3. TOPOLOGY - OBJECT - RELATION - GRAPH (TORG)

The previous section explained why there is a relationship between topology and objects. This section discusses the use a graph to represent this relationship between objects and topology. The graph is named : ***topology - object - relation - graph*** or TORG in short. The term TORG is used when referring to the graph or the principle. TOR (Topology-Object-Relation) is used when referring to just the topological relationships between objects.

Advantage of TORG is that no difficult operations have to be performed (like intersection of polygons etc) till the required structure has been found (Finke, et al, 1993). TORG is an extension of the object - oriented data collection and description. The database that contains this information is populated at the same time when topology is constructed.

Another advantage is that TORG can also be implemented in quadtree based systems where normally topology cannot be , because it is based on the relationship between objects and not on the outline of polygons. At the final stage of the search / modelling when it is necessary to show areas of intersection the "standard" polygon intersection methods may need to be used to get a representation of a solution to a problem.

The GIS keeps tracks where polygons / objects intersect. The TORG is not about trying to see if let say a point is in a polygon or a line intersects another line. The initial spatial position of objects has to be established before TORG can be used. TORG is about manipulating / querying existing / recorded topological relationships between objects. Because topological relationships are invariant the shape of the objects does not matter. Later positions and relationships can be changed and the TORG upgraded.

The TORG-base, the database that holds all the TORG information, could be described as a database containing the topology, relationships with adjacent objects of each object in graph form.

It can safely be stated that 99 % of the boundaries on a solid geology map are interpreted. So we are more concerned with the relationships between rock types. Therefore geological GIS interpretations should be more based, at least in initial stage, on topological relationships than exact coordinates. A continuous classification as proposed by McBratney & De Gruijter (1992) based on fuzzy set approach may be helpful, to define possible geological boundaries.

A boundary between two rock types is often uncertain (how uncertain is uncertain ; all boundaries between rocktypes are for 99 % interpretation anyway ; especially in gradual change. However the two polygons representing the rocks do not meet or touch but should overlap. To quantify the overlap is difficult because creates another class.

Adjacent rock types mostly have fuzzy boundaries one could possibly say the classifications overlap. The boundary between a rock unit and a fault may also be fuzzy. In the 2D plane a fault is in general represented as a line, while in reality it is a polygon that could be displayed on a map if the scale is large enough. A fault may have zero width or maybe a kilometer wide. A fault system represents an object-class. For example in Fig 9 a fault is determined at five different localities in between these points the fault is been interpreted. These five locations may differ in style width and composition but are all part of the same fault system. The boundaries between the various parts are uncertain, and maybe non existent because a gradual change from one to the next (very fuzzy boundaries). Is the use of partially ordered sets (posets) (Kainz et al 1993) a solution for this type of classification?.

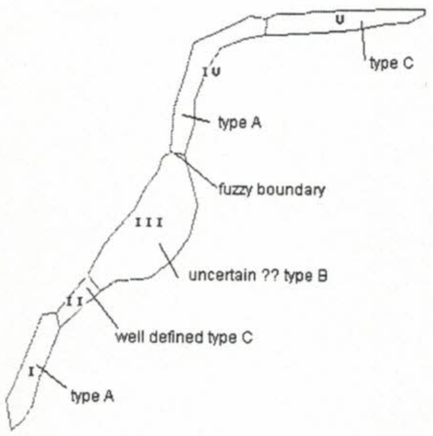

Fig 9 : Fault object system.
Parts I - V are boards along the joist.

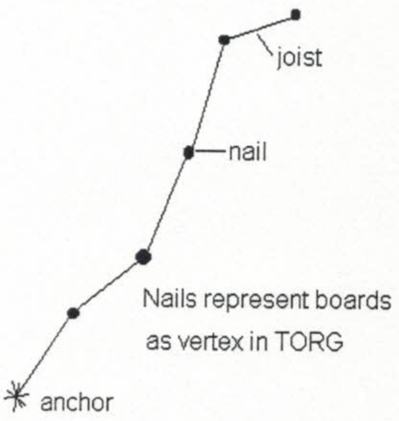

Fig 9 A : TORG of fig 9

In the Fig 9 an 9A example the fault is built up of five topological units namely the fault was "sampled" at these five locations. These five units have a topological relationship, that is they form a *joist* where

I meets II   but I does not meet III, IV, V

II meets I & III but II does not meet IV & V and so on.

It should be noted that the fault system can have parts of the same type. Objects can be build up of more than one topological entity especially strings. For example faults build then as a series of concatenated topological units . In this way a fault system (object "fault") can be built up of many sub- objects , some of which are interpreted, others are delineated, others are inferred. The various building blocks can signify a narrow width with little movement or a kilometers wide shear zone with a mylonite zone (fig 9). These objects form together a super-object.

The topological relation of the objects rock types adjacent to a fault are important. In reality no two faults, rock type associations, or mineralised areas are the same, however certain geological features are important for the forming of orebodies. So in a model driven search a pattern of rock types next to faults can be used to find prospective areas. The model is the caricature of reality.

An area of known mineralisation is selected. The polygons that contain rock types and fault associated with this mineralisation will form the control-object (c-o) that will be used in the pattern search. A library of control-objects can be setup, i.e. the control-object does not have to part be of the map or dataset one investigates but can originate from Canada or Africa.

