Geoinformatics 2004
Proc. 12th Int. Conf. on Geoinformatics – Geospatial Information Research: Bridging the Pacific and Atlantic
University of Gävle, Sweden, 7-9 June 2004

DATA UPDATE ACROSS MULTI-SCALE DATABASES

Hae-kyong Kang[1], Jung-Wook Moon[2] and Ki-Joune Li[2]
[1]Dept. of Geographic Information Science, [2]Dept. of Computer Science,
Pusan National University, 609-735, South Korea. Email : {hkang, lik}@pusan.ac.kr

Abstract

This paper discusses on the update problem of multi-scale databases when the multi-scale databases, which is several spatial databases covering the same geographic area with different scales, are derived from an original one. Although the integrity between original and derived multi-scale databases should be maintained, most of update mechanisms do not respect it since the update mechanisms have assumed that the update of source objects propagates to objects directly derived from the source. In order to maintain the integrity of multi-scale databases during updates, we must propagate updates on a source database to objects derived from both the updated source objects and other related objects. This is an important functional requirement of multi-scale database systems, which has been not supported by existing spatial database systems or geographic information systems. In this paper, we propose a set of rules and algorithms for the update propagation and show a prototype developed on ArcGIS of ESRI. Our update mechanism provides with not only the consistency between multi-scale databases but also incremental updates.

1. INTRODUCTION

Multi-scale databases are a set of spatial databases, covering the same geographic area with different spatial types and scales. Multi-scale databases can be derived automatically from existing spatial databases with constraints. In this case, the consistency between multi-scale databases should be respected and it implies that derivation constraints should be preserved based on (Egenhofer 1997). This paper focuses on the update problem of the derived multi-scale databases to maintain their consistency by preserving the derivation constraints.

Precedent researches(Chen 1989 and 1995) on the update problem of derived data have assumed that the update of source objects propagates to those which are directly derived from the source. For example, suppose that two objects o_1 and o_2 in the source database are aggregated to o_1' in a derived database as shown by in Figure 1(1). In this case, an update on o_2 causes an update only on o_1' . On the other hands, updates on a source database propagate to objects derived from both the updated source objects and other related objects on the update problem of multi-scale databases. For instance, the update on s_3 in Figure 1(2)-E propagates to not only s_1' but also s_2' , which is not derived from s_3 in Figure 1(2)-D. In addition, the insertion of new objects s_3' and s_4' , and the deletion of an existing object S_1' in Figure 1(2)-F must be performed due to the update of s_3 . These propagations have to be handled for the consistent multi-scale databases.

However, the update problem of multi-scale databases has been rarely considered in the multi-scale research fields. Thus we consider the update problem of multi-scale spatial objects across multi-scale databases in this paper.

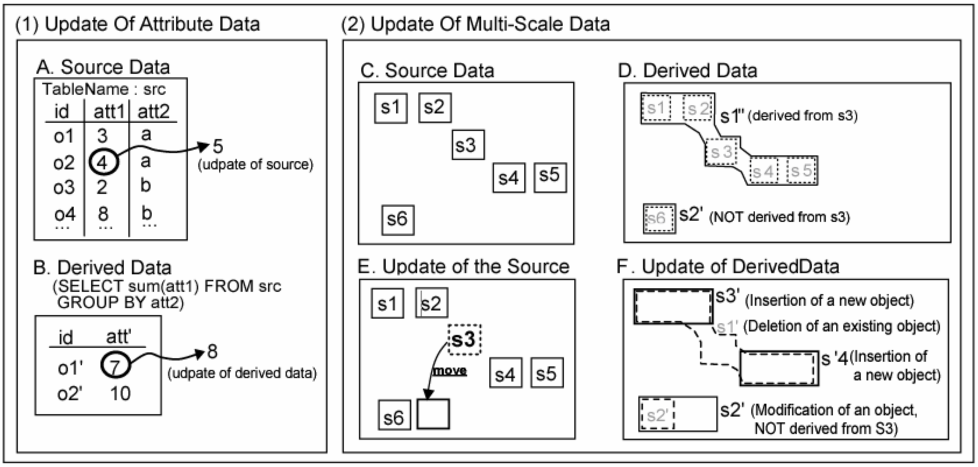

Figure 1 : Update comparison of a current traditional database and a multi-scale database

This paper is organized as follows. In section 2, previous researches are briefly investigated. We describe a multi-scale data model, rules and algorithms for the update on multi-scale databases in section 3. A link structure between source objects and derived objects is shown to describe the update rules and algorithms, the correspondence between two databases and derivation processes. We show a prototype based on this link structure in section 4, and conclude this paper in section 5.

