A comparison between ESRI Geodatabase topology and Laser-Scan Radius Topology

Marco Baars
Section GIS technology
TU Delft, Faculty of Civil Engineering and Geosciences
November 2003.

Preface

This is the report written as part of the research done in accordance with the Geo-DBMS Case Study on the TU Delft. This Case study is one of the elective courses in the education of a Geodetic Master of Science. The main goal of this course is to learn about the technical aspects of Geo-DBMSs. This report is written for researchers who are interested in topology issues and for the people who work at Laser-Scan or at ESRI. This report is not an official publication and therefore not meant for quotations.

This research is about a comparison between two methods of treating topology. ESRI released with ArcGIS 8.3 the Geodatabase Topology, which is based on topology rules and Laser-Scan has developed the Radius Topology, where you persist the data into a fixed topological structure. In chapters 3 and 4 you can find the results of this research. For the conclusions and recommendations I refer to chapter 5.

Without the help of a few people, this research could not have been done. First I would like to thank my supervisors Jantien Stoter and Edward Verbree of TU Delft. They helped a lot with their tips and knowledge about the subject. Second I would like to thank the people who helped me from ESRI and Laser-Scan, J. van Winden (ESRI), E. Eijkelenboom (ESRI), E. Hoel (ESRI), D. Warner (Laser-Scan), P. Woodsford (Laser-Scan) and P. Watson (Laser-Scan). With their critical comments, they improved the quality of this report a lot.

Best regards,
Marco Baars.

Summary

In (Zlatanova, 2003), a research has been carried out on the usability of Oracle Spatial within the Rijkswaterstaat (RWS) organization. In the report of this research, Zlatanova recommends to organize the DTB_Nat data into a topological structure. Therefore, a comparison should be made between the ESRI Geodatabase topology, Laser-Scan Radius Topology and Oracle Spatial Topology Manager (release 10). The last one is not available yet, so only a comparison between Radius Topology and ESRI Geodatabase topology could be examined.

In this report a functional analysis between Laser-Scan Radius Topology and ESRI Geodatabase Topology is described. The comparison is about a persisted topological structure (Radius topology) and an approach where the topology is enforced by user-defined rules (Geodatabase Topology). The functional analysis is carried out with some tests to see what the data structures look like and to find out in which way both methods treat objects that do not meet the topological rules.

For these test, the DTB_Nat is used. This is a dataset of Rijkswaterstaat. They use this dataset for water management purposes. This dataset contains a lot of topological errors and on surface level this dataset is a planar graph (according to the product description).

The main differences between Radius Topology and Geodatabase topology are drawn in the next table.

ESRI Geodatabase topology	Laser-Scan Radius Topology
No persisted topological structure	Explicit topological structure
Topology based on rules	Topology defined in a manifold
User decides what to do with errors	User has little influence on what to do with errors
After validation, data is still not in topological structure	After validation, topological structure is correct
Possibility for topology based on semantics	No possibility for topology based on semantics

In the Geodatabase concept, the user decides which topological relationships by topology rules have to be defined and what to do with objects that do not meet these topology rules. In the concept of Radius Topology, the topological structure is fixed. The objects are persisted into a topological structure.

It is important to be aware of some uncertainties:

- If you have defined a topological rule (in the Geodatabase concept) and there are features that do not meet these rules. Then you are not certain if the data is incorrect (a real error) or if the rule is not well defined (not a real error). The user has to be aware of this uncertainty while he is using the Geodatabase concept.
- Whether you persist topological relationships or not, queries still give incorrect results when you have incorrect data compared to the real world. Both the Geodatabase concept and the Radius concept give tools to minimize the amount of incorrect data. The only thing is that in the Geodatabase concept, the data is validated and then put into a topological structure by the user. In a persisted data structure like Radius Topology, the user has less influence, what makes it easier for the DBMS to check the correctness.

Contents

1. Introduction

At TU Delft, a research has been carried out on the usability of Oracle Spatial within the Rijkswaterstaat (RWS) organization (Zlatanova, 2003). Rijkswaterstaat is the organization, responsible for public works and carries out the policy of the Ministry of Transport, Public Works and Water management. That document reports the results of the tests performed with the DTB-Nat, Beheerkaart-Nat and Regiokaart-Nat as they are organized in the currently available Oracle Spatial geometry types. These datasets are produced by Rijkswaterstaat and map all the rivers, lakes and other inland waterways in the Netherlands.

The conclusions of that report, describe that topology is important to maintain a high level of data quality. The datasets contain errors. Many of the errors (invalid geometries, overlapping objects, non-complete coverage) can be easily solved, when using topology. Topologically structured objects have many advantages (according to Zlatanova, 2003):

- It avoids redundant storage (more compact than storing full geometry).
- It is easier to maintain consistency of the data after editing.
- It is the natural data model for certain applications.
- It is efficient for certain visualizations or query operations (e.g. find neighbors).
- It can help detecting topological errors in the dataset.

Therefore, Zlatanova firmly recommends organizing the RWS data in a topological structure. There are some directions given in the report for further research about this topic. One of the directions is to investigate an appropriate topological structure for RWS, e.g. Laser-Scan Radius Topology vs. Oracle Spatial Topology Manager (release 10) vs. ESRI topology.

A comparison between data stored with Laser-Scan Radius Topology and data structured without topology in the geometrical model of Oracle Spatial is made by Louwsma (2003). She did a functional analysis and a performance test using Laser-Scan Radius Topology. Comparing the results of the actual performance test, it seems clear that topological structured data is much slower in finding answers concerning geometry questions. Finding answers on queries, concerning relationships, Radius Topology should be faster, but Louwsma could not examine this, due to missing shapes of the parcels in the original dataset.

This research is part of the education at TU Delft. The first aim of this Case Study is to get competence with Geo Database Management Systems and technical aspects of GIS technology. Another thing that has to be taken into account is that this research is not a Master Thesis, but a Case Study. That means that there's only 25 days to work on this project instead of 25 weeks.

Explicit storage of topology or geometry

When ESRI released its new product ArcGIS 8.3, they also released a new type of data storage. Hoel, Menon and Morehouse (2003), explain this type of data storage in the article "Building a Robust Relational Implementation of Topology". In short, it says that no topology is stored in the database, but only the geometry. Each object is described by a list of coordinates, instead of a list of nodes or edges. If you would like to check your dataset on topological consistency, you define a set of topologic rules, e.g. "parcels must not overlap" or "buildings must not overlap". Afterwards a topology check can be done on the data you're working with (a part of the whole dataset, or the whole dataset). In this way, topological errors can be detected and solved.

At Laser-Scan Radius Topology, a different way of data storage is used. A dataset, stored in this way, is topologically structured. Radius Topology, stores the data into sets of topological primitives (nodes, edges and faces) and relationships between the primitives. The primitives are held in standard tables in the database. Relationships are maintained through join tables, to enable efficient querying of these relationships. The Radius Topology Triggers prevent adding faults when a user modifies data or adds objects in the database.

This research

In this research, a comparison between the datamodel of ESRI and the datamodel of Laser-Scan, will be made. This comparison contains a functional analysis of the topology. A functional analysis will be done on the dataset of RWS, the DTB-nat, which was also used by Zlatanova. This dataset contains a lot of topological errors, so a comparison can be made which datamodel can find these errors and what happens to the topological structure if you want to add polygons to or delete polygons from the dataset.

The main question of this research will be formulated as:
What are the main differences between a datamodel where topological consistency is implemented with a rule-based topological structure (ESRI) and a datamodel that applies an explicit topological structure (Laser-Scan)?

To answer this question, the following "subquestions" have to be answered:

- ***What are the main theoretical differences between Radius Topology and the ESRI datamodel?***
- ***Which topological rules can be formulated to structure the dataset with both datamodels?***
- ***In which way, ArcGIS 8.3 deals with topological consistency?***
- ***In which way, Laser-Scan Radius Topology deals with topological consistency?***

A test has to be done, to answer these questions. First of all, a set of rules has to be defined, to implement in ArcGIS 8.3. The second step is to validate the dataset with these topological rules and to see which tools ArcGIS gives to solve errors that arise after the validation of the topology.

Radius works in a complete different way. First of all, you have to define a topological structure for your dataset. You can choose between a planar graph or network topology as possible topological data structures. The next step is to put the dataset into the formulated structure and see what happens to topological errors in the dataset.

The following points will not be taken into account in this research:

- This research is not a performance test. It's a functional analysis.
- Editing the same dataset with more than one user can cause topological errors. In this research, multi-user aspects will not be taken into account.
- In the recommendations of Zlatanova (2003), also the topological model of Oracle Spatial Topology Manager (release 10) is mentioned. But because of the short time given for this research and because Oracle 10 is not released yet, this topological model will be left outside this research

The dataset used for research

The data, used for this research, is a part of the DTB-Nat. Rijkswaterstaat uses this dataset for watermanagement purposes. In figure 1.1, the complete dataset is drawn. According to (Zlatanova 2003), several types of problems can be identified with polygons in DTB-Nat:

- invalid attribute names

- invalid geometries (duplicate points and self intersection)
- incorrect geometries (missing or superfluous “holes”)
- Inaccurate geometries (small corners or “slivers”overlapping with other objects)
- Incomplete coverage (gaps)

Because of these errors, the dataset is very usable for this research. With topology rules, it has to

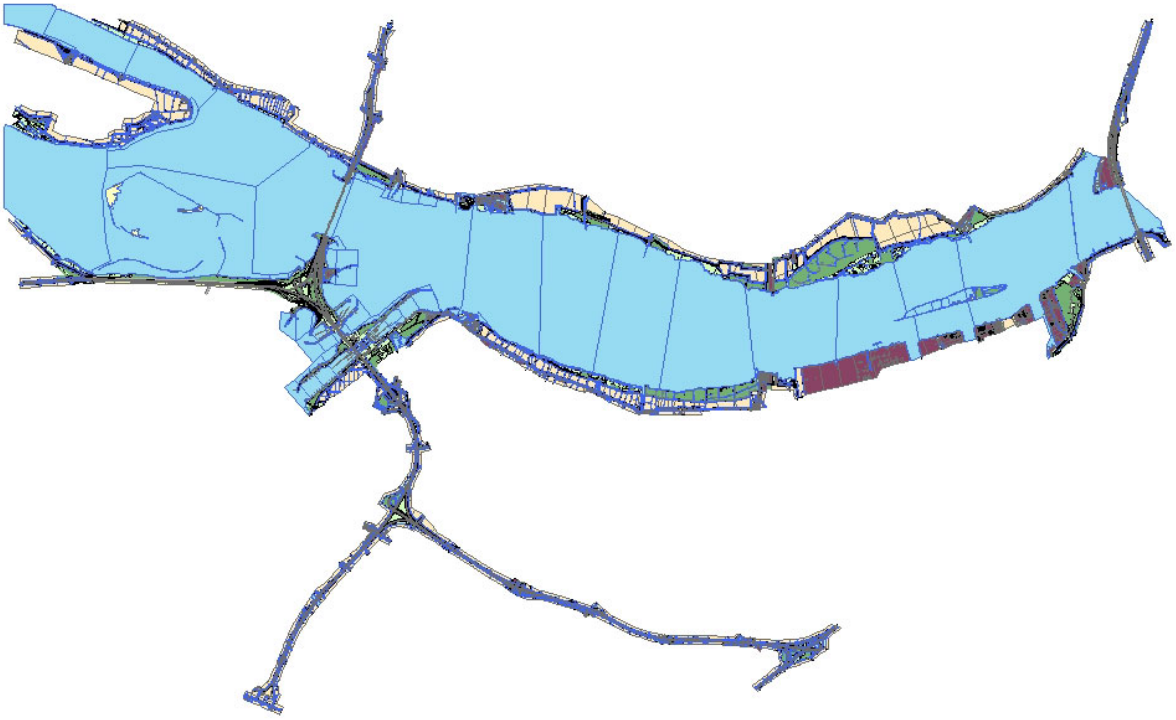


Figure 1.1. The complete dataset used in this research

be able to detect these errors. Because this dataset is large, a few polygons from this dataset are used to describe what the data structures of the Geodatabase and Radius Topology look like. These used polygons are drawn in figure 1.2.

