.g. the family set will correspond to a class in object-oriented implementation or to a table in relational implementations (see Chapter 7). The supplementary indexes specify the belonging of the **CnsO** objects to **GO** objects. For example, a node $N_i \in ND$, which has *i*-index among all the sets of nodes, can be *part-of* point, line, face, surface and body (as part of face or individually), which is represented by the notations N_i^p, N_i^l, N_i^f, $N_i^{s,f}$, $N_{,i}^{b,f}$, N_i^b, N_i^{nb}.

The Simplified Spatial Model differs from the 3D spatial topological models reported in the literature in the number of constructive objects used, i.e. nodes and faces. The geometric objects (known as complexes, cell complexes, feature objects) are the same. It is similar to 3D FDS and some modifications of the cell model in permitting singularities. The model allows arbitrary shapes (but convex faces) as the cell models and 3D FDS do. Similarly to TEN, triangulation of real surfaces is a basic operation to resolve interactions (point on, line on surface or face) with other objects and complex shapes (holes, concave faces). The complete triangulation of all the surfaces may be considered a modification of TEN. The TRIANGLE and ARCLINE tables will contain only the identifiers of the nodes instead of arcs.

The model is expected to be appropriate for reconstruction, visualisation and query of 3D urban models due to the following considerations:

- faces can have an arbitrary number of nodes, which is frequently observed in urban areas
- bodies can be partitions according to semantic considerations (not constructive)
- restrictions imposed on faces ensure correct display by any rendering engine
- the two constructive objects maintained speed up the traverse of the data and reduce the storage space (will be shown in Chapter 8)
- the scope of topological relations detectable by the model is as large as the one by 3D FDS and the cell model (will be discussed in Chapter 6).

A possible shortcoming of the spatial model concerning retrieval and updating operations (large searching space) might be seen in the multi-valued concept followed, i.e. a face can be part of many surfaces and a node can be part of many faces, lines and points. The R-tree structure built on top of the spatial model, and the codes derived from it, offer a solution for limiting the search (see Chapter 6).

5.6 SSM for urban modelling

Chapter 3 outlined a number of spatial objects (and the corresponding resolution) of common interest for a municipality (see Table 3-4). The construction rules of SSM permit their successful representation.

Buildings might be described either as bodies or composites of surfaces. Several factors

might influence the choice: the significance of the building (public or private), the complexity of construction, the data currently available (e.g. no information about floors). Simplified buildings are appropriate to be modelled as bodies, while complex constructions may require separate surfaces and consequent assembling in a composite object. According to the model definitions, the interior of the building may remain complete, in contrast to TEN and 3D FDS. Examples of both body and surface representations are shown in Chapters 7 and 8. Doors and windows on the facades (if modelled geometrically) have to be first incorporated in the surface of the wall, i.e. windows and the door will be part of the surface wall. The result will be similar to TEN partition but with relaxed triangulation, i.e. windows and doors remain rectangular. The real object underground has to be considered as a complex building and appropriate partition into surfaces has to be applied. Bridges in SSM will be represented as bodies.

Streets, parcels and parks are to be represented as surfaces integrated in DTM. Trees, monuments and man-made holes are most probably to be associated with points. If the height of the tree is needed, it may be modelled as a line. Utilities (water&sewer, electricity and telephone networks) are to be modelled as lines. To illustrate this, several streets, parking lots, gardens, lamps, man-made holes and trees are modelled in the test sites (see Chapters 7 and 8).

5.7 Summary

The chapter deals with the conceptual organisation of spatial urban data. The volume of data necessary for a municipality to govern the town and communicate with a variety of users is enlarged with the data needed to provide Internet access, virtual reality tools for exploration of 3D graphics and realistic visualisation. In order to classify, unify and structure the data for the four groups of objects selected in Chapter 3, an object-oriented framework was presented. According to the framework, the first differentiation between data is on the basis of thematic or geometric origin. Inside each domain (thematic and geometric), four basic separations, i.e. into attributes, relations, behaviour and scenario, are made.

Further elaboration is provided only for the geometric domain. The components attributes (appearance for the geometry), relations and behaviour are specialised and discussed in detail. Thus, the parameters, which describe geometric and radiometric properties of spatial objects are separated into **GDsc** (shape, size and position) and geometric attributes, i.e. **GAtt** (reflectance). The component behaviour is responsible for specific per object movements or activities (e.g. opening, closing,), i.e. temporal dynamics, which do not permanently change objects' shapes and positions. The importance of the component is twofold: 1) to extent the perception and facilitate navigation in the virtual world and 2) to speed up the access and retrieval of data over Internet. The changes of the objects in the time are not discussed. The clear differentiation between geometry and theme helps in both separation and integration of data. The separation is important for the establishment of different hierarchies (in the thematic and geometric domains) and eventual substitution of object representations (e.g. vector with raster). Spatial and non-spatial objects can be integrated together and complex spatio-thematic analysis can be performed.

The core issue discussed in the chapter is the geometric representation of the objects. The objects with spatial extent are organised according to the rules and restrictions of a spatial model. Three existing models, i.e. 3D FDS, TEN and the cell model, are compared and assessed with respect to their suitability to model and visualise urban data. The models were selected under the criterion of being 3D topological models. This is in synchrony with the general approach in this thesis, i.e. a spatial model maintaining 3D topology to be adapted for fast visualisation over Internet. Analysis of the three models revealed advantages and disadvantages, either in modelling urban geometry (e.g. producing a lot of data) or performance (expected long time for data traversal). Of three models, 3D FDS was elected as the one with less disadvantages under the given constraints.

Further analysis of the 3D FDS, based mostly on the relational implementation of the model, as well as a review of models implemented in computer graphics (CAD packages), gave birth to the hypothesis that the spatial model can be built on the basis of only two constructive primitives, i.e. nodes and faces. The apparent advantages are a reduction in storage space and an improvement in the performance. Although derived from 3D FDS, the new model differs quite significantly in a number of rules and restriction, and therefore, a new formulation is given.

Utilising set theory notations, the Simplified Spatial Model (SSM) is defined. The model has two constructive objects (face and node) and four geometric objects (point, line, surface, body). The fundamental notation is indexed set and family of sets. The concept of indexed set corresponds directly to the establishment of unique identification. The definitions of the objects are quite detailed in order to facilitate composition of construction rules (notations for planar and convex faces, face orientation), implementation (supplementary indexes) and validation of 3D topological relations (explicit definition of topological primitives). The model is designed to 1) be appropriate for urban modelling (faces with arbitrary shape and non-partitioned bodies), 2) to provide consistent data for visualisation (convex, planar faces without nodes in the interior), and 3) be able to distinguish between a large number of 3D topological relations. Therefore the construction rules for lines, faces and surfaces (singularities are forbidden) are more restrictive than for bodies (face and node can be inside the interior of a body).

The model is further validated in the following chapters. Chapter 6 validates the capacity of the model to identify a large number of 3D topological relationships. Chapter 7 discusses approaches to collect data and construct the model and presents a mapping into a relational data model. Chapter 8 demonstrates the improved performance (compare with 3D FDS) with a number of tests.