2. RELATED WORKS

We focus on the update propagation of multi-scale databases. The update propagation may occur when source databases are updated and the multi-scale databases are derived from the source. Thus our work deals with (a) the derivation of multi-scale databases, (b) the consistency on multi-scale databases, (c) the update of spatial databases, and (d) the update propagation over derived multi-scale databases. We briefly introduce some important researches on this theme (Figure 2).

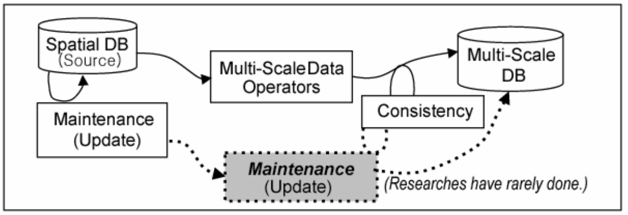

Figure 2 : Researches in multi-scale databases

- **Derivation of Multi-scale Database** : The derivation of a new small-scale database from an existing large-scale database can be realized by cartographic generalization, which is motivated to improve its visual quality by removing conflicts of spatial data. In the context of the cartographic generalization, many researches have proposed a set of operators, such as simplification, and smoothing (McMaster 1992 and Muller 1995). In (Davis 1999) these operators are categorized into map generalization, geometric and spatial analysis operators. When more sophisticated results are requested, a derivation process of new scale databases needs not only spatial distortion mechanism but also semantic abstraction such as merging objects into one (Richardson 1994, Rigaux 1994, and Peng 1996).

- **Consistency on Multi-scale Database** : Consistency at the database level refers to the lack of any logical contradictions within a data model of reality (Egenhofer 1997). We can eliminate logical contradictions by defining constraints, and a database is considered as inconsistent if objects of the database do not respect the constraints described by data model. From this point of view, the consistency of multi-scale databases implies that (i) derivation constraints are preserved, (ii) derived relations are correct. By (Egenhofer 1993) an evaluation method of inconsistencies among multiple representations is proposed. (Tryfona, 1994) suggested a manner to reason proper relations to multi-scale database. (Egenhofer 1994 and Ubeda 1997) proposed to assess the consistency of topological relations.
- **Spatial Data Update** : (Belussi 2000) proposed an integrity driven system to guarantee the topological integrity of spatial databases. The system prevents from destroying the integrity of spatial databases during updates. In (Peled 1996 and 1998) the update problem of spatial databases is considered in terms of update of topographical maps.
- **Update of Derived Multi-scale Data** : (Kidner 1994) introduced a prototype, which maintains multi-scale databases. The update and retrieval of multi-scale databases are performed by deductive processing, employing rules, reasoned from explicit relations and constraints in source databases. This prototype was improved by (Jones 1996).

3. THE UPDATE OF MULTI-SCALE DATABASE

3.1 A Multi-Scale Data Model

We propose a multi-scale data model (Figure 3) consisting of elements to represent source objects, derived objects, and the derivation information of multi-scale databases. The symbols and their meaning, which will be used in the rest of this paper, are given in Table 1.