Worboys & Bofakos (1993) state that object classes form an inheritance hierarchy where subclasses inherit operations from superclasses. Objects may be aggregated into composites , where composites depend for their existence upon their components. Regularisation results in the elimination of points, line objects, cuts and punctures and the inclusion of full boundaries of the areal objects. However in TORG there is no need for regularisation, can use all objects points etc., some of them forming super objects.

In the development of topological analytical procedures it may be useful to represent the geological objects as vertices of a graph and the spatial relationships between adjacent geological objects as edges.

As in the case of the chair, described earlier, there is a relationship between the 8 building blocks which can be represented in graph form (fig 4, 5). The vertices then represent the different classes (polygons and arcs). Make a table (table 1) that records number of adjacent vertices and to which class they belong. Fig 6 & 7 are topological different but the representation in a graph (Fig 6A & 7A) seen from vertex 1 are the same. But in combination with the topological relation of the vertices fig 10 a & b the unique relationship is preserved. Following these above one could convert a geological map to a connected multi-graph (see Fig 18). A <u>multi-</u>

graph is a graph in which more than one edge may exist between two given vertices (Gondran & Minoux 1990)

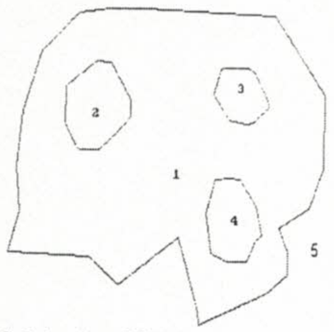

Fig 6 : 1 = polygon ; 2, 3, 4 polygons inside polygon 1; 5 = world

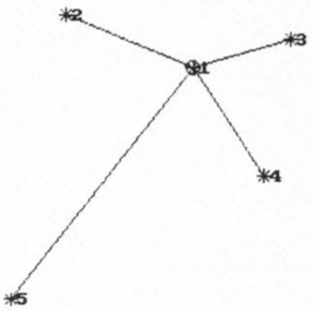

Fig 6A : A graph representing the poygons of fig 6. Based on relationships that poly 1 has with the others

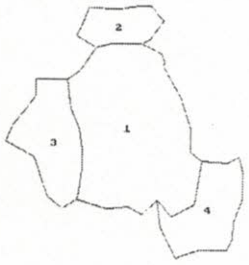

Fig 7 :
1, 2, 3, 4 : polygons meeting
2, 3, 4 : polygons meeting poly 1, but do not meet each other.
5 : "world"

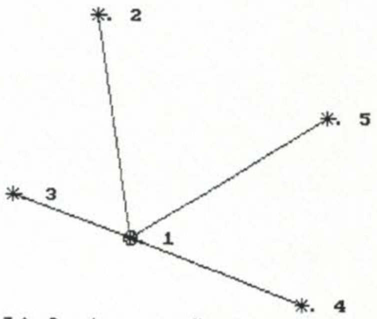

Fig 7 A : Graph representing the polygons of fig 7 based on the realtionships that poly has with the others

object_seq_no	NO_seq_no	object_class
primary key	*primary and foreign key*	
Table 1		

The constructed graph has to be a connected graph Gloss, because all objects in the space have a spatial relationship. However a map of only faults can lead to the construction of a connected or disconnected graph, because faults are strings, that are not necessarily connected.

There are two complementary / supplementary topological systems :

1. Is standard about polygon left & right of arc, point inside / outside polygon etc. This is to build the drawing, intersections of layers, as used in various GIS's

2. Is object-topology, the spatial relationships between objects. The objects may be polygons, arcs or point, that a polygon is built up of 1 or 50 arcs does not change the object only the relationships with adjacent objects of same or different class.

The distance d_{UV}, between two vertices u and v in a graph G, is equal to the number of edges in the path with the shortest number of edges connecting u and v (Foulds 1992) . In TORG the neighborhood of a vertex could be defined as all vertices within a certain distance from a vertex in a graph. In TORG distances are integers, representing the number of nails away from an anchor.

The TORG index is sorted according to vertex degree. The composite object graph can only be reduced to graphs of each vertex showing its relationships with its adjacent vertices. This minimal component of the composite object graph is a graphical representation of what is held in the database. When a new item is inserted or an item is subdivided the TOR between all the parts and neighbors has to be re-established.

The objects A & B have a class and are of a type which is recorded in the object - oriented GIS. In TOR is stored only the object_id, and if the object is 0_dim (point), 1_dim (arc) or 2_dim (polygon), and the relationship with its adjacent objects and their id's[1].

The TORG's are in a sense digraphs because relationships of the central vertex with the adjacent vertices are recorded only. In TORG the edges are used to indicate the type of relationships between adjacent object and the number of vertices indicate the number of different types of relationships and / or number of a certain relationship there are between two adjacent objects.

Important to record that a vertex of a graph is a part of a joist, and is indexed. A joist can be an important relationship between objects of same or different classes. Separate tables are created to record them. A joist can also be used temporarily when carrying out a search.

According to Egenhofer et al (1994b) the overall goal in modelling is to reduce the complexity of an object, therefore the number of significant parts should become smaller. For objects with holes, this means that

[1] The convention is used that "seq_no" is a unique identifier generated by the system and an "_id" is a unique identifier generated by the user

the number of holes will be reduced, either by aggregating two or more holes into a single hole, or by discarding a less significant hole. This guides any decision about the direction into which the invariants should change. The number of holes should decrease towards zero, otherwise additional complexity would be introduced. Likewise, the number of boundary-boundary components between two holes or between the generalised region and a hole should get smaller.