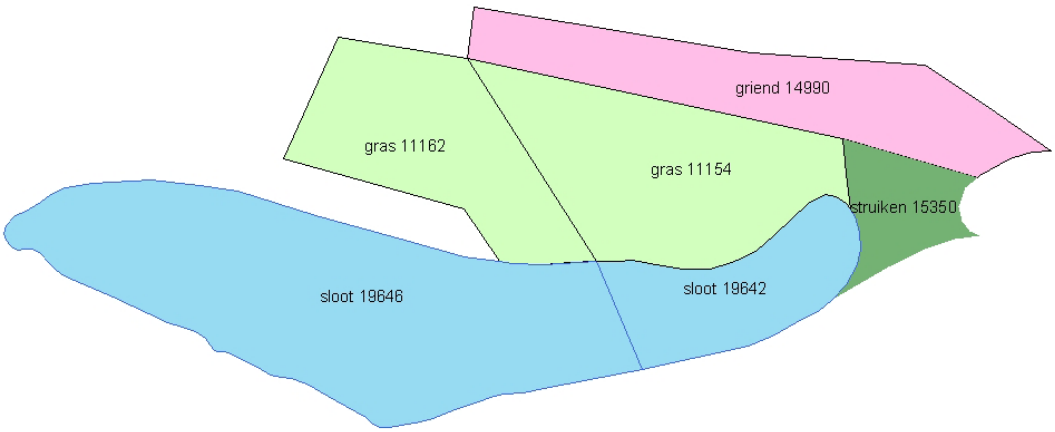
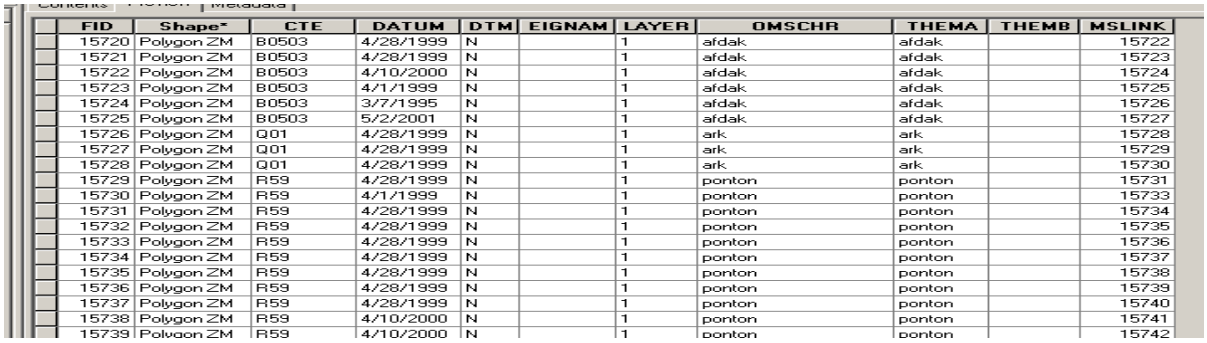


Figure 1.2 The few polygons used for a description of the data structure.

The shapefiles of the used dataset are not really organized. For instance, all the polygons were organized in the same shapefile. A distinction was made between the different object-types as an attribute value. But also these attribute values were not very well-ordered. For instance, there was no object-class like “buildings”. Classes like “house”, “industrial builduing”, “education building”, “houses”, etcetera are distinghuised. A small part of the polygon-shapefile is drawn in figure 1.3. Also polygons from different layers (levels with respect to the surface level) are in the same shapefile. The dataset contains more or less 20000 point objects, 40000 line objects and 20000 polygons.



FID	Shape*	CTE	DATUM	DTM	EIGNAM	LAYER	OMSCHR	THEMA	THEMB	MSLINK
15720	Polygon ZM	B0503	4/28/1999	N		1	afdak	afdak		15722
15721	Polygon ZM	B0503	4/28/1999	N		1	afdak	afdak		15723
15722	Polygon ZM	B0503	4/10/2000	N		1	afdak	afdak		15724
15723	Polygon ZM	B0503	4/1/1999	N		1	afdak	afdak		15725
15724	Polygon ZM	B0503	3/7/1995	N		1	afdak	afdak		15726
15725	Polygon ZM	B0503	5/2/2001	N		1	afdak	afdak		15727
15726	Polygon ZM	Q01	4/28/1999	N		1	ark	ark		15728
15727	Polygon ZM	Q01	4/28/1999	N		1	ark	ark		15729
15728	Polygon ZM	Q01	4/28/1999	N		1	ark	ark		15730
15729	Polygon ZM	R59	4/28/1999	N		1	ponton	ponton		15731
15730	Polygon ZM	R59	4/1/1999	N		1	ponton	ponton		15733
15731	Polygon ZM	R59	4/28/1999	N		1	ponton	ponton		15734
15732	Polygon ZM	R59	4/28/1999	N		1	ponton	ponton		15735
15733	Polygon ZM	R59	4/28/1999	N		1	ponton	ponton		15736
15734	Polygon ZM	R59	4/28/1999	N		1	ponton	ponton		15737
15735	Polygon ZM	R59	4/28/1999	N		1	ponton	ponton		15738
15736	Polygon ZM	R59	4/28/1999	N		1	ponton	ponton		15739
15737	Polygon ZM	R59	4/28/1999	N		1	ponton	ponton		15740
15738	Polygon ZM	R59	4/10/2000	N		1	ponton	ponton		15741
15739	Polygon ZM	R59	4/10/2000	N		1	ponton	ponton		15742

Figure 1.3 Structure of the region table

Structure of this report

After this introduction, in chapter 2, the theory of topology and both Laser-Scan Radius Topology and ESRI’s new datamodel will be described. In the third and fourth chapter, the results of the tests with ArcGIS (chapter 3) and Laser-Scan (chapter 4) will be described. In the last chapter, conclusions will be made and some recommendations will be given.

2. Topology, the theory

In this chapter, the theory of topology will be discussed. First, the common theory of topology will be discussed. After this introduction in topology, the theory of ESRI Geodatabase topology will be discussed. Also some advantages of this method will be mentioned. In 2.3 the philosophy of Laser-Scan Radius Topology will be explained. Also some advantages about this method will be mentioned.

2.1 Topology, the concepts

The use of a topological representation enables the spatial relationships between features to be modeled in terms of their adjacency, connectivity and containment. Topology is sometimes called rubber sheet geometry, because "topologists" view geometric figures as though they were drawn on infinitely-stretchable rubber sheets. For example, imagine a jigsaw puzzle drawn on a rubber sheet that is then pulled down from its center into a cone-like shape. The shapes of the jigsaw pieces distort, but the relationship of each piece to its neighbors does not change. There is still the same number of jigsaw pieces, they are in the same order as before, and lines on the pieces still join where they cross from one piece to the other. (Radius Topology Concepts, 2003)

According to (Clementini et al, 1993), topological relationships are defined as follows: "*the subset of spatial relationships characterized by the property of being preserved under topological transformations, such as translation, rotation and scaling.* Obviously, spatial queries can be easily processed if all the geometric relationships between the objects of interest are explicitly stored; however, such a choice is unsatisfactory since it requires a tremendous amount of disk space and, furthermore, it implies the execution of time-consuming maintenance procedures. It follows that instead of storing all spatial relationships among the objects of interest it is more convenient to compute them."

The 9-intersection model (Egenhofer et al, 1990), is one of the models, in which the topological relationships are defined. According to this model, an object consists of the interior (A^o), the exterior (A^-) and the boundary (∂A). These parts can intersect, in a way that is described in the 9-intersection model. An example is given below in figure 2.1. In this example the operation "overlap" is described for two areas.

$$R(A,B) = \begin{pmatrix} A^\circ \cap B^\circ & A^\circ \cap \partial B & A^\circ \cap B^- \\ \partial A \cap B^\circ & \partial A \cap \partial B & \partial A \cap B^- \\ A^- \cap B^\circ & A^- \cap \partial B & A^- \cap B^- \end{pmatrix}$$

$$R(A,B) = \begin{pmatrix} A^\circ \cap B^\circ & A^\circ \cap \partial B & A^\circ \cap B^- \\ \partial A \cap B^\circ & \partial A \cap \partial B & \partial A \cap B^- \\ A^- \cap B^\circ & A^- \cap \partial B & A^- \cap B^- \end{pmatrix} = \begin{pmatrix} \neg\varnothing & \neg\varnothing & \neg\varnothing \\ \neg\varnothing & \neg\varnothing & \neg\varnothing \\ \neg\varnothing & \neg\varnothing & \neg\varnothing \end{pmatrix}$$

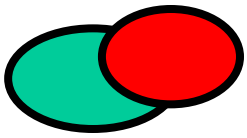


Figure 2.1. The overlap relation in a 9-intersection model

Clementini defined five logical relationships between objects (in 0D, 1D and 2D). The touch, in, cross, overlap and the disjoin relation. He proves in these relationships are mutually exclusive. Using a decision tree, the correct relation can be found. This decision tree is drawn in figure 2.2.

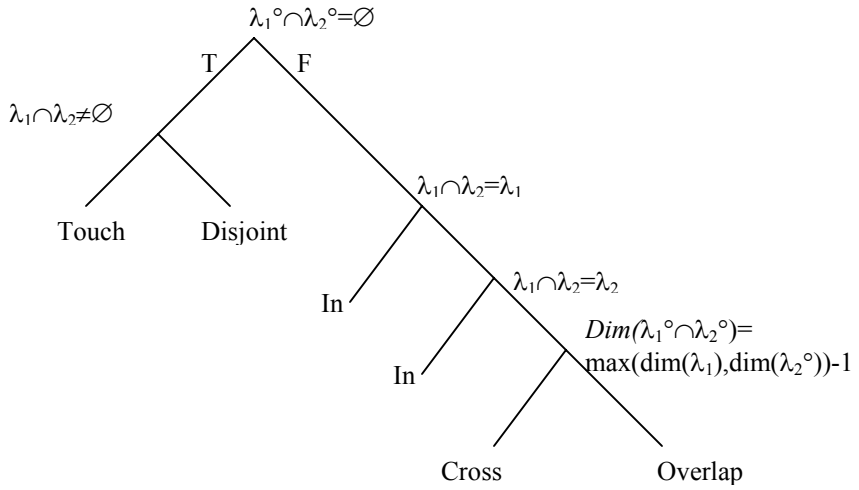
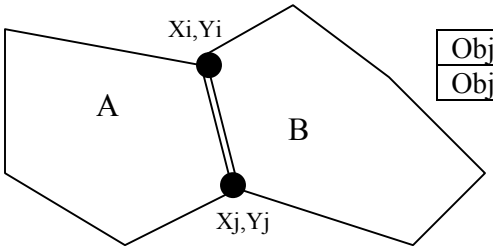


Figure 2.2 Decision tree 4-intersection (Clementini et al, 1993)

In this report, the results of the research are described, that carries out the differences between the way ESRI uses topological relationships and the way Laser-Scan uses it.

In the concept of the ESRI Geodatabase (not to be confused with a Geo Database Management System, a GeoDBMS), the user defines topological relations via a set of rules. This means that the user decides which relationships are important and in which way these rules have to be treated. "The topology is specified by zero or more topology rules. The topological primitives are not persisted as a special type of feature; instead, feature geometry is persisted (figure 2.3) and topological primitives (along with their geometry) are inferred. The process of topological integration (validation) results in vertex equality where features share underlying topological primitives. Given vertex equality, reconstruction of topological primitives is straightforward. Vertices on feature geometries in this scheme play the same role as assigned to embedded foreign keys in data structures that explicitly model topological primitives." (Hoel, Menon, Morehouse, 2003)

According to Laser-Scan (ref: e-mail Peter Woodsford e.a.), reconstruction of topology primitives is neither logically straightforward nor is it a negligible computational expense. They state that this remark demonstrates a lack of understanding and is thoroughly misleading. They also say that it is ridiculous to suggest that vertices play the same role as foreign keys in an "explicit model". Foreign keys refer to topology primitive objects, which refer in general to geometries containing many vertices, not one.