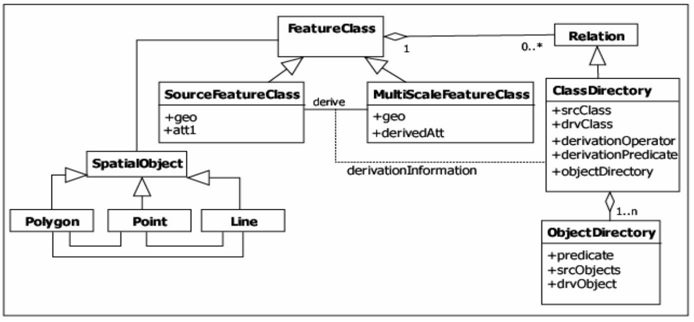

Figure 3 : A multi-scale data model.

Details about the elements of the multi-scale data model are as follows. **MultiScaleFeatureClass(** FC_{DRV} **)** is derived from **SourceFeatureClass(** FC_{SRC} **)** by a semantic abstraction. The semantic abstraction in a database represents a group of objects with common properties. The properties include adjacency, proximity, or same attribute values in spatial databases. In terms of the semantic abstraction, we consider only two derivation operators in this paper; *GeoAggregation* merges objects within a proximity distance and *Classification* combines objects with same attribute values.

Table 1. Symbols and Meanings

Symbol	Meaning	Description
FC	Feature Class	a set of features; FC_{SRC}: a set of source features; FC_{DRV}: a set of derived features from FC_{SRC} the constraints
f	A set of features $\in FC$	f_i : i_{th} feature, $f.geo$: a spatial shape of a feature
R_{CDRY}	Class Directory Class	a set of relations between FC_{SRC} and FC_{DRV}
R_{ODRY}	Object Directory Class	a set of relations between f_{SRC} and f_{DRV}
att	Attribute	$FC.att$: an attribute of FC; $f.att_j$: j_{th} attribute of a feature f
$dom(att)$	Domain of att	the ranges of attribute values of att
v	An attribute value	$f.a = v$: v is an attribute value of $f.a$

ClassDirectory(R_{CDRY} **)** represents the relationships between FC_{SRC} and FC_{DRV}. The *derivationPredicate* represents constraints used in derivation processes. The types of constrains are (i) attribute-specified ($FC_{SRC}.att$), (ii) attribute-value-specified ($FC_{SRC}.att = v$), and (iii) value-specified(v) by user. In the next section, we describe a set of rules for each type of constraints because the derivation process of FC_{DRV} is determined due to the constraints in different ways.

3.2 Update Propagation Rule

We consider four types of updates; insertion, deletion, geometric change and attribute-value change of objects. The update rules deal with the update propagation of FC_{SRC} into FC_{DRV} and we propose update rules for each update type. In Table 2, Rule1 (R1) handles the update of FC_{DRV} derived from FC_{SRC} with the attribute-specified constrain when f_{ins} is inserted into FC_{SRC}. Each update rule is specified by an algorithm given in appendix 1.

Table 2. Rules and Algorithms for Update Propagation.

Constraint	Update Type	Rule	Algorithm	Constraint	Update Type	Rule	Algorithm
Attribute-Specified	Insert	R1	A1	Attribute-Value Specified	Insert	R5	A1
	Delete	R2	A2		Delete	R6	A2
	ChangeGeometry	R3	A3		ChangeGeometry	R7	A3
	ChangeAttribute	R4	A1, A2		ChangeAttribute	R8	A1, A2
Value-Specified	Insert	R9	A4	Value-Specified	ChangeGeometry	R11	A4, A5
	Delete	R10	A5				

- **Case 1 : Attribute Specified as a derivation constraint**

Let FC_{DRV} be a derived class of FC_{SRC} with a constraint, $FC_{SRC}.att$.

[Rule1] New f_{ins} is inserted into FC_{SRC}. The f_{ins} causes f_{drv} of which predicate is $att = v_1$ to be merged with f_{ins} , if $f_{ins}.att = v1$ is in dom($f_{ins}.att$). If $f_{ins}.att = v$ is not in dom($f_{ins}.att$) then f_{ins} is to be inserted into FC_{DRV} .

[Rule2] f_{del} is deleted from FC_{SRC}. Then, f_{drv} of which predicate is $att = v$ is modified.

[Rule3] f_{chgeo} is a subset of f_{src} in FC_{SRC} which geometry is changed. The f_{chgeo} causes f_{drv} from f_{chgeo} to be modified.