The assessment of relation homeomorphism for regions with holes is more complex, because it is necessary to guarantee that all relations in which holes are involved, are preserved. Holes that are not involved are those that are disjoint from the other object's generalised region. Such holes do not influence the topological relation between the complex objects and, therefore, may be dropped as one generalises from one level to a less detailed level. All other relations among the generalised regions and the holes must remain the same. In the TORG method they are automatically not part of the search graph.

It is unavoidable in any modelling assumptions and simplifications are made, so further simplification moves the model even further away from reality. The reduction of holes to reduce complexity is not a desirable answer to the problem. In the TORG approach there should be no need for reduction in complexity. Except if e.g. detailed maps are compiled into regional scale maps where detail is unusable. But then detailed super objects are represented as a combo. In comparison using TORG does not change the topological relationship only the way they meet. So there is a need to indicate in the TORG tables if an object is a combo even if the object originally was not one. Later when more information is available or on a different scale it maybe "subdivided" and be represented as a combo. So now in the table the combo ID can be entered.

Egenhofer and others seem to imply that the class of the hole is the same as the class of the surrounding polygon. Egenhofer also talks about bounded and unbounded. To clarify all this we need to answer the question : "What is a hole ?". If one argues that a hole is an object of another class (could be empty, world or something else) then in the case it is not empty and not of the same class one can get overlap. If there are two holes of an empty class or the same class meet then one still has to be careful in merging these two because there may have originally a reason why these holes were separate in the first place. It could for example be that these holes only meet temporarily. In real situations the space between and surrounding two objects is not empty, it is something even if it is just called background. So that space that is often called 'world' when empty can be part of TORG. Naturally the world vertex can also be queried.

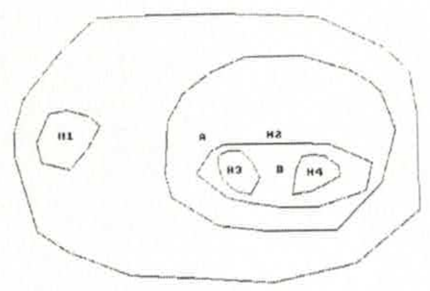

Fig 11 : Figures base on Egenhofer, Clementini & Di Felice (1994) dealing with holes.
H1, H2, H3, H4 : holes

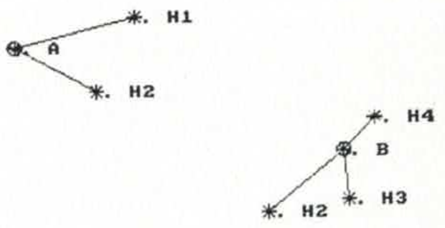

Fig 11 A & B : TORG for point A & B

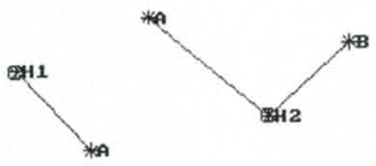

Fig 11H1&2 : TORG for points H1 & H2

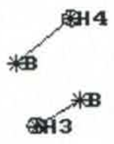

Fig 11H3&4 : TORG for points H3 & H4

As mentioned before, in his paper Egenhofer et al (1994b) explains that it is necessary to remove holes in objects in order to reduce complexity. In TORG this is not necessary and holes can easily be accommodated. However if there is an island in the hole, the n the complexity becomes different and holes cannot be removed. The anchor of a hole has only a "coveredBy" relationship with its surrounding object. If objects are disjoint they do not appear in their respective adjacency tables. This is different from the Egenhofer et al (1994b) approach where the disjoint holes are recorded. Recording disjoint holes would create data redundancy in the TORG database.

The approach by Egenhofer et al (1994b) regarding holes in objects the tables tells only something about the topological entities, while the TORG incorporates the different classes / objects.

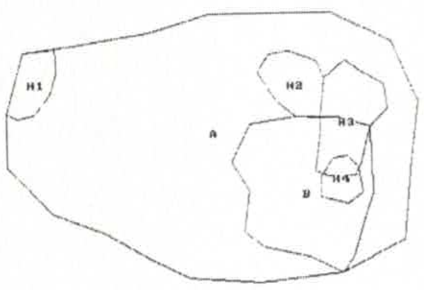

Fig 10 : Figure after Egenhofer & Franzosa (1994).

