CHAPTER FIVE

Applying the STDM: public boundaries evolution

> The act of drawing a diagram does not constitute analysis or design. A diagram simply captures a statement of a system's behaviour (for analysis), or the vision and details of an architecture (for design). If you follow the work of any engineer—software, civil, mechanical, chemical, architectural, or whatever—you will soon realize that the one and only place that a system is conceived is in the mind of the designer. As this design unfolds over time, it is often captured on such high-tech media as white boards, napkins, and the backs of envelopes.
>
> G.Booch, *Object-Oriented Analysis and Design with Applications*

This chapter provides a comprehensive set of diagrams to illustrate the significant modelling decisions when applying the STDM to a knowledge domain. And the knowledge domain has been chosen from historical geograph—maintenance of the political boundary record. It is illustrated by a process diagram that describes a set of processes for four different scenarios and their allocation to processors in the physical view of the STDM. Class diagrams describe the relationships among states and events, as well as the operations and properties associated with them. They are also used to describe the semantic dependency between classes and the ability to walk through the model from one scenario to another. Finally, interaction diagrams illustrate the execution of each scenario and can be used to visualise the process involved in 'making space-time paths' within the STDM. ObjectMaker release 2.1 by Mark V Systems Limited has been used as the CASE tool for supporting the proposed diagrams of the STDM. The key to the symbols is given in Appendix A.

5.1 PUBLIC BOUNDARY RECORD MAINTENANCE

Public boundaries represent the line of physical contact between administrative units in Great Britain. The arrangement of public boundaries forms an irregular tessellation of polygons that represent the whole hierarchy of local government and European constituency areas. Approximately 3000 changes occur to public boundaries in England every year, increasing the volume of data at the rate of 5 to 10 megabytes a year (Rackham, 1992). Records of the history of each public boundary can be found in documents and published material such as parliamentary acts and orders retained by the Ordnance Survey. These contain legal information defining the line on the basis of which each public boundary is attached to a physical feature on the ground.

Maps portraying the public boundaries when they have been delimited and demarcated also contain important historical information for identifying the physical features on the ground. Fieldwork records kept by the Ordnance Survey are another valuable source of historical information about the evolution of public boundaries. One example is the register of discrepancies between the true (legal) description of a public boundary and its demarcation on the ground. Another example is the collection of perambulation cards noting the occurrence of some change in the location of a physical feature on the ground from its previous public boundary demarcation.

5.1.1 The knowledge domain

Public boundaries are linear objects that experience a succession of changes in their positions during their lifespans. The history of each public boundary is unique and shows the geographical significance of a public boundary over the development of landscapes, socio-economic policies and historical conflicts. Attempts to devise historical processes involved in political boundary evolution have been successful in identifying a set of reliable events by which human actions can be connected with the evolution of the majority of public boundaries. De Lapradelle (1928) identified preparation, decision and execution as the three main historical events which most political boundaries go through:

> The [event]…of preparation precedes true delimitation. The problem of the boundary's location is debated first at the political level then at the technical level. The question is, in general, of determining, without complete territorial debate, the principal alignment which the boundary will follow…. The decision involves the description of the boundary or delimitation…. The execution consists of marking on the ground the boundary which has been described and adopted, an operation which carries the name demarcation. (de Lapradelle, 1928, p. 73)

In adopting these first delineations for historical events in political boundary evolution, Jones (1945) extended them to allocation, delimitation, demarcation and administration. The administration event would deal with the maintenance of the physical features which have been allocated to be a public boundary. More recently, Prescott (1987) pointed out the following three main processes in political boundary evolution:

- *evolution in definition:* The historical events suggested by Lapradelle and Jones are proposed as boundary-making events; 'allocation refers to the political decision on the distribution of territory; delimitation involves the selection of a specific boundary site; demarcation concerns the marking of the boundary on the ground; and administration relates to the provisions for supervising the maintenance of the boundary' (*ibid.,* p. 69).

- *evolution in position:* This means 'how long the boundary has occupied particular sites' (*ibid.,* p. 77).
- *evolution in the state functions:* Evolution of state functions applied at the boundary means 'the effectiveness with which the boundary marks the limits of sovereignty' (*ibid.,* p. 80).

Analysing the evolutionary aspects of political boundaries represents the study of human activities that have been relevant to the location of a particular boundary. The STDM considered here was formulated and constructed using evolution in definition. This process is characterised by the changes related to different states acquired by every individual boundary after going through its historical events (allocation, delimitation, demarcation, administration).

Rackham (1987, pp. 32–3) identified from Booth (1980) six different states which most public boundaries can go through:

- *Draft:* proposed but not yet confirmed by an act or order.
- *Proposed works:* referred to in an order related to a physical feature which has not yet been constructed (e.g. a new road).
- *New:* made in an act or order but not yet mered.
- *Disputed:* mered but not certified because of some disagreement.
- *Old:* ascertained on the ground, certified by the relevant authorities and therefore fixed in alignment.
- *Obsolete:* old boundary no longer used to demarcate administrative areas (obsolete boundary may revert to old boundary if it is reused at a later date to demarcate an administrative area).

The main challenge in designing a spatio-temporal data model for this knowledge domain is to handle systematically the changes related to different *states* (draft, proposed work, new, disputed, old, obsolete) acquired by every public boundary after going through its historical *events* (allocation, delimitation, demarcation, administration). This can be achieved by exploring the synergy of historical events and states of public boundaries through the creation of space-time paths for each public boundary.