[Rule4] f_{chatt} is a subset of f_{src} , where $f_{src}.att = v_{original}$ is changed to v_{change}. The f_{chatt} causes $f_{drv_{originVal}}$ of which predicate is $att = v_{original}$ to be modified. If $f_{chatt}.att = v_{change}$ is in dom($f_{src}.att$), then $f_{drv_{chVal}}$ of which predicate is $att = v_{change}$, is merged with f_{chatt} . If $f_{chatt}.att = v_{change}$ is not in dom($f_{src}.att$) then f_{chatt} is to be inserted into FC_{DRV} .

• Case 2 : Attribute-Value Specified as a derivation constraint

Let FC_{DRV} be a derived class of FC_{SRC} by $FC_{SRC}.att = v$, where v is a subset of dom($FC_{SRC}.att$).

[Rule5] New f_{ins} of which attribute satisfies v is inserted into FC_{SRC}. It causes f_{ins} to be inserted into FC_{DRV} (that is, f_{drv} which predicate is $att = v$ to be merged with f_{ins}).

[Rule6] f_{del} is deleted from FC_{SRC}. Then, f_{drv} of which predicate is $att = v1$, is modified.

[Rule7] Change of geometry of f_{chgeo} in FC_{SRC} causes f_{drv} derived from f_{chgeo} to be modified.

[Rule8] f_{chatt} is a subset of f_{src} which $f_{src}.att = v_{original}$ is changed to v_{change}. It causes $f_{drv_{originVal}}$ of which predicate is $att = v_{original}$ to be modified as well as $f_{drv_{chVal}}$ of which predicate is $att = v_{change}$ to be modified.

• Case 3 : A value specified by a user as a derivation constraint

Let FC_{DRV} be a derived class from FC_{SRC} by v.

[Rule9] New f_{ins} is inserted into FC_{SRC}. If $f_{ins} \cup f_{drv}$ satisfies with v, then new $f_{drv\,new}$ derived from $f_{ins} \cup f_{drv}$ is inserted into FC_{DRV}. f_{drv} is deleted from FC_{DRV}.

[Rule10] Deletion of f_{del} from FC_{SRC} causes f_{drv} derived from f_{del} to be modified. New $f_{drv\,new}$ derived from $f_{src} \cup f_{del}$ with v, where f_{src} is used in derivation of f_{drv}, is inserted into FC_{DRV}. Then, f_{drv} is deleted from FC_{DRV}.

[Rule11] Change of f_{ch} in FC_{SRC} causes f_{drv_i} in FC_{DRV} derived from f_{ch} to be modified as well as f_{drv_j} in FC_{DRV} which is derived from f_{ch}. That is, $f_{drv\,new1}$, derived from $f_{src} - f_{ch}$ with v where f_{src} is the source f_{drv}, is inserted into FC_{DRV}. Then f_{drv_i} is deleted from FC_{DRV}. New $f_{drv\,new2}$ derived from $f_{ch} \cup f_{drv_j}$ is inserted into FC_{DRV}, where $f_{ch} \cup f_{drv_j}$ satisfies with the v. f_{drv_j} is deleted from FC_{DRV}.

3.3 Simple Example

This example shows that R11 is applied to a multi-scale database. Figure 4 shows a multi-scale data model at instance level.