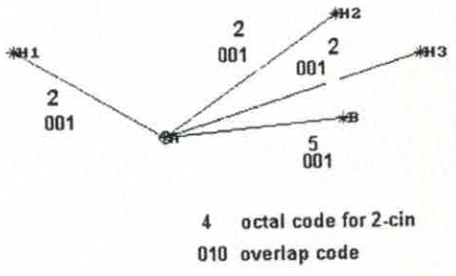

Fig 10 A

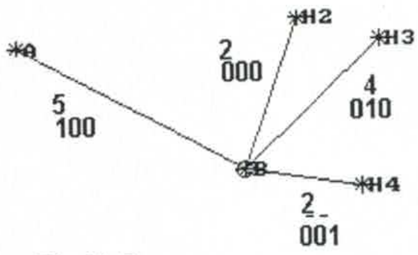

Fig 10 B

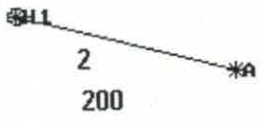

Fig 10 H1

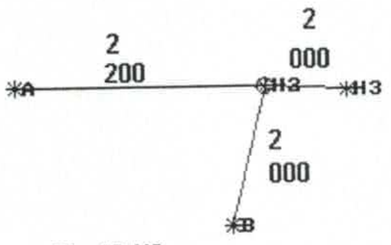

Fig 10 H2

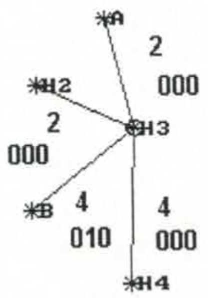

Fig 10 H3

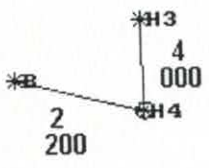

Fig 10 H4

see Egenhofer et al (1994b) fig 8 & table 5 . The fact that a piece is in between H2 and H3 makes no difference to the relationships of both H2 & H3 with A.

Two representations are homeomorphic if they are relation-homeomorphic and object-homeomorphic Egenhofer & Franzosa (1995).

Give the sets a dimension, which numbers can be used in an octal representation system. :

empty set	-1
point	0
line	1
polygon	2

In a TORG there are 2 parts :

1. the **vertices** = representing the object classes and the adjacency relationship

2. the **edges** = representing the topological relationships between vertices, i.e. a qualifier of vertices.

However if the query is formulated from the relation type on could search first the edges.

The overlap of A & B ie part of A and B occupy the same space (topologically, geometrically, abstract etc). A "cross" is a special case of overlap.

There is a continuos change from disjoint to contains and vice versa. Each of them characterised by different type of intersections (see table 2).

disjoint → meet → touch → overlap / cross → covers → contains

dim(-1) dim(0) dim(1) dim(2) dim(2) dim(2) dim(2)

Egenhofer & Franzosa (1995) focus in their paper at the number and type of boundary crossings that influence topological relationships. Using their figures , the TORG equivalents are shown below.

1 - overlap (1-ov)	2 boundary crossings C in & C out (2-bc)
2 - overlap (2-ov)	4 boundary crossings C in & C out C in & C out (4-bc)
.	
n - overlap (n-ov)	2 n boundary crossings (2n-bc)

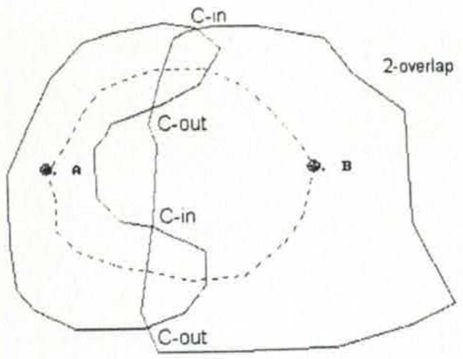

Fig 12A : Egenhofer & Franzosa (1995) focus in their paper at the number and type of boundary crossing, that infkluence topological realtionships. Using their figure, the TORG equivalents are shown.

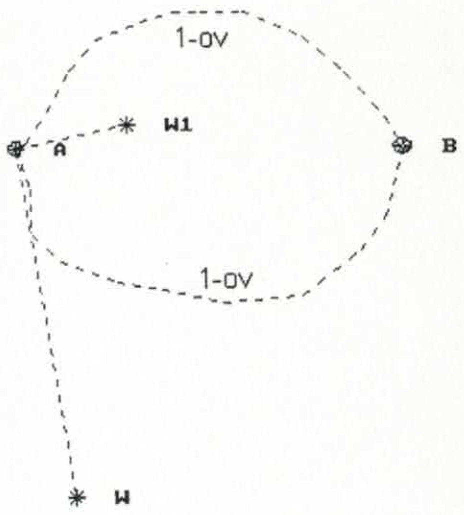

Fig 12B : TORG of Fig 12A combined A has a 2 overlap with B

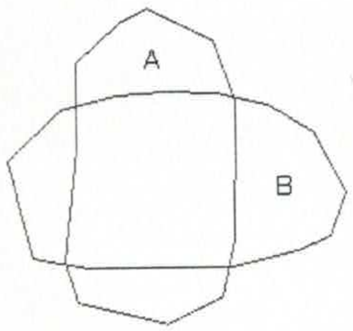

Fig 13 : A overlaps B and B overlaps A.

2 - Overlap

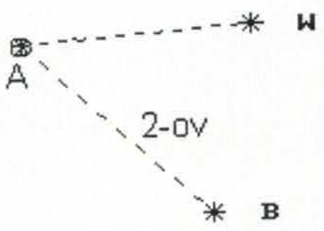

Fig 13A : TORG of fig 13

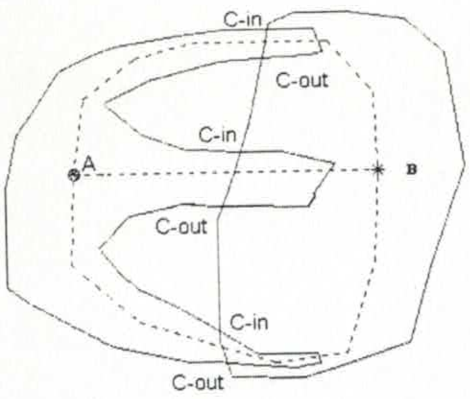

Fig 14 : 3 - Overlap. (see fig 12)