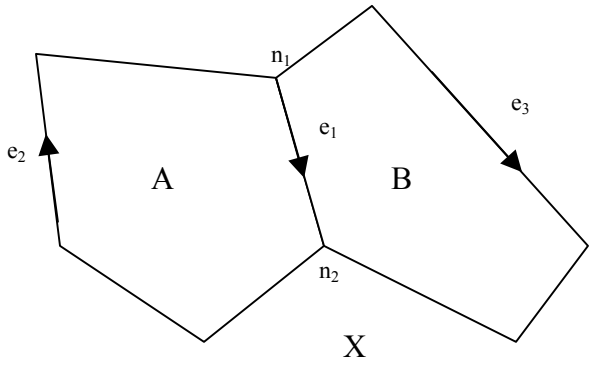


Object A	$Xa_1, Ya_1, \ldots, Xi, Yi, Xj, Yj, \ldots, Xa_n, Ya_n$
Object B	$Xb_1, Yb_1, \ldots Xi, Yi, Xj, Yj, \ldots, Xb_n, Yb_n$

Figure 2.3. A datamodel of only persisted feature geometry. With on the left the map and on the right the feature table.

The concept of Laser-Scan Radius Topology is different than the Geodatabase topology. Not only the feature geometry is persisted, also the topological structure is persisted. Radius Topology provides the user with the option to store both the geometries and the topological primitives or to

only store the topological primitives and derive the geometries. The user defines the topological structure and after that, the DBMS calculates the topological primitives and the topological relationships. In Radius Topology, the topology of a map can be represented as a linear (one-dimensional) network topology or as a two-dimensional planar topology. A network topology uses node and edge primitives to describe interconnected linear features and points on a map. Planar topology uses the face primitive in addition to nodes and edges to describe two-dimensional areas on a map. An area consists of one or more faces. An example of a planar map is drawn in figure 2.4. The geometry of the nodes is described in the table with nodes. The topological structure of Radius Topology is conceptually identical as in figure 2.4. In paragraph 2.3 and chapter 4, that topological structure is described.



Edge	Begin Node	End Node	Left Area	Right Area
e_1	n_1	n_2	B	A
e_2	n_2	n_1	X	A
e_3	n_1	n_2	X	B

Figure 2.4 An example of a planar graph with a persisted topological structure.

2.2 The theory – ESRI Geodatabase Topology

At the introduction of ArcGIS 8.3, ESRI released a new type of data organization. In this datamodel (the geodatabase), no topological structure is maintained. The data is stored as objects, with geometry, represented as a list of coordinates. The topology has to be created with a set of topology rules, for instance “Parcels must not overlap” or “buildings must not overlap with water”. If you have created these topological rules, you can do a topology check on a dataset. Topological errors will be put in an “error-table” and suggestions will be given to solve these errors. In the next figure 2.5 (Hoel, Menon, Morehouse, 2003), these elements of the Geodatabase concept are drawn.

As said before, topological integrity is defined with respect to a collection of topology rules. Topology rules are used to define constraints on the permissible topological relationships between features in one or more feature classes that participate in the topology. The collection of topology rules that are associated with the topology, are selected on the basis of which topological relationships are important for the user’s model. There is a set of 25 topology rules (Appendix B)

that are associated with all topologies; instead, topologies may be specified with zero or more rules.

ESRI (ref: e-mail E. Hoel) added also a number of implicit topology rules that are not mentioned yet:
1. A line or polygon object must be larger than the cluster tolerance.
2. Edges may not intersect.
3. Edge endpoints may not be within the cluster tolerance unless the endpoints are coincident.
4. No edge endpoint may be within the cluster tolerance of another edge.

He states that rules 2, 3 and 4 are referred to as Milenkovic's third normalization rule. These four rules are always enforced on a topology, irrespective of whether or not any other explicit topology rules are defined on the topology.

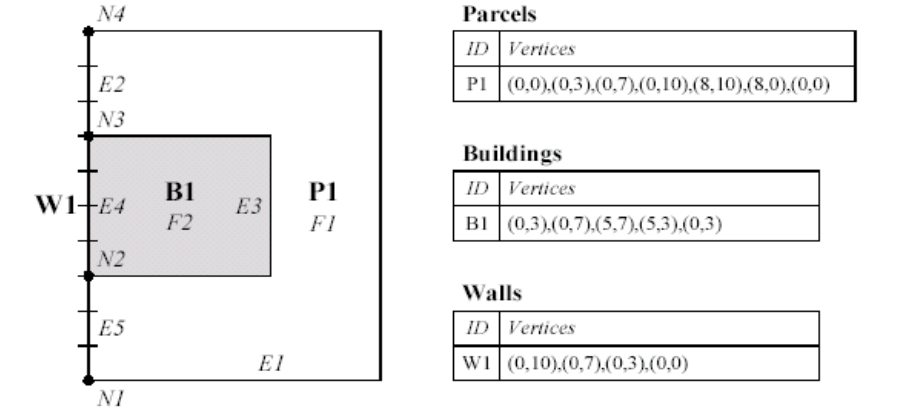


Figure 2.5. ESRI Geodatabase structure (Hoel, Menon, Morehouse, 2003)

The validation process is a fundamental operation on a topology performed by a topology engine. The validation process is responsible for checking all specified topology rules and generating topology errors at locations where rules are violated. The validation process does not need to span all features within the topology dataset. A validation can be performed on a subset of the space spanned by the dataset.

A topology can have an associated dirty area – a dirty area corresponds to the regions within the topology extent where features participating in the topology have been modified (added, deleted or updated) but have yet to be validated. When the geometry of a feature that participates in a topology is modified, the extent of the dirty area is enlarged to encompass the extent of the bounding rectangle of the modified geometry. In order to ensure that the topology is correct, the topology in the dirty areas will need to be validated.

Some *advantages of this Geodatabase model* are:

- Query performance. With a fixed topology model, information of several tables has to be combined when you do a simple geometric query on the dataset. With the geodatabase model, only one, or maybe two tables are necessary to answer simple geometric queries.
- The Integrity Mechanism. It is possible to create rules, based on the semantics of the data. For instance, "water must not overlap with buildings" is a topological rule, based on the semantics. With these rules you create certain integrity in the data.
- Complexity of Updates. Simple edits to a topological data model can result in a complex sequence of changes to the associated topology primitives and relationships. Adding,

moving or deleting an object from the geodatabase, is according to this model, much easier, because you only edit one record of one table.

Laser-Scan (ref: e-mail Woodsford e.a.), has its remarks on these advantages:

- Query Performance: Persistent topology will indeed have more tables but the information content of the tables is much simpler and much better suited to native, optimized query in a RDBMS. Standard relational joins can be used to answer complex spatial queries, avoiding unnecessary computational geometry calculations, each and every time a query is executed. This has a profound impact on scalability of large spatial database systems. This cannot be said to be an advantage for Geodatabase Topology. Rather, it is a serious disadvantage.
- The integrity mechanism Radius Topology provides as different but comparable set of rules for prescribing feature level semantics (classes, tolerances, priorities). These were not adequately explored in this study. In its role as a component, supporting many applications rather than just one, Radius Topology does not currently enforce higher level application semantics but does provide a rich toolkit through the topology domain index an query operators to simply support such feature modeling in a application (e.g. SHARE_FACE). It is inaccurate to portray this as an essential advantage of Geodatabase Topology or a limitation of Radius Topology.
- Complexity of updates: These are entirely irrelevant implementation details, which are hidden from the user. While they may decrease the complexity of the topology engine internally, how does this offer any real advantage tot he user? If only one row is genuinely changed than the topology model is corrupted and must be tidied op later. This simply breaks the data and postpones and hides the necessary processing until later when the net database task (block I/O) is more not less expensive.

2.3 The theory – Laser-Scan Radius Topology

Radius[1] Topology is a dynamic topology management layer stored in Oracle9*i*, creating a server-side topological solution for handling the structure of spatial information. Radius Topology acts as a "gatekeeper": only topologically correct, accurate data is permitted to enter the database. Inconsistent data is automatically corrected, within the bounds of user-defined tolerances and preferences.

ESRI (ref: e-mail E.Hoel) says that this statement also holds for Geodatabase Topology. During the topological integration phase of validation, the Milenkovic conditions are enforced. Overlaps and gaps that are smaller than the cluster tolerance will be removed when the vertices are clustered during topological integration.

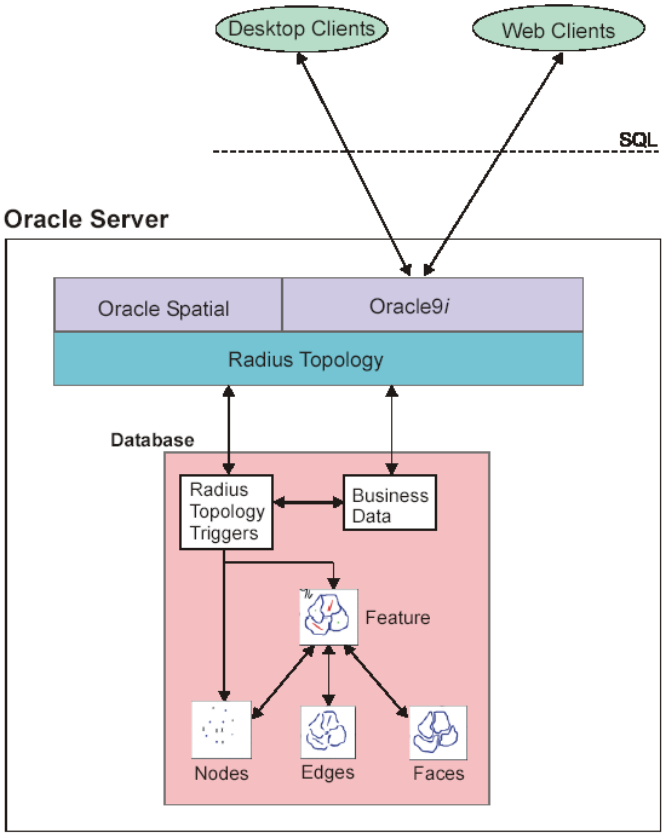


Figure 2.6. Radius as a dynamic topology layer

Figure 2.6[2] shows how Radius Topology can be deployed with clients that update and query an Oracle Spatial database. On the server, a set of Radius Topology database triggers cause the

[1] From http://www.radius.laser-scan.com/about/index.htm (1-jul-03)
[2] From Radius Topology Users Guide

Radius Topology Manager to construct the required topology model when data is input from a client. Feature geometries are broken down into sets of topological primitives (nodes, edges and faces) held in standard tables in the database. Relationships between the primitives are maintained trough join tables, so enabling efficient querying of these relationships.

Radius Topology supports one-dimensional topological mapping, which is described by a network topology and a two-dimensional mapping, planar topology. Often applications want data to be separated into coverages, e.g. administrative areas, vegetation type or land use. These separate topologies are termed ***manifolds***. Manifolds allow the application to build and manipulate independent sets of topology in the same schema. If data is held in different spatial reference systems, then a manifold is required for each, since all data within a manifold must use the same spatial reference system. If you want to use different topology models, you have to use more than one manifold. A ***feature class*** enables topological relationships to be defined between separate groups of features. In figure 2.7 a simple topology configuration is drawn.