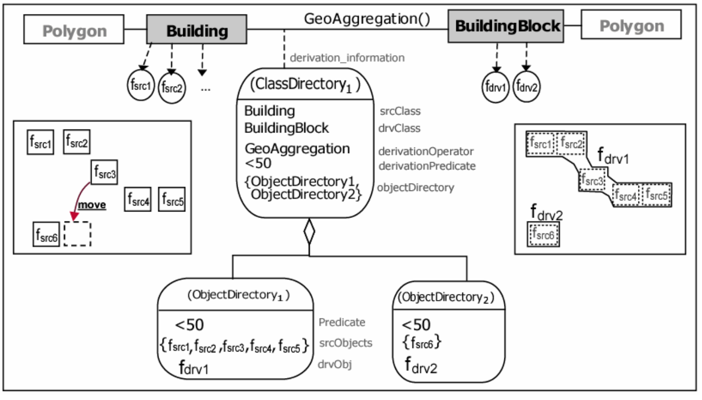

Figure 4 : The example of Rule11

Given Data Set : A class *BuildingBlock* is derived from a source class *Building* by *GeoAggregation* with a predicate “distance < 50”.
Update of Source : f_{src3} is moved into the next of f_{src6}.
Update Propagation: R11 is employed to update *BuildingBlock* because a constraint type of the derivationPredicate “distance<50”, is a value-specified constraint and the update type is *changeGeometry*. R11 is performed by algorithm 4 and 5. For example, line 9 of algorithm 5 calls GeoAggretaion(). As a result, two new objects, f_{drv3} and f_{drv4}, are inserted into FC_{DRV} and f_{drv1} is deleted from FC_{DRV} in the right figure.

```
Algorithm 5
 1: ListUpdateObj := {fsrc3}
 2: while( ListUpdateObj ≠ φ )
 3: fdel := fsrc3
 4: fdrv := fdrv1
 5:
 6: ListSrcObjs := { fsrc1 , fsrc2 , fsrc3 , fsrc4 , fsrc5 }
 7: ListObjs := { fsrc1, fsrc2, fsrc4, fsrc5 }
 8: ListUpdateObj := { φ }
 9: fdrv3 := { fsrc1, fsrc2} and fdrv4 := { fsrc4, fsrc5 }
    //new objects are created.
10: delete fdrv1 from FCDRV
11: End while
```

4. A PROTOTYPE

We have been developing a prototype of multi-scale database manager with a spatial library of ESRI (ArcObjects), which is a component of ArcGIS (an ESRI product). Main components of the prototype are as follows (Figure5).

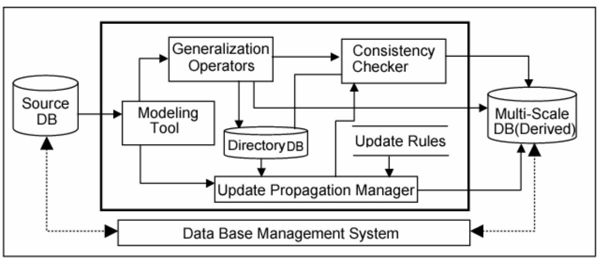

Figure 5. The Architecture of the Multi-Scale Data Manager.

Modeling Tool allows users to customize the derivation process through a graphical interaction. It is possible to specify source classes to be loaded, new classes to be derived, generalization operators and predicates to be used in the process. **Generalization Operators** supports six operators (Kang 2001). The operators generate multi-scale databases and their derivation information. **Directory DB** gives corresponding information between source databases and multi-scale databases. *ClassDirectory* and *ObjectDirectory* classes of the diagram in Figure 3 are stored in the directory DB. **Consistency Checker** assesses the multi-scale databases optionally whether relations are correct or not. **Update Rule** supports update propagation rules, proposed in Section 3. These rules are to be employed by the update propagation manager. **Update Propagation Manager** propagates the update of sources to its derived multi-scale databases by referencing the directory DB and employing update propagation rules.

5. CONCLUSION

In this paper, we considered the update problem of multi-scale databases when the multi-scale databases are derived from an original one. In order to maintain the integrity between original and derived multi-scale databases during updates, we must propagate updates on a source database to objects derived from both the updated source objects and other related objects. Nevertheless, most of update mechanisms do not respect it since the update mechanisms have assumed that the update of source objects propagates to objects directly. Thus we proposed update rules and algorithms for handling update propagations on multi-scale databases. We have been developing a prototype with ArcObjects of ESRI product. Our update mechanism contributes to the consistency and incremental update of a multi-scale database.

ACKNOWLEDGEMENT

This work was supported by the program for cultivating graduated students in regional strategic industry of Korea Industrial Technology Fundation.