Overlap = intersection of same sort

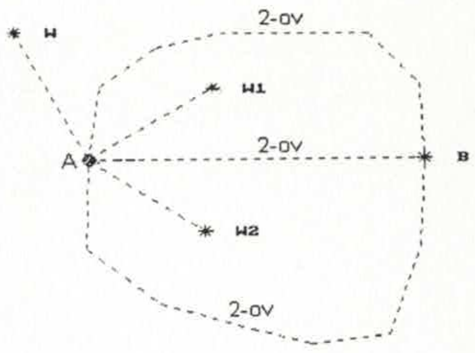

Fig 14A : TORG for fig 14

Egenhofer & Franzosa (1995) state that a component (hb : confusing terminology) of Y is the largest connected non-empty subset of Y. It is topological invariant, i.e. under any topological transformation a component transforms to a component. It can neither disappear -- turn into an empty set --- nor can it merge with another component. Egenhofer & Franzosa (1995)also state that "obviously the pure counting of the components is insufficient to determine whether two pairs of topological relations are identical or not. In order to determine whether two pairs of 2-disks (polygons) have the same topological relation, it is necessary to consider topological invariants of each individual component as well as topological properties in the relationship between an intersection and all its components".

When using TORG it is not necessary to be concerned about the detail of how the inner or outer crossings etc as explained by Egenhofer &

Franzosa (1995). With TOR the aim is first to find topological relations between objects then when successful find the physical boundaries and see if they match the requirement. I doubt such detail as Egenhofer & Franzosa (1995), propose for modelling is justified. Well at least in earth sciences, where the position of bodies are fuzzy (99 % of the time) and there are made so many underlying assumptions that the detailed modelling gives most likely a wrong answer. Also when modelling becomes complex a slightly incorrect starting point may produce totally different answers (one of the basic principles of the chaos theory)(Stewart, 1990).

One could describe the meet, touch, overlap and disjoint as intersections meaning that is the resultant of intersections are :

NAME	TYPE INTERSECT	ABBREVIATION
disjoint	-1_cell intersection	-1_cin (minussin)
touch	0_cell intersection	0_cin (zerosin)
meet	1_cell intersection	1_cin (onesin)
overlap/cross/ equal / inside/covers/contains /coveredBy	2_cell intersection	2_cin (twosin)
	3_cell intersection	3_cin (threesin)

In the graph the vertices represent the objects, their class etc, the connecting arcs typify of relationships between the objects. Naturally the object type puts a constraint on the type of intercepts that are possible. So:

CELL TYPE		TYPE INTERSECTION
two 0-cells	are possibly disjoint or	-1_cin
	equal	0_cin
two 1-cells	have possible a	-1_cin
		0_cin
		1_cin
two 2-cells		-1_cin
		0_cin
		1_cin
		2_cin
1-cell & 2-cell		-1_cin
		0_cin
		1-cin
0-cell & 1-cell		-1_cin
		0_cin
0-cell & 2-cell		-1_cin
		0_cin

0-Cells have no dimension, so they can only be disjoint or equal. From the table the following definition can be stated.

Definition : *The highest order intersection between two types of cells is equal to the lowest cell type.*

A cross and an overlap are not distinguished in TORG because from TORG viewpoint that are the same, i.e. parts of polygons of A & B are shared and parts not. Where these parts are exactly in space is not important, because that is not part of a topological relationship description.

When polygons A & B are equal means that the overlap is 100 %, which is logically true. But from an object / data modelling point of view there can be remarkable differences between overlap and equal.

Egenhofer & Chaldee (1992) point out in their paper that topological relations can change in time or due to other reasons. So if this is known this information can be recorded in the TORG and than later used when searches are or modelling is done.

The relationships between A & B can be coded for ease of manipulation. It is also common that topological relationships change and to cater for these possibilities a coding system has to set up that can handle this. Therefore I propose to use the following octal system. The object-relation-values (ORV) used are 0, 1, 2 and 4

0	disjoint (only when certain and necessary ; in general not in use in TORG because there is always the world in between two disjoint cells)
-	-1-cin
1	0_cin
2	1_cin

4	2_cin	

As argued by Egenhofer & Chaldee (1992) often topological relationships change. When this is known this can be coded into the object-relation-values as follows.

-		-1-cin
0		no topological relationship known, so normally object not recorded. May be used as flag to indicate establishing of topology still to be carried out.
1	A & B	0_cin
2		1_cin
3		0_cin or 1_cin
4		2_cin
5		0_cin or 2_cin
6		1_cin or 2_cin
7		0_cin or 1_cin or 2_cin

To indicate that the ORV also can be disjoint the minus sign can be used. If an ORV has a -1-cin it also has to have a 0-cin. ORV's make it possible to record and manipulate object relations that have more than one value. For example see fig 15.