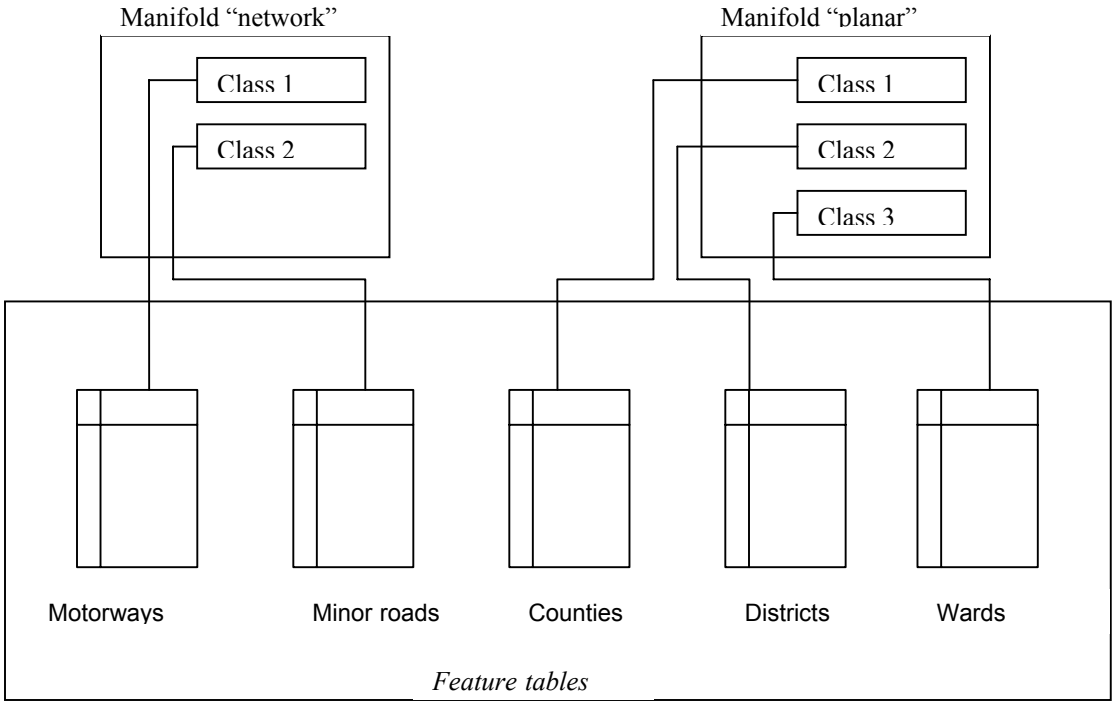


Figure 2.7 Feature class organization in Radius Topology

Topological primitives are the first part of the topological structure within the dataset. The other part is topology rules. Within a manifold, data is structured by applying topological rules that determine the relationships between feature classes. ***Node Formation Constraints*** determine how nodes are created in the topology network to represent the feature objects. ***Topological priorities*** determine which object will move when snapping occurs between objects.

There are three types of node formation constraint. These are:

- ***Share-Node***: When a Share Node tolerance is defined, new objects will share existing object nodes if they are within this defined tolerance
- ***Node-Split-Edge***: If a Node-Split-Edge tolerance is defined, the node of a new object can split the edge of an existing object if it is within this tolerance, causing the objects to

share a node. Likewise, an existing node can split the edge of a new object if within tolerance.

- ***Edge-Split-Edge***: If an Edge-Split-Edge tolerance is defined, the edge of a new object can split the edge of an existing object if it intersects or comes within this tolerance, creating a new shared node.

Important about Radius Laser-Scan Topology is that an explicit topological structure is used.

Some *advantages about Radius Laser-Scan Topology* are:

- It avoids redundant storage (an edge is stored only once in the database).
- It's easier to maintain consistency of the data after editing (because of the fixed topology)
- It is efficient for certain query operations (topological queries like "what are the neighboring parcels of parcel X".

According to ESRI (ref e-mail E. Hoel), it is important to not that although an edge may only be stored once within Radius Topology, the original feature geometry is also stored (see Section 4 of the Laser-Scan DBA Guide). The GDB Topology model actually requires much less storage given that they both store the feature geometry, but only Radius stores the topological primitives and relationships. In comparison, the storage required for dirty areas in the GDB Topology is significantly smaller. The test as written gives the impression that representing features in a topology using Radius requires less storage than is necessary with GDB Topology.

3. ESRI Geodatabase, the results

In this chapter, the results will be discussed of the tests done with ArcGIS 8.3. With the first test, described in 3.1, a description of the datamodel will be explained.

3.1 The Geodatabase data structure

In this paragraph, a description of the datamodel will be given of the datamodel ESRI is using in his new Geodatabase concept. This description will be given with respect to the example given in the introduction. The six polygons are drawn again below in figure 3.1. The attribute table that belongs to these objects is drawn in figure 3.2. In the attribute table, no coordinates are written.

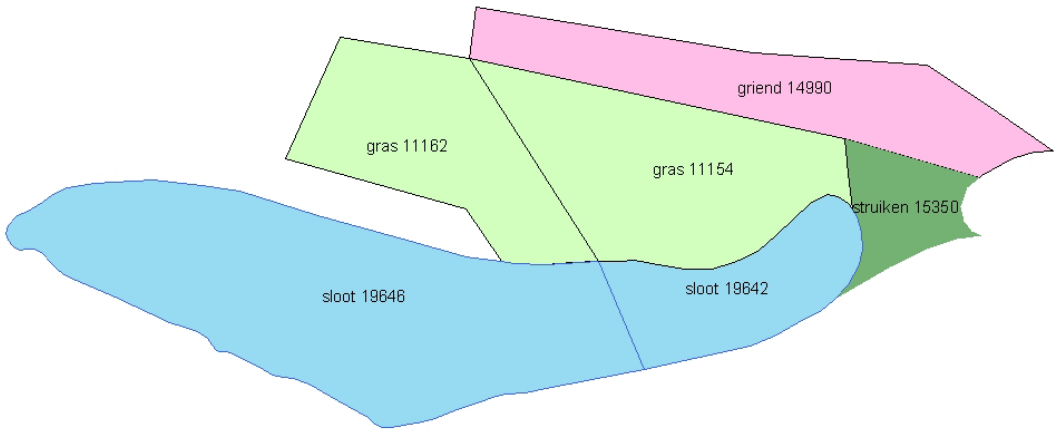


Figure 3.1 The few polygons used for a description of the data structure.

Attributes of stukje

FID	Shape*	OBJECTID	CTE	DATUM	DTM	EIGNAM	LAYER	OMSCHR	THEMA	THEMB	MSLINK	SHAPE_Leng	SHAPE_Area
0	Polygon	11154	N050101	4/28/1999	J		1	gras	gras		11155	170.266773	1350.544398
1	Polygon	11162	N050101	4/28/1999	J		1	gras	gras		11163	148.318668	1058.289632
2	Polygon	14990	N0503032	4/28/1999	J		1	griend	griend		14991	220.083314	1148.544868
3	Polygon	15350	N0503036	4/28/1999	J		1	struiken	strkn		15351	95.155548	350.557091
4	Polygon	19642	W030703	4/28/1999	J		1	sloot	sloot		19645	123.281444	699.694073
5	Polygon	19646	W030703	4/28/1999	J		1	sloot	sloot		19649	251.383321	2584.396843

Figure 3.2. The attribute table of the polygons

A description of the Geodatabase datamodel is given in the ArcGIS Desktop Help:

> *The geodatabase supports a model of topologically integrated feature classes, similar to the coverage model. However, it extends the coverage model with support for complex networks, topologies, relationships among feature classes, and other object-oriented features.*

In a Geodatabase, feature classes are organized in feature datasets. A feature dataset is a dataset that share the same spatial reference. It is strange that the "Help" says that the geodatabase concept extends the coverage model. Coverages are used to store feature data in a fixed

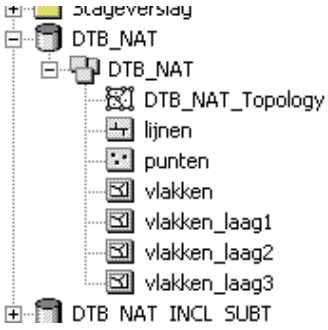


Figure 3.3. The geodatabase structure

topological representation. In the Geodatabase concept, the topological relationships between objects are not persisted. The user can add topological relationships to the feature dataset. The former ESRI model with coverages and persisted topological relationships is still supported in ArcGIS 8.3 environment, according to the Desktop Help. The Geodatabase also provides support for other data types (e.g. TINs, GRIDs) in a RDBMS as well as additional capabilities for representing complex data models, the ability to support an optimistic long transaction model and some form of distributed data management.

Adding topological relationships is easy. With a wizard, you can easily implement rules. These rules are stored in the database as a table with rules. During the validation-step, the objects in the geodatabase will be compared to these rules. If one or more objects do not meet the defined rules, in a separate table, the errors are listed. As a test, topological rules are defined to check if the six polygons from figure 3.1 meet the constraints for a planar graph. The defined rules are:

"Polygons must not overlap" and
"Polygons must not have gaps"

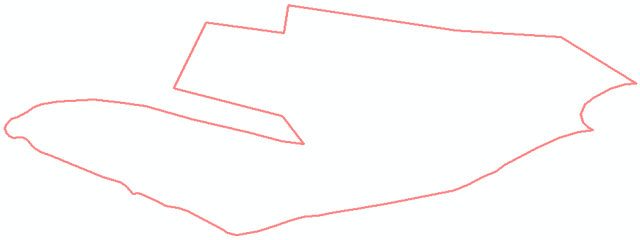


Figure 3.4. The only error is the exterior gap

And in figure 3.4 is drawn that there's only one error. There's no exterior polygon defined, so the exterior is marked as a gap. This is according to the expectations. But this error is no real error, but is a topological error. ArcGIS has built a tool to mark topological errors as an exception. When you do this, a flag is set "1" in the error-table in the record of that particular error.

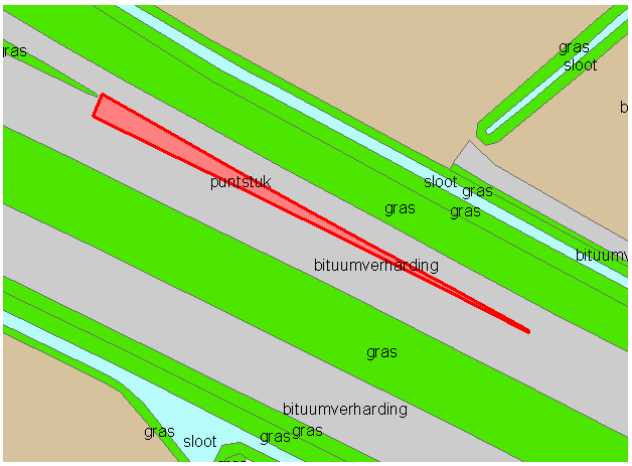


Figure 3.5 Overlap at the marks on the asphalt.

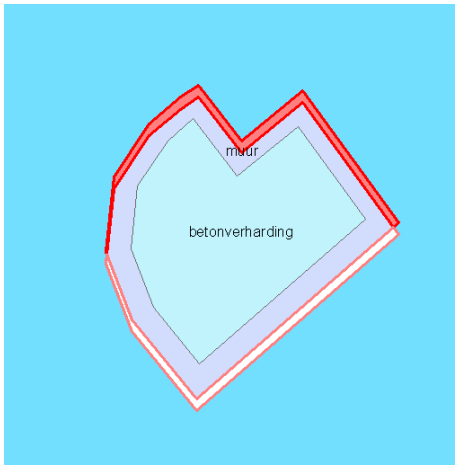


Figure 3.6 A moved object. A gap and overlap is the result.

In the product description of DTB-Nat Rijkswaterstaat mentions that on surface level, the product is a planar graph. With ArcGIS 8.3 you can easily check if this is true, with the same method as

described above. The test is done and a conclusion is that there are a lot of errors in layer one. A lot of polygons overlap and there are numerous gaps. Examples of these gaps and overlap are given in figure 3.5 and 3.6.

3.2 Topological rules based on the semantics

One of the strongest points of the way of treating topology in the ESRI way is that the user is able to define rules, based on the semantics of the data. In stead of rules like "polygons must not overlap", the user can define rules like "buildings must not overlap". One of the advantages of this model is that the user can create its own topology model that meets the needs of the user. In this case, the user is not an end-user, but a user of the Geodatabase Topology, often a "datamanager".

bnat_reg : Verf	Must Be Covered By	dtbnat_reg : Verhar
bnat_reg : Rivieren	Must Not Overlap ...	dtbnat_reg : Bebou
bnat_reg : Rivieren	Must Not Overlap ...	dtbnat_reg : Verhar
bnat_reg	Must Not Have Gaps	
bnat_sym : Hoogs...	Point Must Be Cove...	dtbnat_lin : Hoogsp
bnat_lin : Waterniv...	Must Be Covered B...	dtbnat_reg : Riviere
bnat_reg : Rivieren	Boundary Must Be ...	dtbnat_lin : Waterni
bnat_lin : Hoogspa...	Must Not Have Dan...	
bnat_reg : Rivieren	Must Not Overlap ...	dtbnat_reg : Bouwl.
bnat_reg : Rivieren	Must Not Overlap ...	dtbnat_reg : Verf
bnat_reg : Rivieren	Must Not Overlap ...	dtbnat_reg : Gras

Figure 3.7 Used topology rules based on semantics

Addition of Laser-Scan (ref: e-mail Woodsford e.a.): The topological model in ESRI's Geodatabase is no more or less fixed than Radius Topology. This cannot be said to be an advantage.