REFERENCES

Belussi A., Negri M., and Pelagatti G., 2000: An Integrity Constraints Driven System for Updating Spatial Databases. *ACM Int. Workshop on Advances in Geographic Information Systems*, 121-128.

Chen I. and McLeod D., 1989: Derived Data Update in Semantic Database. *Proc. of the 5th International Conference on Very Large Data Bases*, 225-235.

Chen I., Hull R. and Mcleod D., 1995: An Execution Model for Limited Ambiguity Rules and Its Application to Derived Data Update. *ACM Transactions on Database Systems*, 20(4), 365-413.

Davis Jr. C. A. and Laender A. J. F., 1999: Multiple Representations in GIS: Materialization Through Map Generalization, Geometric, and Spatial analysis Operations. *ACM Int. Workshop on Advances in Geographic Information Systems*, 60-65.

Egenhofer M. and Sharma J., 1993: Assessing the Consistency of Complete and Incomplete Topological Information. *Geographical Systems 1(1)*, 47-68.

Egenhofer M., Clementini E. and Felice P., 1994: Evaluating Inconsistencies Among Multiple Representation. *Int. Symposium on Spatial Data Handling*, 901-920.

Egenhofer M. J., 1997: Consistency Revisited, *GeoInformatica*, 1(4), 323-325.

Davis Jr. C. A. and Laender A. H. F., 1999: Multiple Representation in GIS : Materialization Through Map Generalization, Geometric, and Spatial Analysis Operations. *ACM Int. Workshop on Advances in Geographic Information Systems*, 60-65.

Jones C. B., Kidner D. B., Luo L. Q., Bundy G. Ll. and Ware J. M., 1996: Database Design for a Multi-scale Spatial Information System, *Int. Journal of Geographical Information Systems*, 10(8), 901-920.

Kang H., Do S., and Li K., 2001: Model-Oriented Generalization Rules. *Proc.(in CD) ESRI User Conference*, San Diego, USA, July.

Kidner D. B. and Jones C. B., 1994: A Deductive Object-Oriented GIS for Handling Multiple Representations. *Int. Symposium on Spatial Data Handling*, 882-900.

McMaster R. and Shea K., 1992: Generalization in Digital Cartography. *The Association of American Geographers*, 1-134.

Muller J. C., Lagrange J. P. and Weibel R., 1995: GIS and Generalization:Methodology and Practice, *Taylor & Francis Inc.*, 1-252.

Peng W., and Tempfli K., 1996: An Object-Oriented Design for Automated Database Generalization". *Int. Symposium on Spatial Data Handling*, 199-213.

Peled A., 1996: Spatial Database Update-The key to effective automation. *Int. Archives of Photogrammetry and remote sensing*, XXXI(B4), 955-961.

Peled A., 1998: Toward Automatic Updating of the Israel National GIS Phase II. *Symposium on GIS Between Visions and Applications*, 32(4). 467-472.

Richardson D., 1994: Generalization of Spatial and Thematic data using inheritance and classification hierarchies. *Int. Symposium on Spatial Data Handling*, 957-972.

Rigaux P., and Scholl M., 1994: Multiple Representation Modeling and Querying. Geographic Information Systems: *Int. Workshop on Advanced Information Systems*, 59-69.

Tryfona N. and Egenhofer M. J., 1997: Consistency among Parts and Aggregates: A Computational Model. *Transactions in GIS*, 1(3), 189-206.

Ubeda Th., Egenhofer M. J., 1997: Topological Error Correcting in GIS. *In: Advances in Spatial Databases(SSD), Int. Symposium on Large Spatial Databases*, 283-297.