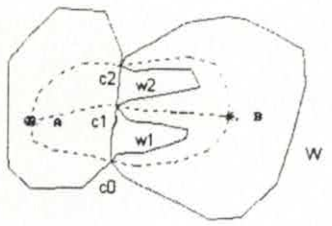

Fig 15 : After Egenhofer & Franzosa (1995).
2 unbounded boundary components c0, c2
1 bounded boundary component c1
All these examples assume that the relationships are stationary
w1 & w2 objects are of class W

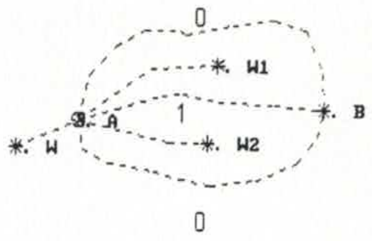

Fig 15A : TORG of fig 15

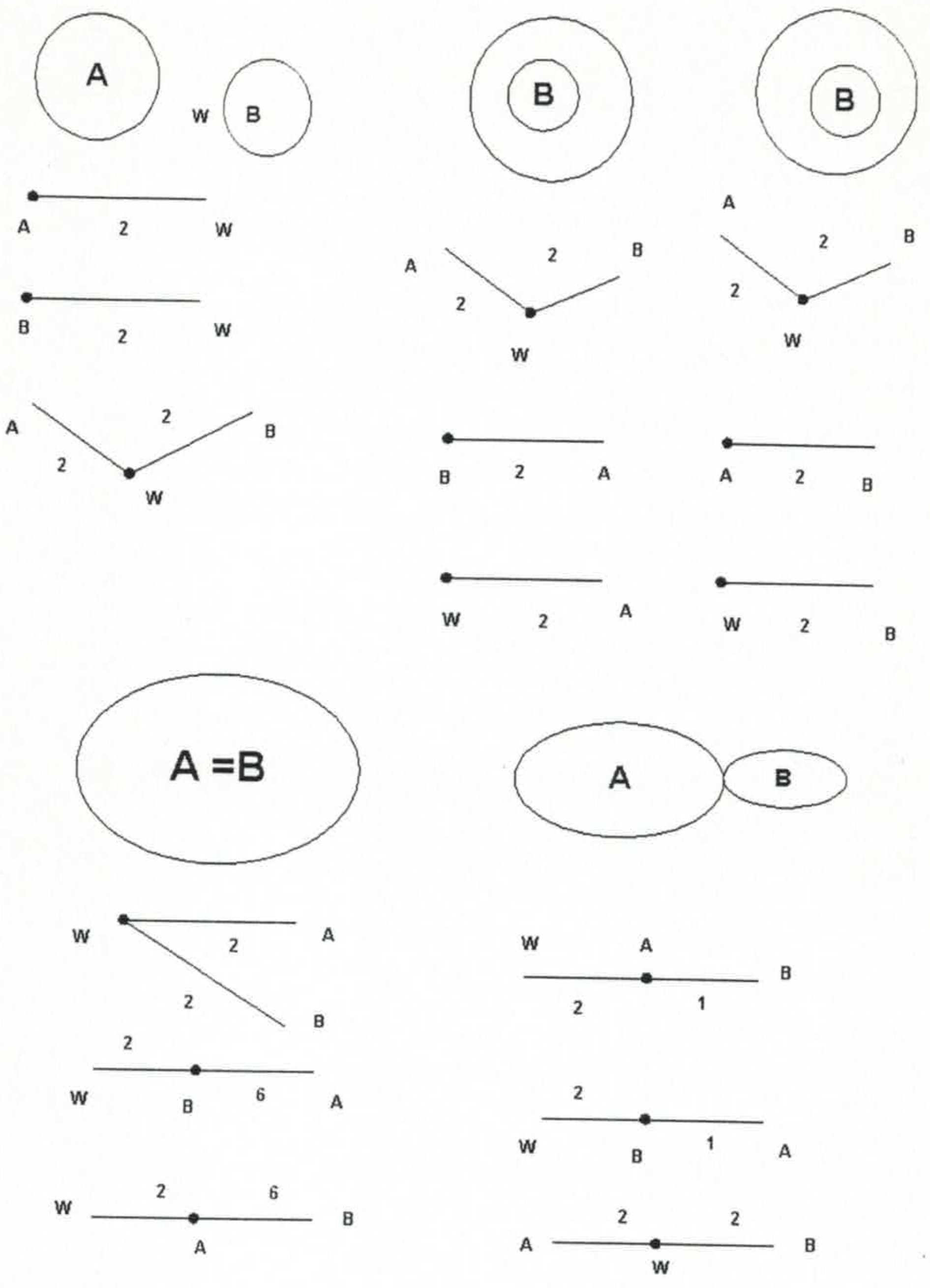

Fig 16 : TORG of 2 objects that are disjoint, touch, contain, cover or are equal

Because of its nature TORG makes no distinction between boundary and interior when it uses 2 cells.

In TORG it is not important what type of objects (0_dim, 1_dim or 2_dim) intersect, but what type of intersections are produced, i.e. a 0_cin, 1_cin or 2_cin.

Overlap, equal, covers, coveredBy, inside and contains are all forms of 2_cin. Therefore no distinction has been made in object-relation-values between them. Egenhofer & Chaldee (1992) argue and show in their Closest-Topological-Relationship-Graph there are distinct differences and relationships between these six. If this additional information about the 2_cin of two objects is available and needed it can be coded as follows. Namely a three digit code , each column can have the digits 0, 1, 2 or 3. (for explanation see also fig 33 modified from fig 23 of Egenhofer, & Chaldee (1992)) .