On the complete dataset of DTB_nat, some topology rules based on the semantics are formulated. There are a lot of possible rules available in ArcGIS. But when you start implementing your topology rules, some shortcomings arise in the software. For instance, it is possible to implement the rule "POINT_OBJECT" must be in "POLYGON_OBJECT". But the negative rule is not possible. So, the rule "TREES" must not be in "WATER" could not be implemented. In figure 3.7, a list of the successful implemented rules is given. These rules include point objects, line objects and planar objects.

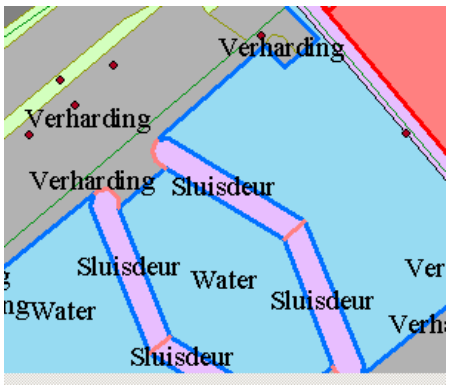


Figure 3.8 Topological errors. Are they real errors or is it a bad rule?

After implementing the rules and after the validation process, the objects that do not meet the conditions are displayed in red. About 600 errors were found in the geodatabase. Perhaps, not all the errors are real errors (in the user's eyes, but they are real topological errors). For instance, many errors appeared due to the rule "WATERLEVEL_LINE must be covered by boundary of "RIVERS" (some of these errors are drawn in figure 3.8. But the question is now, are these errors real errors, or is it a bad rule?

Solutions to solve topological errors

Now there is a geodatabase that contains errors according to the implemented topological conditions. The next step is to remove these errors. There are two possibilities. The first is to mark the error as an exception (the error is not a real error), or second, solve the error. ArcGIS 8.3 contains a toolbox to remove these topological errors. For instance, if there's a gap, ArcGIS gives the option to create a feature in the gap. This works well in some cases, but sometimes you can see that the error can be better fixed with another method (for instance, to move an object or

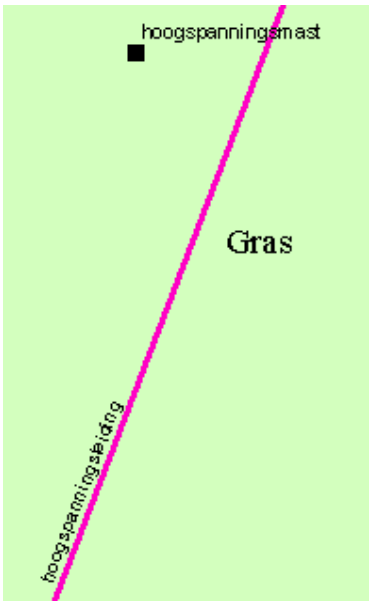


Figure 3.9 A pylon that is not on a power line. An error with insufficient tools to solve.

put an object into another layer). ArcGIS just has a brief toolkit to solve errors. In many cases, this toolkit is not sufficient. An example is given in figure 3.9.

In this example you want to snap the pylon (Dutch: hoogspanningsmast) to the power line (Dutch: hoogspanningsleiding) or otherwise, snap the power line to the pylon. It is possible to do this action, but there's no easy tool available. You have to move one object to another and validate the topology again and hope that the distance between the pylon and the power line is lower than the tolerance level.

Addition of ESRI (ref: e-mail E.Hoel): ArcGIS topology comes with a large collection of tools specifically intended to address topology errors. These tools are intended to accelerate the process of fixing the errors. It is important however to realize that in many situations, the existing generic editing tools are the most appropriate mechanism for addressing the topology error - i.e., the situation posed in Figure 3.8 - the editor has tools by which to snap an existing point to the nearest line. Finally, the topology error correcting tool framework is user extensible. Users may author their own tools for correcting particular topology errors and register them with the topology error correcting framework. These user authored tools will then be appropriately displayed in the context menus when a topology error is highlighted.

3.3 Advantages and disadvantages of the Geodatabase topology

After doing the tests described above, you can mention some advantages and disadvantages about the Geodatabase topology of ESRI and the way ArcGIS is treating the topology.

Some advantages of the Geodatabase topology are:

- The topology is not fixed. The user decides which topology is important for his dataset. The user has to say on which conditions the data has to suffice.
- The user decides what should happen to objects if they do not suffice all the conditions. Topological errors are not solved automatically, but the user has to solve errors manually.
- The validation process is relatively short. It just took about one hour to validate the entire dataset. A remark has to be made, that with this validation, only the errors are marked, not solved.
- After editing some objects in the geodatabase, not the entire dataset has to be validated again. The edited area is marked as a dirty area and only the dirty areas have to be validated again.
- The Geodatabase topology is easy to use. You don't need to have special knowledge about topology to create a set of logical conditions for the data. And also the wizard to implement the topological rules is very useful.

Comments of Laser-Scan (ref: e-mail Woodsford e.a.):

- A system that relies on users to resolve topological errors cannot, in any circumstances, be said to have an advantage over a system that will automatically solve topological errors without further user intervention.
- What is the true cost of the validation process? This process has only highlighted the problems, it still requires a human operator to resolve them. Comparing like with like, it needs to be identified how long it took to find and resolve the topological errors.

- Radius Topology does not require to be revalidated. In fact the opposite is true, as it will continuously validate new or edited features 'behind the scenes'. Therefore the requirement for Geodatabase to validate 'dirty areas' again is a disadvantage. Radius Topology automatically checks and cleans the ‘dirty areas’ as the updates are supplied. This cannot be said to be a comparative advantage of Geodatabase Topology.

Some disadvantages of the Geodatabase topology are:

- After validation of the topology, the dataset still does not satisfy the conditions. Only the errors are marked. So, at least one extra step is necessary.
- If there appears an object that does not meet the defined conditions, you have to be aware that this does not have to mean that there’s a topological error. It is also possible that the defined condition is not defined well.
- The toolkit in ArcGIS to solve the topological errors is not very extensive. Some errors can be solved, but for many errors a more advanced toolkit is necessary.
- Not all the possible topological rules can be implemented. For instance the rule “POINT_OBJECT must not be in POLYGON” cannot be implemented. To see which rules are implemented, Clementini’s decision tree is used (figure 2.2). The rules that can be implemented with ArcGIS 8.3, can be found in Appendix B. In this appendix, also the topological relations (disjoint, in, touch, cross, overlap) are drawn. According to Clementini, 52 possible topological relationships exist. In ArcGIS 8.3, only 25 are implemented and a lot of the rules are double (for instance “must not overlap” and “must not overlap with”).

Comments of ESRI (ref: e-mail Hoel)

- bullet 1: The features that are not associated with topology errors are topologically correct following validation. Only those feature that are associated with errors are not topologically correct and will need to be addressed in a subsequent validation following the correction of the error conditions.
- bullet 2: This perceived disadvantage is actually a data modeling problem on the user's part (i.e., not understanding the semantics of topology rule). The topology errors are consistent with how the user defined their topology.
- bullet 3: Topology errors are corrected using either a specific error correction tool or the generic editor functionality. In addition, the set of error correction tools is user extensible.
- bullet 4: The collection of topology rules is not complete in the formal sense of the word. The implemented rules were selected based upon extensive communications with real world users of topology. In the real world, some topology rules/relationships that are found in Clementini (or Egenhofer's dimensionally extended 9 intersection model) are simply not found/utilized. It was determined that the utility of implementing the most obscure of these rules was quite minimal, thus, they were not implemented. Finally, the "duplicate" rule is actually two different rules. One rule applied to features within a single feature class, while the second rule applies to features found in two different feature classes.

4. Laser-Scan Radius Topology, the results

In this chapter, the results will be explained about the tests done with Laser-Scans Radius Topology. The theory of Radius Topology is explained in chapter 2. For the tests, Radius Topology is implemented on an Oracle 9i Spatial database. The tests were done with the same dataset as described in chapter 3, the DTB-Nat from Rijkswaterstaat. Laser-Scan Radius Topology version 2.0 is used for this research. It all works on a UNIX server and is accessed with a Windows 2000 PC using Exceed.

4.1 Radius Topology; about how it works

In Radius Topology, you can create a manifold where you define your topological structure. The most common structures are network topology and the planar graph. In the previous chapter, it is said that the six selected polygons should meet the conditions of a planar graph (except for the exterior polygon). So, without errors, it has to be possible to put these polygons into the persisted structure of Radius Topology.

The script

To proof this, a script is used to put the dataset into the topological structure. Another option is to use the more user-friendly wizards to define the tolerances and preferences. The complete script is drawn in Appendix A and explained below. The first thing is to create a manifold with a planar graph.

```
-- Create manifold 'DTB_Nat_man' for DTB_Nat (maintain faces)

CALL lsl_topo_manifold.create_manifold(
    'DTB_Nat_man',NULL,97400,413100,97600,413300,0.00001,1,1,
     NULL,0.0001,0.0001,0.0001,'USERS','INDX','INDX',0,NULL);
```

In the function create_manifold, some parameters has to be given. You can find these in the Supplied PL/SQL Packages Reference of the Radius Topology Guide. The main parameters are the boundary box (`97400,413100,97600,413300`) and the tolerances for the Share Node, Node-Split-Edge and Edge-Split-Edge constraints. All these tolerances are set on 0.0001. These constraints are explained in paragraph 2.2.

The second step in the script is that you have to create classes. I only used one class, but it is possible to define more than one class (for instance if you use more than one layer).

```
-- Create a single class 'DTB_Nat_laag1' (tols: 0.0001 m = 0.1 mm)

DECLARE
  man_id  INTEGER;
BEGIN
  lsl_topo_manifold.get_manifold_id_from_name('DTB_Nat_man',man_id);

  EXECUTE IMMEDIATE 'INSERT INTO lsl_class$'||man_id||
                    ' VALUES(''DTB_Nat_laag1'',102,20,10)';

  EXECUTE IMMEDIATE 'INSERT INTO lsl_rule$'||man_id||
                    ' VALUES(102,102,0.0001,0.0001,0.0001)';
  COMMIT;
END;
/
```

```
show errors
```

In this part of the script you have to set priorities on classes. If you have more than one class, the class with the lower priority snaps to the class with higher priority in case two objects are within a certain tolerance. In this test, only one class is used, so the objects will snap to each other if they are within a tolerance of 0.0001 meter.

Laser-Scan (ref: e-mail Woodsford e.a.) says that the use of a single class and tiny rule tolerances gives an extremely limited view of the semantic modeling possibilities within Radius Topology.

In the next part of the script, you locate the geometry, which is necessary for creating the topology. In the other part mentioned below, finally the geometry is put into different tables with geometry and relationships.

```
-- Activate 'shape' column, add topo_id and mani_id columns

CALL lsl_topo_struct.upgrade_table('DTB_Nat',
    'geometry',NULL,'geometry','topo_id','mani_id',
    NULL,'''DTB_Nat_man''',NULL,'''DTB_Nat_laag1''');

-- Create topology for feature table

CALL lsl_topo_struct.structure_in_place('DTB_Nat','geometry');
```

The calculation process to create the topology takes a lot of time. In this case, about 22000 objects had to be put into many different tables. It took about 48 hours to calculate all the topology. The result was a list of tables:

```
LSL_AREA_TO_FACE$6
LSL_CLASS$6
LSL_EDGE$6
LSL_EDGE_TO_EDGE$6
LSL_EDGE_TO_NODE$6
LSL_FACE$6
LSL_FACE_TO_EDGE$6
LSL_LINE_TO_EDGE$6
LSL_LOCK_TCN$6
LSL_NODE$6
LSL_RULE$6
LSL_TOPO$6
LSL_TOPO_PART$6
```

And on the input feature table, two attributes were added, a manifold ID and a Topology ID. So every object refers to a topological structure.