Appendix 1. Update Propagation Algorithms

Algorithm 1 : Insertion
Input : $FC_{SRC}, FC_{DRV}, R_{CDRY}, R_{ODRY}$
Output : FC_{DRV} , updated
Method :
Let $ListUpdateObj$ be a list of inserted objects(f_{ins}) into FC_{SRC}
$constraint\,C \leftarrow R_{CDRY}.derivationPredicate$
While ($ListUpdateObj \neq \phi$) **Do**
  Get f_{ins} from $ListUpdateObj$
  $constraint\,Obj \leftarrow R_{ODRY}.derivationPredicate$ of f_{ins}
  If (v of $f_{ins}.att \in dom(FC_{SRC}.att)$), where $att = constraint\,C$
    Get f_{drv} which $R_{ODRY}.predicate = v$
    $FC_{DRV} \leftarrow$ merge(f_{drv} , f_{ins})
  Else if (v of $f_{ins}.att \notin dom(FC_{SRC}.att)$)
    $FC_{DRV} \leftarrow$ Insert(f_{ins})
  End if
End while

Algorithm 2 : Deletion
Input : $FC_{SRC}, FC_{DRV}, R_{CDRY}, R_{ODRY}$
Output : FC_{DRV} , updated
Method :
Let $ListUpdateObj$ be a list of deleted objects(f_{del}) from FC_{SRC}
Let $ListSrcObjs$ and $ListObjs$ be a list of objects each.
Let f_{drv} be a subset of FC_{DRV} , where f_{drv} is derived from f_{del}
While ($ListUpdateObj \neq \phi$) **Do**
  Get f_{del} from $ListUpdateObj$
  $ListSrcObjs \leftarrow R_{ODRY}.srcObjects$ of f_{drv}
  $ListObjs \leftarrow \{ListSrcObjs\} - \{ListUpdateObj\}$
  $ListUpdateObj \leftarrow \{ListUpdateObj\} - \{ListSrcObjs\}$
  While ($ListObjs \neq \phi$) **Do**
    $FC_{DRV} \leftarrow$ union($ListObjs$)
  End while
End while

Algorithm 3 : Change Geometry
Input : $FC_{SRC}, FC_{DRV}, R_{CDRY}, R_{ODRY}$
Output : FC_{DRV} , updated
Method :
Let $ListUpdateObj$ be a list of changed objects(f_{chgeo})
Let $ListSrcObjs$ be a list of objects.
Let f_{drv} derived from f_{chgeo} be a subset of FC_{DRV}
While ($ListUpdateObj \neq \phi$) **Do**
  Get f_{chgeo} from $ListUpdateObj$
  $ListSrcObjs \leftarrow R_{ODRY}.srcObjects$ of f_{drv}
  While ($ListSrcObjs \neq \phi$) **Do**
    $FC_{DRV} \leftarrow$ union($ListSrcObjs$)
  End while
End while

Algorithm 4 : Insertion
Input : $FC_{SRC}, FC_{DRV}, R_{CDRY}, R_{ODRY}$
Output : FC_{DRV} , updated
Method :
Let $ListUpdateObj$ be a list of objects(f_{chgeo})
 in FC_{SRC} which geometry is changed.
Let f_{drv} be all objects of FC_{DRV} .
$constraint \leftarrow R_{CDRY}.derivationPredicate$
While ($ListUpdateObj \neq \phi$) **Do**
  Get f_{del} from $ListUpdateObj$
  $FC_{DRV} \leftarrow$ union($f_{chgeo} \cup f_{drv}$),
   where ($f_{chgeo} \cup f_{drv}$) satisfies with $constraint$
End while

Algorithm 5 : Deletion
Input : $FC_{SRC}, FC_{DRV}, R_{CDRY}, R_{ODRY}$
Output : FC_{DRV} , updated
Method :
Let $ListUpdateObj$ be a list of objects(f_{del}) deleted from FC_{SRC}
Let $ListSrcObjs$ and $ListObjs$ be a list of objects each.
Let f_{drv} be a subset of FC_{DRV} ,where f_{drv} is derived from f_{del}
While ($ListUpdateObj \neq \phi$) **Do**
  Get f_{del} from $ListUpdateObj$
  $ListSrcObjs \leftarrow R_{ODRY}.srcObjects$ of f_{drv}
  $ListObjs \leftarrow \{ListSrcObjs\} - \{ListUpdateObj\}$
  $ListUpdateObj \leftarrow \{ListUpdateObj\} - \{ListSrcObjs\}$
  call GeoAggretaion($ListObjs$ **)**
  delete f_{drv}
End while