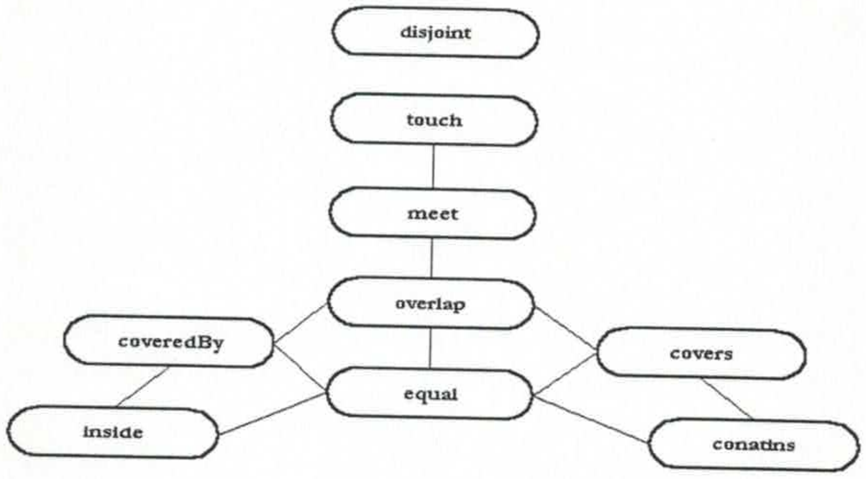

Fig 33 : Closest-Topological-Relationship-Graph after Egenhofer & Khaled(1992)

column 1	column 2	column 3
0 = none of below	0 = none of below	0 = none of below
1 = coveredBy	1 = overlap	1 = covers
2 = inside	2 = equal	2 = inside
3 = 1 & 2	3 = 1 & 2	3 = 1 & 2

So e.g. code 130 means that object A can be coveredBy B or overlays B or is equal to B. Even though digits are used this does not mean that they can be ranked or otherwise be mathematically manipulated, to state that 130 is larger than 031 is nonsense. The selection of the columns is just based on how Egenhofer & Chaldee (1992) drew their diagrams and has no influence on modelling. For example see fig 20.

Naturally if there is a transition from let say "coveredBy" to "contains", there has to at least the intermediate step of being "equal". Other options are coveredBy → inside → equal → contains, or coveredBy → overlap → covers → contains. This constraint may true in reality but in theory there is no reason to object to that an object A can have only coveredBy and contains relationships with B.

Even if a hull is convex it does not make any difference with topological relationships as Worboys & Bofakos (1993) shows in their Fig 2 (see Fig 17). When a convex polygon (A) touches with its two ends polygon (B) and the enclosed enclave is W2. The enclosed object W2 may be of the same class as the world. But enclosed object W2 may go from be existent to non-existent. This depends on the ORV that the two ends of polygon A have with B.

Fault as barrier : suit I does not necessity influence suit II , ie influence of II U on I G is less than I D even though distance is shorter. The whole suit II may be different (Fig 18) Could faults be classified as similarly as streams, ie 1st , 2nd order etc. This will be difficult. Faults may act as barriers if there has been enough displacement between the two sides.

Should an alteration Halo be treated as an object or as an attribute of a rock . Should be separate object that links with to rock objects. An alteration halo put another dimension to the existing TORG.

In modelling it is every ones pipedream to get an exact answer. But because of the whole concept of modelling , abstractification and leaving put detail the accurate positions may not be of great importance. What is of interest are conceptually correct relationships.

Database technology developed by Oracle, Sybase, Informix and others provide very efficient and fast searching of last databases. If TORG data is stored in these databases one can make use of their efficiencies.

A very important part of modelling is to get the topological relations correct so that the number of assumptions can be reduced.

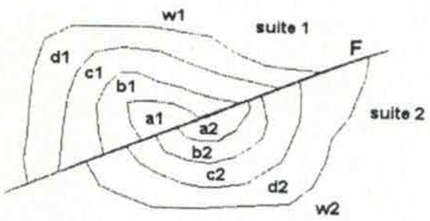

Fig 18 :
a1 = a2 , b1 = b2 , c2 = c2 , d1 = d2 are rocktypes
w1 & w2 = world
F = fault