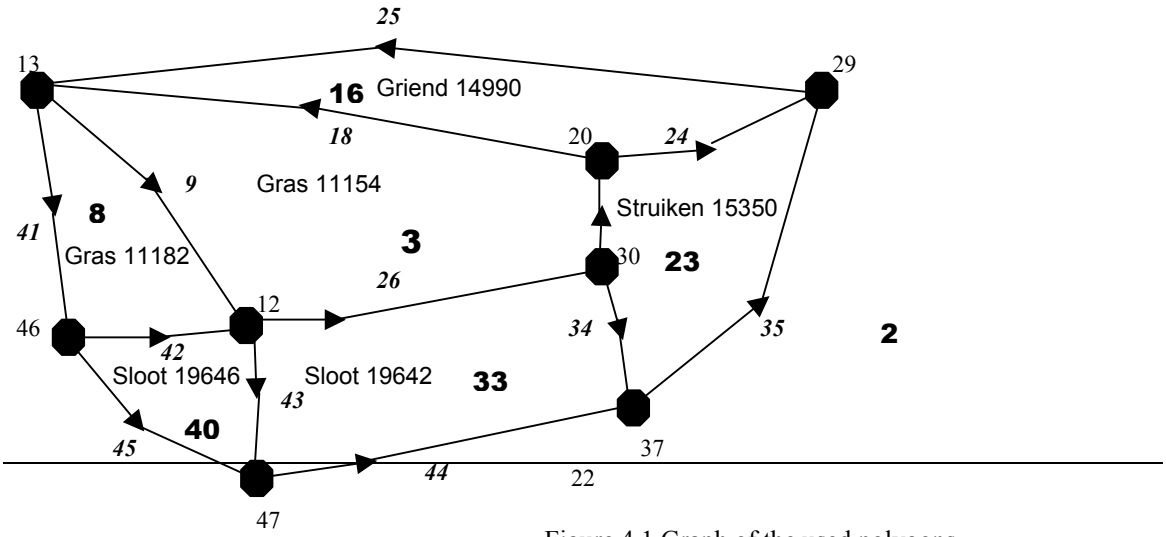


Figure 4.1 Graph of the used polygons

The datastructure

In the previous part of this paragraph, the steps to go from Oracle geometry to Radius Topology are described. In this part, the topological structure, the result of the used script, of the six polygons (a planar graph) that were put into Radius Topology are explained.

When you look into the tables, you can check the topological structure. The topological graph is drawn in figure 4.1. In the feature table, a topology id and a manifold id are added. There are 3 tables with the primitives:

- LSL_EDGE: In this table the geometry of the edges is stored.
- LSL_NODE: In this table the geometry of the nodes is stored and also a node if it's not connected to an edge.
- LSL_FACE: Except an ID, the geometry version is stored. This is just an index that's used when the topology primitives are updated.

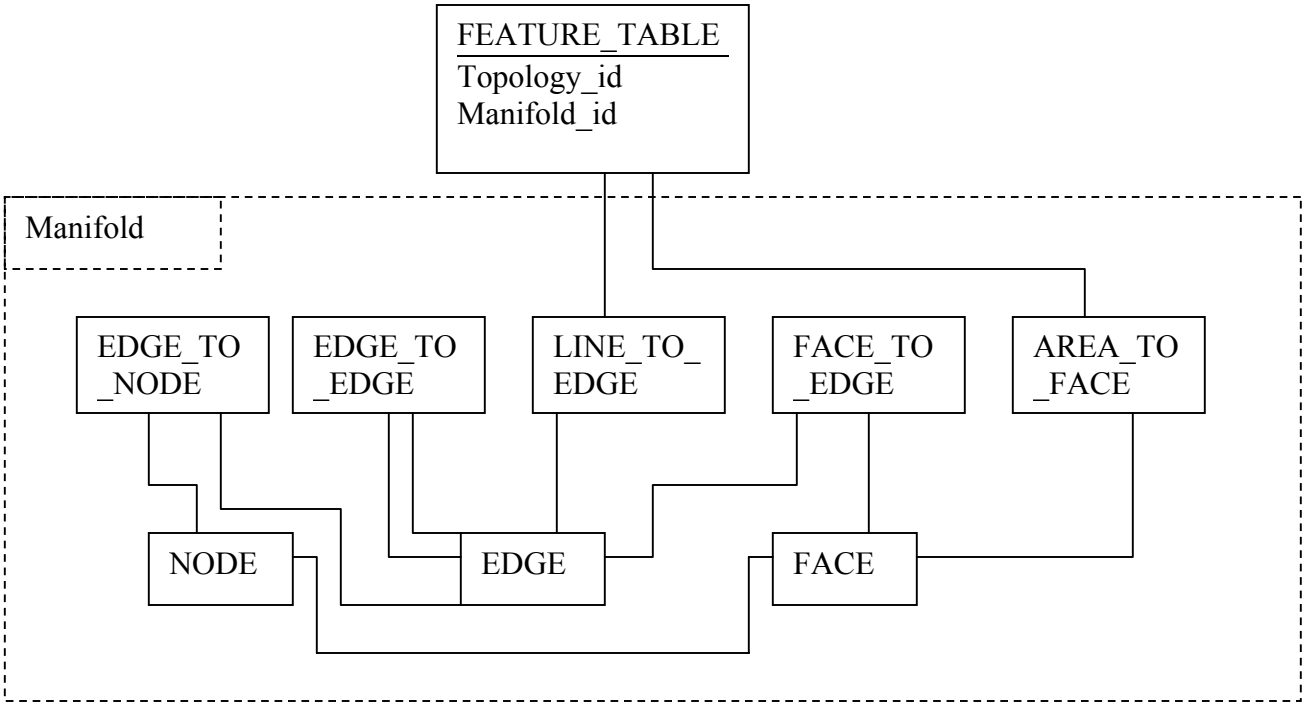


Figure 4.2 Connections between tables in Radius

The topological structure is completed with a some "reference-tables":

- LSL_AREA_TO_FACE: In this table, the areas are defined. Areas are a group of faces, which are somehow related.
- LSL_LINE_TO_EDGE: A line consists of multiple edges, and the relationships between them are defined by edge-to-line and edge-to-edge references in the line-to-edge Reference table. This table joins the Topology part table with the Edge table, and is used to find lines that share an edge and to find edges that define a line. This table contains a flag, which is 0 when the line to edge reference is used reversed and 1 when it's used forward.
- LSL_EDGE_TO_NODE: This table implements the edge-to-node and node-to-edge references. This table also contains a flag, which is 0 when the node is a starting node and is 1 when the node is an ending node.

- LSL_EDGE_TO_EDGE: This table implements the edge-to-edge-clockwise and the edge-to-edge-anticlockwise references. It defines the ordered set of directed edges that define a line or ring. In effect, the table describes what is sometimes known as wing-edged topology. This table also contains a flag, which describes how two edges are referred to each other.
- LSL_FACE_TO_EDGE: It describes to which face, each edge is referred. Also this table contains a flag, 0 for a face on the left side of the edge and 1 for a face on the right side.

In figure 4.2, the connections between all the tables are drawn. As you can see, the edge is the main primitive. Nearly everything is related to the table with edges. In figure 4.2, not all the tables are drawn. There are also some other tables, which are not really important for the relationships between the primitives. In the non-drawn tables, the metadata is described and the priorities between the defined classes (in this case, only one class is defined, so the priorities are not important in this case).

4.2 Gaps and overlap in Radius Topology

During the implementation of the six polygons, no problems raised. And this is exactly according to the expectations, because the input was correct. You put a planar graph into a planar structure. The next step of the test is moving one object in a way that there rises a gap and overlap. The input polygons are drawn in figure 4.3. The expectation is that Radius at least has to do something with the gap and the overlap.

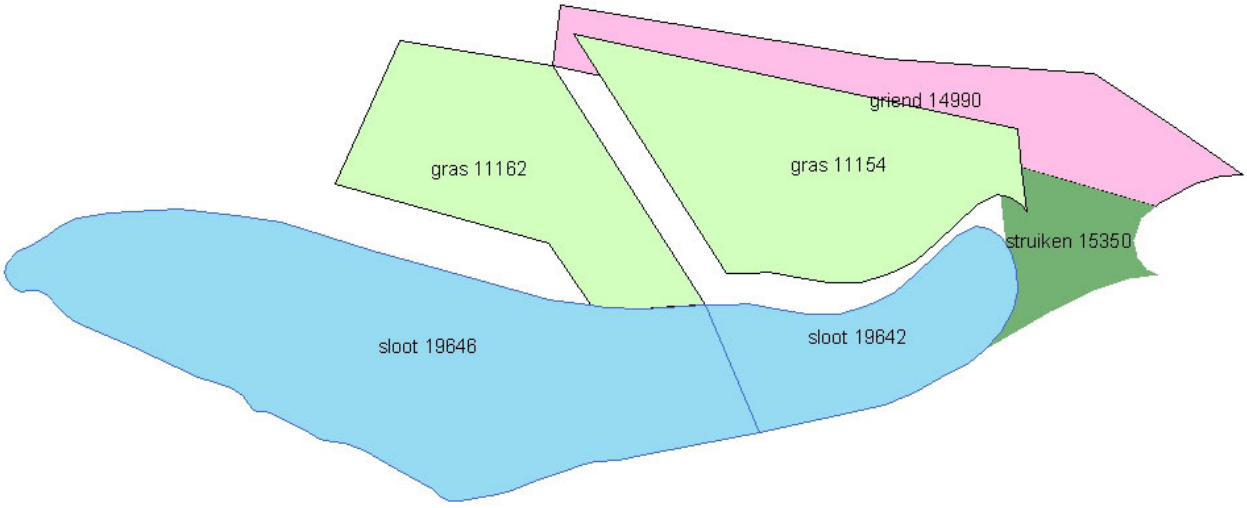


Figure 4.3 The input polygons with one moved object.

During the insertion of these polygons, no problems raised. Radius accepted the six polygons. But after analyzing the tables, you could see that on the places where the polygons overlap, extra faces were created and at the gap, also a face raised. So, the topological structure looked like figure 4.4.

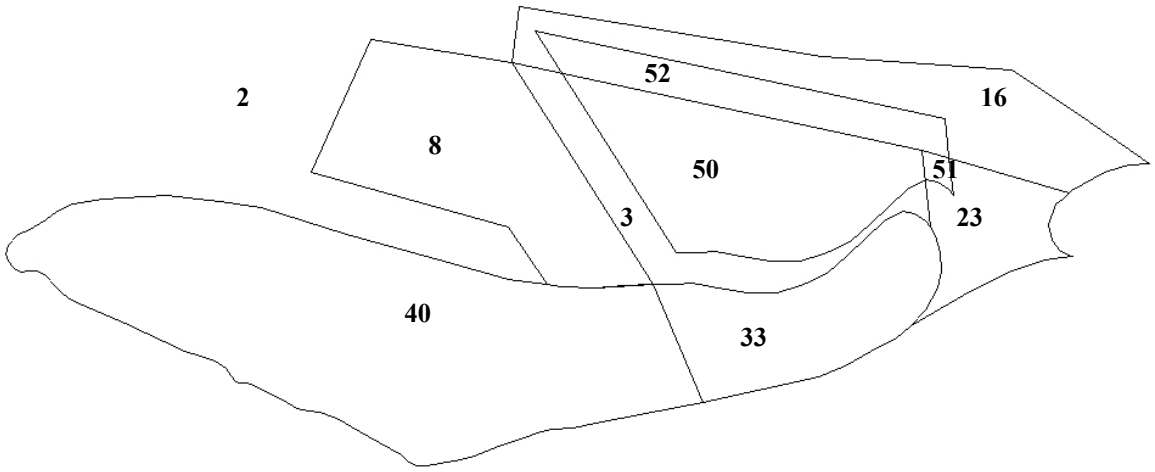


Figure 4.5 The faces in Radius after moving an object.

These are not strange results; as long as there are areas defined that represent the objects. The LSL_AREA_TO_FACE table looks like this:

LSL_AREA_TO_FACE

AREA_ID	FACE_ID
6	3
14	8
21	16
31	23
38	33
48	40
65	52
65	51
65	50
6	50
21	52
31	51
68	8
71	52
71	16
73	51
73	23
75	33
78	40

As you can see, for instance, face 8, is mentioned two times with a different area_id. This could mean that:

1. Face 8 belongs to two different areas
2. Both areas are objects in the feature table
3. Only one of the areas is an object, the other one is redundant
4. Both areas are redundant.

For an answer you have to look into the LSL_TOPO_PART table, which is described below. You can see that there are 12 topo_ids defined. But only 6 objects have been put into Radius.

LSL_TOPO_PART

ID	GEOM_TYPE	NODE	IN_AREA	TOPO_ID
6	2			1
14	2			7
21	2			15
31	2			22
38	2			32
48	2			39
65	2			49
68	2			66
71	2			69
73	2			72
75	2			74
78	2			76

The areas 14 and 68 have got topo_ids 7 and 66. In the feature table (drawn below), only object with topo_id 66 exists. You can say that area 14 is redundant. And when you look further to other faces, areas and objects, you can see that there are numerous redundant areas and topo_ids defined.

Laser-Scan (ref: e-mail Woodsford e.a.) says that the reason the table is showing redundant data is because the data has unnecessarily been re-validated.

FEATURE TABLE

<GEOMETRY>	RECORDNUMBER	<ATTRIBUTES>	TOPO_ID	MANI_ID
SDO_GEOMETRY	1		49	7
SDO_GEOMETRY	2		66	7
SDO_GEOMETRY	3		69	7
SDO_GEOMETRY	4		72	7
SDO_GEOMETRY	5		74	7
SDO_GEOMETRY	6		76	7

A conclusion is that there's a lot of redundant storage in the tables of Radius Topology. The gap in the data, is also defined as part of an area (together with face 50), not as an object. In the overlapping faces, the face belongs to multiple areas and to multiple objects.

Queries

Querying the tables works well in Radius. For instance getting the geometry of the objects. You can do this with the SQL statement:

```
SQL> select lsl_topo_util.face_get_geometry(7,topo_id,2) from stukje2;
```

The answer is six areas with the coordinates of the edges.

```
LSL_TOPO_UTIL.FACE_GET_GEOMETRY(7,TOPO_ID,2)(SDO_GTYPE, SDO_SRID,
SDO_POINT(X, Y
------------------------------------------------------------------------
---------
SDO_GEOMETRY(2003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 1003, 1),
SDO_ORDINATE_ARR
AY(97400, 413100, 97600, 413100, 97600, 413300, 97400, 413300, 97400,
413100))

SDO_GEOMETRY(2003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 1003, 1),
SDO_ORDINATE_ARR
AY(97400, 413100, 97600, 413100, 97600, 413300, 97400, 413300, 97400,
413100))

SDO_GEOMETRY(2003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 1003, 1),
SDO_ORDINATE_ARR
AY(97400, 413100, 97600, 413100, 97600, 413300, 97400, 413300, 97400,
413100))

SDO_GEOMETRY(2003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 1003, 1),
SDO_ORDINATE_ARR
AY(97400, 413100, 97600, 413100, 97600, 413300, 97400, 413300, 97400,
413100))

SDO_GEOMETRY(2003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 1003, 1),
SDO_ORDINATE_ARR
AY(97400, 413100, 97600, 413100, 97600, 413300, 97400, 413300, 97400,
413100))

SDO_GEOMETRY(2003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 1003, 1),
SDO_ORDINATE_ARR
AY(97400, 413100, 97600, 413100, 97600, 413300, 97400, 413300, 97400,
413100))


6 rows selected.
```

Another query is if there are overlapping areas. We know the answer (yes) already, but let's check:

```
SQL> select a.topo_id from stukje2 a, stukje2 b where
lsl_topo_relate(a.topo_id, b.topo_id, 'share_face',7)='true';

   TOPO_ID
----------
        49
        69
        72
        66
        49
        69
        49
        72
        74
        76
```

```
10 rows selected.
```

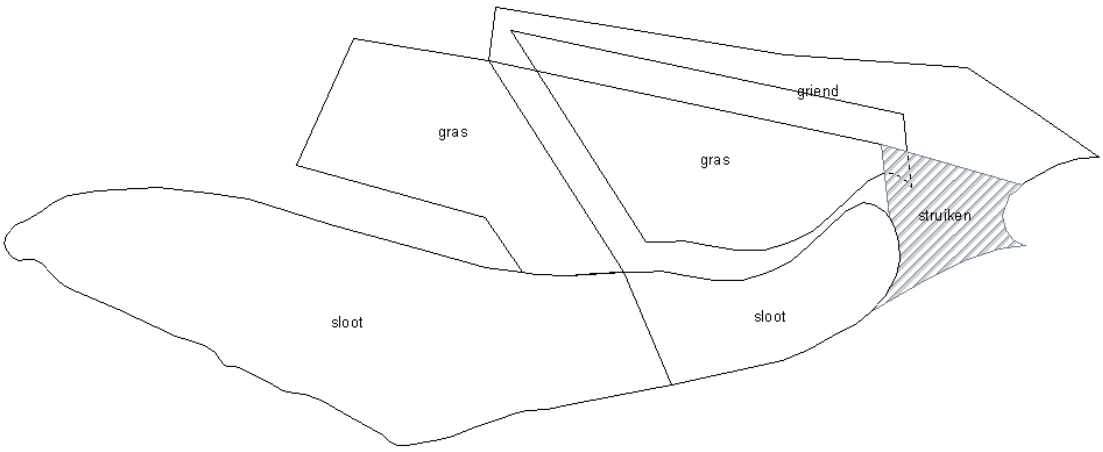


Figure 4.5. Query: Which areas overlap with the marked area?

And of course neighboring questions has to be easy to ask to a topological structured dataset. For instance which objects overlap with object 72 (struiken)? This is drawn in figure 4.5. The answer is:

```
SQL> select topo_id from stukje2 where
lsl_topo_relate(72,topo_id,'face_anyinteract',7)='TRUE';

   TOPO_ID
----------
        49
        72
```

Except for “topo_id”=49 (gras-object), the object overlaps also with itself. The query “which objects interact (neighboring, overlapping, touching) with object 72, gives more results.

```
SQL> select topo_id from stukje2 where
lsl_topo_relate(72,topo_id,'anyinteract',7)='TRUE';

   TOPO_ID
----------
        49
        69
        72
        74
```

A conclusion is that the queries work well in Radius Topology. The answers are correct. Working with queries is easy, at least, if you know which SQL-statements you have to use.

Implementation of the complete DTB_Nat dataset

The last test done is to put the complete DTB_Nat into the Radius Topology datastructure. This could be interesting, because this dataset is full of errors and does by far not meet the conditions of a planar graph. The complete dataset consists of about 20.000 polygons. The implementation of these polygons took a long time (more than 48 hours). The error-log table was created with errors. (That was according to the expectations). A conclusion of this test is that building the Radius Topology data structure of a large dataset takes a lot of time.

Comments of Laser-Scan (ref: e-mail Woodsford e.a.): This experiment should be repeated. It is highly anomalous that it took 48 hours to structure just 22000 objects. As described earlier, this is most likely due to a serious configuration error. Laser-Scan would be pleased to help resolve this issue and prevent its re-occurrence as part of our continuous improvement programme

4.3 Advantages and disadvantages of Laser-Scan Radius Topology

Analogue to paragraph 3.4, you can mention some advantages and disadvantages of using Laser-Scan Radius Topology. The lists mentioned below, are based on the tests and on the literature about Radius Topology.

Some advantages of Laser-Scan Radius Topology are:

- The topology is explicitly stored. The user can only put data into a database, when it's topologically correct.
- After putting data into the topological structure, the user doesn't have to worry about topology anymore. The topology is correct. This can be useful for instance when a company sells a dataset to customers. The customer and the seller are sure that the topological structure is correct.
- With SQL statements, you can query the database in an easy way. The answers are correct.
- In spite of the fixed structure of Radius topology, the user can define some elements of the topological structure. For instance, the user can define which topological model he uses, which priorities he gives to objects (which object snaps to which other object), he can define tolerances, etcetera.
- In this topological model, an edge is only stored once in the database.
- Advantage added by Laser-Scan (ref e-mail Woodsford e.a.): Radius Topology is standards-based, open and interoperable and can be a component of a diverse enterprise-wide solution and is therefore not tied up to a single vendor's solution.

Comments of ESRI (ref: e-mail Hoel):

- bullet 2: The same statement can be made concerning geodatabase topology. Once all the features are present in the GDB topology and there are no dirty areas and no topology errors, then the user is guaranteed that the features are topologically correct (with respect to how they defined their topology).
- It is also very important to note that merely storing explicit topological primitives does not guarantee topological correctness. Explicit topology data structure or schema, with associated "built-in" RDBMS integrity rules (primary key, foreign key constraints, etc.) is not sufficient to ensure data integrity. Foreign keys can reference existing, but incorrect primary keys; foreign keys can be null; the geometry of the topology primitives can be inconsistent with the topology relationships; etc. Additional logic must be executed to validate the consistency and integrity of the data structure. Computational geometry (line intersection code, etc.) must be used validate the integrity of the topology relationships with regard to the geometry of the topology primitives - topology is a planar graph, not an arbitrary graph. Data structure and referential integrity constraints alone cannot declare or enforce this planar geometry constraint.
- bullet 4: Note that with geodatabase topology, users can not only control which rules constitute a valid topology, but they may also specify the cluster tolerance as well as the class ranking (these two concepts control how features snap together - e.g., who snaps to whom).
- bullet 5: The edge may only be stored once, but the original feature geometry remains on the feature. Thus, there is redundancy in the storage of geometry.

Some disadvantages of Laser-Scan Radius Topology are:

- The validation process takes a lot of time. But you have to do it only one time (create once, use many).
- The elements of the topological structure that can be defined by the user are limited. For instance, topological rules based on semantics are not possible.
- Due to the topological structure and the multiple reference tables, a lot of redundant storage is carried out. For instance, the coordinates of the nodes are at least three times stored in the database (nodes-table, edges-table and feature-table). There are areas defined, which are no object in the feature table.

Comments of Laser-Scan (ref e-mail Woodsford e.a.): The statement concerning redundant storage is inexact and poorly thought out. While node coordinates may be stored in multiple locations, this pales into insignificance beside the multiple redundant edge geometries stored on each and every line and area feature within the Geodatabase model. This cannot be said to be a comparative disadvantage for Radius Topology. The spurious area features are due to user error.

5. Conclusions and recommendations

In this chapter, an answer is going to be given on the questions described in chapter 1. The main question of this research is:

What are the main differences between a datamodel where topological consistency is implemented with a rule-based topological structure (ESRI) and a datamodel that applies an explicit topological structure (Laser-Scan)?

To answer this question, the following "subquestions" have to be answered:

- ***What are the main theoretical differences between Radius Topology and the ESRI datamodel?***
- ***Which topological rules can be formulated to structure the dataset with both datamodels?***
- ***In which way, ArcGIS 8.3 treats topological errors?***
- ***In which way, Laser-Scan Radius Topology, treats topological errors?***

First, in 5.1, a comparison will be made between Laser-Scan Radius Topology and ESRI Geodatabase Topology. This comparison results from the literature and the tests described in chapters 3 and 4. With this comparison, the questions above will be answered. This comparison leads to a list of conclusions described in 5.2. In 5.3, some recommendations will be given for further research and to improve both Radius Topology and ESRI Geodatabase topology. Some comments, given by Laser-Scan and ESRI, will be given in 5.4.

5.1 Comparison between Geodatabase Topology and Radius Topology

Both, Radius Topology and ESRI Topology have advantages and disadvantages. A conclusion cannot be that one is better than the other one. It depends on the purpose for using the topology. There are a lot of differences between Laser-Scan Radius Topology and ESRI Geodatabase Topology.

The theoretical differences are:

- In an ESRI Geodatabase, objects are stored with only the geometry. The user by means of topology rules adds the topology. In a dataset, stored with Radius Topology, the user has the option to store both the geometry and the topology or to store only the topology.
- Geometrical queries can easier be answered by a datamodel based on geometry than by a model based on topology. Topological queries can easier be answered by a datamodel based on topology.

The rules that can be formulated with ESRI Geodatabase Topology are:

- Rules based on semantics, like "BUILDINGS must not overlap with "WATER"
- Rules, based on the complete dataset, for instance, layer 1 must not contain gaps and must not have overlaps.

The rules that can be formulated with Laser-Scan Radius Topology are:

- Priorities; The object with lower priority snaps to objects with higher priority
- Tolerances for "Edge-split-edge", "share-node" and "node-split-edge".
- Different topologies in different manifolds.

With ArcGIS 8.3, the topology is treated by rules. If you would like to use topology, you have to define a set of rules and validate the dataset on these rules. (rule-based topology) The objects that do not meet the defined conditions are marked and can be edited by the user. The number of tools to edit these "error-objects" is limited. Dirty areas define the non-validated regions. It is possible to validate only a part of the geodatabase, for instance only the dirty areas.