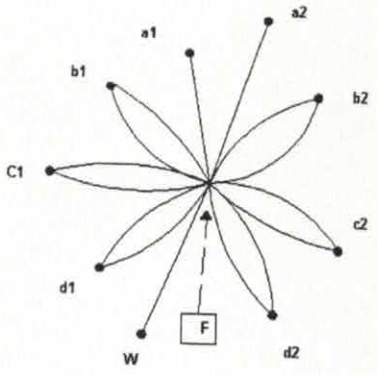

Fig 20A : F-vertex graph all edges have the value of a 1-cin

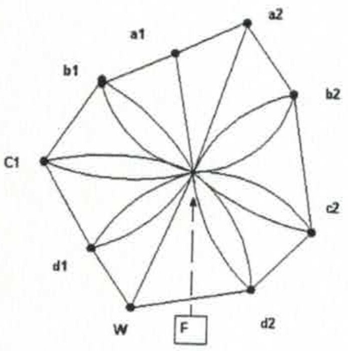

Fig 18C : A full graph of all object-relations of fig 18a

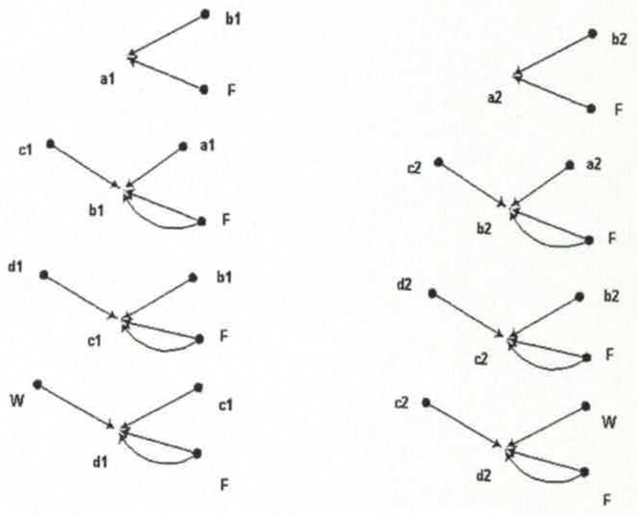

Fig 18 b : All 1-cin intersects. Individual vertex graphs

The graph is a way of visualising the relationships of an object with its neighbors. So there is no need to store ID's of the arcs that connect the vertices.

		connector_seq_no	orv
primary key	primary key	primary key	primary key

F	b1	1	3
F	b1	2	-7

point_seq_no	graph_degree	field indicating number of different object classes that are assoc with the point
This table could be seen as a view combining information from other tables.		

A TORG-base could be described as a database containing the topology and relations of adjacent objects of each object in a graph form.

In TORG you get a relationship between outface and the plane. The intersection can be

o- cin	touch in a point
1- cin	meets in a line
2- cin	meets in a surface

Object orientation becomes an important because eg faults like the Norseman Wiluna Fault in WA, is so long that if one makes a TORG of it , the degree will be so large that it becomes useless. However it could be used as combo if one is modelling on a very large scale. Needs to be devised into smaller parts.

Also when graphs of desired degree are found with the correct attributes have been found the search can be extend to if necessary to find certain topological relationships of the object next to the anchor object.

Before doing a search with TORG it has to be determined how many variables are part of the search. The types of variables are basically the object classes. The TORG for an object , eg a fault becomes more complex when relationships are recorded with more classes. New TORGs have to be created whenever new types of searches are established. The various tables with TORS are existent and established at the time of digitising and building the topology. So the TORG for a certain search is established as a "SQL"database view for future use.

Faults are very long objects. So they are from an emperica point not suitable to form the anchor of a control object.

Data search. Maybe to speed up TORG search to indicate if the ORV'S are with an object of the same or a different class , have a field in the table, with a 0 and 1 switch . Using "0' for the same class and "1" for a different class.

One can build separate tables to record all combos in the system. This would be an option for later modelling. These tables maybe kept in a "view" type database.

When using control-objects to find topological equivalents first the search is done in "tier 0 ", then "tier 1", "tier 2" etc. That means when a combo gets recorded the number of tiers that a vertex is away from the anchor object has to be recorded. See also fig 17.

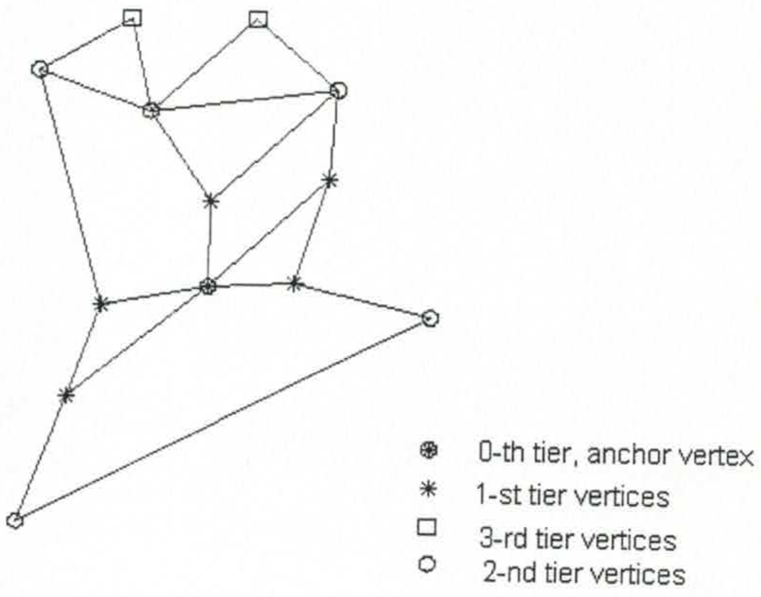

Fig 17 : Distances. In TORG because it is a graph about relationships the distance between two adjacent vertices are of unit value. So the distance between two vertices is so many edges away, an integer value.
All vertices that a re 1 edge away from a vertex are located in the 1-st tier.
All vertices 2 edges away in the 2-nd tier, and so on. The concept of tiers can be used in modelling.

The graph is a way of visualising the relationships of an object with its neighbours. So there is no need to store ID's of the arcs that connect the vertices.

In a non-object oriented GIS, which is based on layers the building of the TORG is done first for each layer., at the time of building the topology. Then when layers are intersected a TORG can be build for these combined layers. Because then TORG becomes n-dimensional it becomes impossible to display the relationships.

The topology of polygons, arcs etc is established when building the GIS dataset. Naturally it has to be established before any modelling can be done.

Fig 30

Superimposing an alteration halo over rocks two things can be done.

- Intersect the two types of objects/covers/layers and create new objects and in this way update TORG. More or less the way currently relation are established and used by a GIS.
- Just establish a 3D TORG. This a more flexible way of doing.