Laser-Scan Radius Topology uses fixed topology. On beyond, the user has to define a topological structure, in which the data has to fit. Once you put the data into this structure, the dataset is topologically correct. The user does not have to do anything afterwards, besides maintaining the data. The user can define some elements of the topological structure (the structure, some priorities and tolerances), but the possibilities are limited. Overlap and gaps in a planar structure leads to redundancy.

In a schema (figure 5.1) the differences between both described methods are drawn. This is also an answer on the main question.

ESRI Geodatabase topology	Laser-Scan Radius Topology
No persisted topological structure	Explicit topological structure
Topology based on rules	Topology defined in a manifold
User decides what to do with errors	User has little influence on what to do with errors
After validation, data is still not in topological structure	After validation, topological structure is correct
Possibility for topology based on semantics	No possibility for topology based on semantics

Figure 5.1 Differences between Geodatabase topology and Radius topology

Uncertainties

Both the Geodatabase concept and Radius Topology have uncertainties. About one uncertainty has been spoken before in this report (chapter 3). If you have defined a topological rule (in the Geodatabase concept) and there are features that do not meet these rules. Then you are not certain if the data is incorrect (a real error) or if the rule is not correct (not a real error). The user has to be aware of this uncertainty while he is using the Geodatabase concept.

Whether you persist topological relationships or not, queries still give incorrect results when you have incorrect data compared to the real world. Both the Geodatabase concept and the Radius concept give tools to minimize the amount of incorrect data. The only thing is that in the Geodatabase concept, the data is validated and then put into a topological structure by the user. In a persisted data structure like Radius Topology, the user has less influence, what makes it easier for the DBMS to check the correctness. (This statement comes from Scott Morehouse and Peter van Oosterom in an e-mail discussion)

5.2 Conclusions

ESRI and Laser-Scan have a different way of treating topology. After the comparison between both methods, some conclusions can be found.

- ESRI Geodatabase Topology is rule-based topology, because the user decides which topological rules are important and what to do with errors.

- In Laser-Scan Radius Topology the influence of the user is limited. Once you defined a topological structure, organized in a manifold, you can hardly influence the way Radius Topology is treating topological errors.

- A combination of Geodatabase topology and Radius topology could also be useful. For instance when you work with the DTB. You edit the data with a set of defined rules in the Geodatabase topology and solve the errors. When you are going to sell a new version of the dataset, you put it into Radius Topology and you are certain that the topology is correct.

Both the described methods have their advantages and disadvantages. Which method has got more future, cannot be concluded.

5.3 Recommendations

After the conclusions, follow some recommendations:

In this research, a comparison is made between Radius Topology and Geodatabase Topology. When the new version of Oracle is released, it is recommended to compare the Oracle Topology with both Radius and Geodatabase topology for a complete overview of topology methods.

Another recommendation for research is to do a performance test on a large dataset. This research was only a functionality test.

I recommend both Laser-Scan and ESRI to take the advantages and disadvantages (from paragraph 3.4 and 4.3) into account to improve their own method of treating topology.

5.4 Comments of Laser-Scan and ESRI

In this paragraph, parts of the reaction of Laser-Scan (ref e-mail Woodsford e.a.) and ESRI (ref e-mail Hoel) will be given, with respect to the uncertainties, conclusions and recommendations mentioned in this chapter.

Reaction of ESRI:

- It is noted that with Radius topology, users may specify priorities to control snapping. The same capability exists with geodatabase topology - r.e., the feature class ranking model. With the geodatabase ranks, the features with lower ranks snap to features with higher ranks if the y are within the cluster tolerance of each other.
- With geodatabase topology, the most appropriate"tools" to correct certain topology errors are the generic editor tools. Additionally, users can augment the ArcGIS framework with their own topology correction tools if they feel it necessary.
- The statement that the "computer fixes errors" is not quite correct. The type of error correction that Radius topology can perform is related to either snapping vertices or not persisting features that are in error. No other fixes are performed. With geodatabase topology, certain errors may analogously be "fixed" during validation if they can be corrected via topological integration (e.g., vertex snapping). The two technologies are very similar in this regard. However, the geodatabase topology does come with a collection of custom tools (which augment the generic editor tools) that may be used to correct topology errors.
- -With geodatabase topology, the data is topologically correct following validation (presuming there are no topology errors). Correctness is not the issue here - Radius merely chooses to persist the topological primitives
- Regarding uncertainties - this is really a data modeling issue. All topology errors are in fact true errors according to how the user defined their topology. If the user considers these to not be "real errors", then they need to reevaluate their data model and how they chose to define their geodatabase topology. For well designed user data models, there is no uncertainty with geodatabase topology.

- The final statement (in the last conclusion) is incorrect. It implies the following:
    1. When data is stored in Radius topology, it is guaranteed to be correct.
    2. Only storing data in Radius topology will guarantee that the data is correct.
    3. You cannot guarantee that your data is correct when storing it in geodatabase topology.

  Each of these three implications is incorrect. As we've noted previously, explicit storage of topological primitives does not guarantee correctness. Also, the data can be topologically correct without the explicit storage of topological primitives.
- Another general issue - although validation times on the test dataset were noted in the report (i.e., 1 hour for geodatabase topology and 48 hours for Radius topology) and the differences were very significant, they were not mentioned in the conclusions. A difference of this magnitude raises a number of questions regarding fundamental performance on large datasets (i.e., 10-100 million features) as well as the scalability of the technology.
- Finally, there are several other significant issues that must be considered when comparing topology technologies. These issues include:
    1. performance
    2. scalability
    3. transaction isolation
    4. multi-user editing
    5. distributed processing

  Failure of an organization to consider these would be a critical mistake.

Reaction of Laser-Scan:

- It is misleading to state that in 'Radius Topology the influence of the user is limited'. This suggests that the system gives limited options for configuration which is clearly untrue. In Radius Topology the job of configuring the spatial data model is expected to be done infrequently with a degree of care and attention by a spatial database administrator or other qualified person. This removes the duty to train every application user, however occasional, in the fine details of topology and spatial data modelling. Therefore, this is an essential security and budgetary requirement of database administration. The rules, if found to be ineffective, may be changed at any time just as for Geodatabase Topology. However, our view is that this is not a typical job for an application user.
- Radius Topology ensures at all times that all the topological relationships are maintained and never broken. These relationships may be established either automatically by Radius because they fall within the prescribed rules and tolerances or as a result of interactive user edits (by one of the many edit clients that can be used with Radius). The topology is unbreakable. If the boundary of a county lies along the river Avon it will lie along the river Avon until this is explicitly changed. If this information is true, queries concerning the boundary will give true results – if it is not true they will give misleading results; similarly, with which roads connect to which roads, or any other topological relationships. In Radius Topology, query results, whether arrived at via geometric computation or via topological relationships, will always give identical answers. The two models are guaranteed to be synchronised at all times.
- The choice of translating a land parcel as the 'use case' in 3.1 and 4.2 is not a meaningful one (and probably reflects inexperience). Parcels are not shifted in reality. The key use cases for parcel maintenance are:
    - Modify (shared) parcel boundaries
    - Split or merge parcels.

  And the key use case for establishing (cleaning) parcel data is: remove/clean slivers and overlaps
- The most significant difference between the modes of topology computation in the two systems is never addressed. When inserting a new feature into the database, Geodatabase technology must perform a number of spatial joins according to the number of rules to determine whether the data conforms with the model. In Radius Topology, a single spatial query is executed on a unique topology primitive table irrespective of the number of rules. This disparity may have a significant impact of the scalability of the two systems when handling complex spatial data models with many 10's or 100's of rules.
-

Literature

Zlatanova, S., Tijssen, T.P.M., Oosterom, P.J.M. van, Quak, C.W., *Research on usability of Oracle Spatial within the RWS organization*. Delft, 2003.

Louwsma, J.H, *Topology versus non-topology storage structures*. Delft, 2003.

Hoel, E., Menon, S., Morehouse, S., *Building a Robust Relational Implementation of Topology*. 8th International Symposium on Spatial and Temporal Databases. Santorini, Greece, 2003.

Laser-Scan. *Technical Product Description - Topology Concepts*, Cambridge, UK, 2003.

Laser-Scan *Technical Product Description - Topology Users Guide*, Cambridge, UK, 2003.

Clementini, E., Di Felice, P., & van Oosterom, P. (1993). *A small set of formal topological relationships suitable for end-user interaction*. In D. Abel, & B. C. Ooi (Eds.), Advances in spatial databases - Third International Symposium, SSD'93, Singapore (pp. 277-295). Berlin: Springer.

Egenhofer, M. J. and J. R. Herring,(1990): *A mathematical framework for the definition of topological relationships*, in: Proceedings of Fourth International Symposium on SDH, Zurich, Switzerland, pp. 803-813

ArcGIS Desktop Help

E-mail discussion between P.J.M. van Oosterom (TU Delft) and S. Morehouse (ESRI)

http://www.radius.laser-scan.com/about/index.htm (July 2003)

E-mail from Laser-Scan (P.A. Woodsford, D. Warner, P.J. Watsen) as a reaction on a previous version of this report (24 October 2003)

E-mail from ESRI (E. Hoel) as a reaction on a previous version of this report (27 October 2003)

Appendix A: The used script for Radius Topology

```
spool mk_dtbreg_topo.log
set lines 128
set pages 500
set trim on
set trimspool on
set timing on
set serveroutput on
set echo on

select count(*) from stukje2;

-- Create manifold 'stukje_man' for stukje (maintain faces)

CALL lsl_topo_manifold.create_manifold(
    'stukje_man',NULL,97400,413100,97600,413300,0.00001,1,1,
     NULL,0.0001,0.0001,0.0001,'USERS','INDX','INDX',0,NULL);

-- Create a single class 'stukje_laag1' (tols: 0.0001 m = 0.1 mm)

DECLARE
  man_id  INTEGER;
BEGIN
  lsl_topo_manifold.get_manifold_id_from_name('stukje_man',man_id);

  EXECUTE IMMEDIATE 'INSERT INTO lsl_class$'||man_id||
                    ' VALUES(''stukje_laag1'',102,20,10)';

  EXECUTE IMMEDIATE 'INSERT INTO lsl_rule$'||man_id||
                    ' VALUES(102,102,0.0001,0.0001,0.0001)';
  COMMIT;
END;
/
show errors

-- Activate 'shape' column, add topo_id and mani_id columns

CALL lsl_topo_struct.upgrade_table('stukje2',
    'geometry',NULL,'geometry','topo_id','mani_id',
    NULL,'''stukje_man''',NULL,'''stukje_laag1''');

-- Create topology for feature table

CALL lsl_topo_struct.structure_in_place('stukje2','geometry');

-- Check results

select count(*) from stukje2 where topo_id is NULL;

select feature_table,topo_id,manifold_id,operation,
    error_msg,to_char(time,'DD-MM-YYYY HH24:MI:SS.FF')
  from user_lsl_error where feature_table = 'stukje2';

spool off
exit
```

Appendix B: Possible topology rules in ArcGIS 8.3

Area rules	**Relation**	**Topological relation**
Must not overlap	A	disjoint
Contains points	A:P	in
Must be covered by feature class of	A:A	in
Must not overlap with	A:A	disjoint
Must not have gaps	A	overlap
Boundary must be covered by	A:L	in
Must be covered by	A:A	in
Must cover each other	A:A	in
Area boundary must be covered by boundary of	A:A	in
Line rules		
Must not have dangles	L	touch
Must not overlap	L	disjoint
Must not intersect	L	disjoint
Must not intersect or touch interior	L	disjoint
Must not overlap with	L:L	disjoint
Endpoint must be covered by	L:P	overlap
Must not have pseudo-nodes	L	none
Must not self-overlap	L	disjoint
Must not self-intersect	L	disjoint
Must be single part	L	disjoint
Must be covered by feature class of	L:L	in
Must be covered by boundary of	L:A	in
Point rules		
Must be properly inside polygons	P:A	in
Must by covered by boundary of	P:A	in
Must be covered by endpoint of	P:L	touch
Point must be covered by Line	P:L	in

A=Area, L=Line, P=Point
Topological relationships based on Clementini (figure 2.2)