5.1.2 The space-time path

Each public boundary has its own space-time path that represents its lifespan. As mentioned in Chapter 4, the STDM is deemed to deal with the explanatory task in the spatio-temporal reasoning domain. This involves producing a description of the

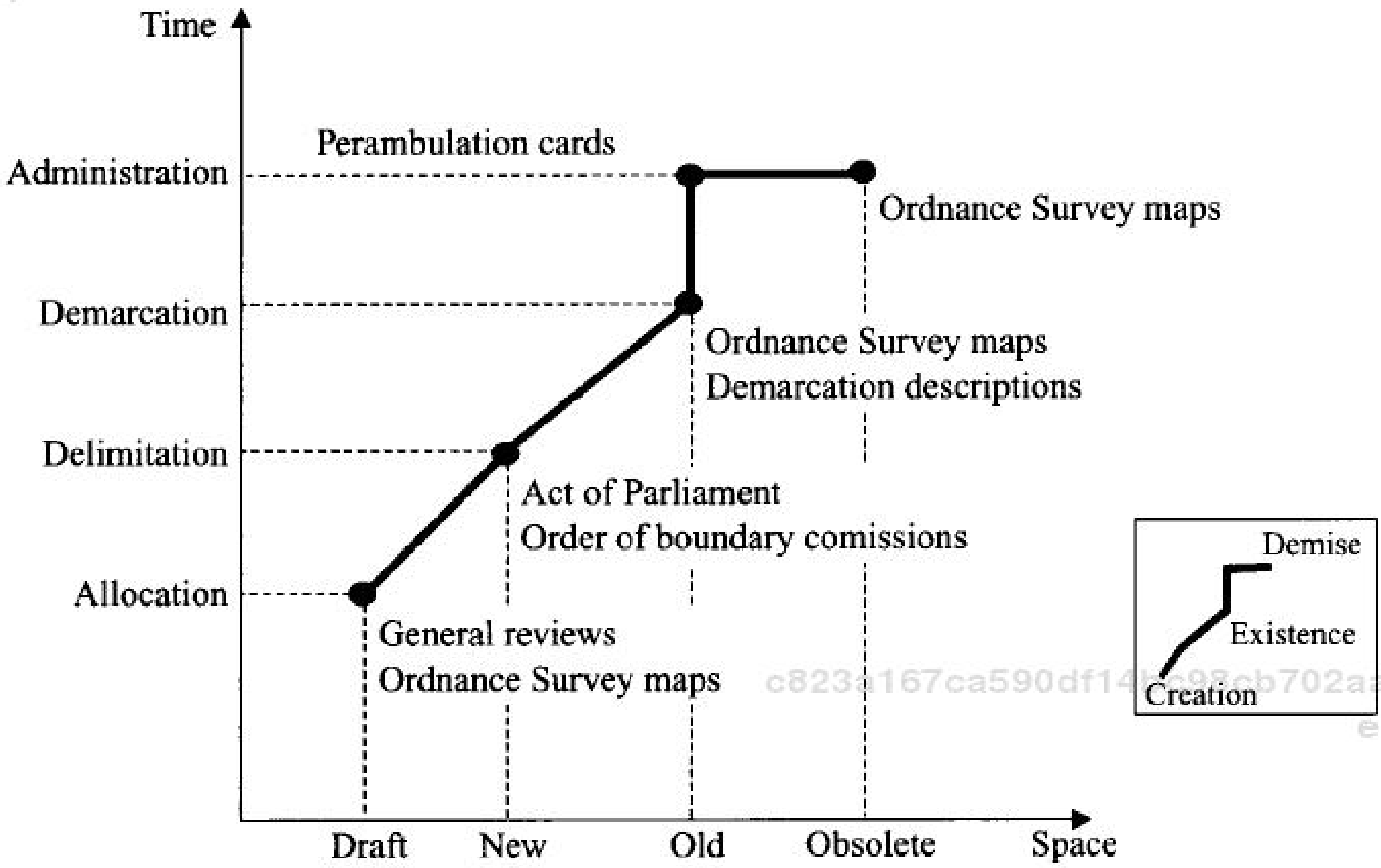

Figure 5.1 An example of a possible space-time path for a public boundary

evolution of public boundaries at an earlier time that accounts for the public boundaries being the way they are at a later time. Considering the evolution of public boundaries, this explanatory task implies that the space-time path has a longitudinal configuration. There is no possibility of having a branching configuration for a space-time path of a public boundary over its past. The creation, existence and demise circumstances are primary components of a space-time path.

Creation

Creation represents the space-time origin of a public boundary lifespan that will evolve towards the future or past from its origin (Figure 5.1). At the creation circumstance, an *allocation* event takes place in selecting a ground feature to be a future public boundary. Allocation events are responsible for representing the dimension valid time within the STDM. Time is represented as a nominal value that indicates the actual date when the allocation took place. One example of this case is the general reviews of public boundaries carried out by parliamentary boundary commissions and local government commissions (Coombes *et al.,* 1993). These reviews are needed to investigate positions or arrangements of new public boundaries to ensure a uniform representation of the electoral population for every constituency (ward, electoral division, district, region and parliamentary constituency) in Great Britain. The parliamentary boundary commissions have carried out three general reviews since 1944. If the creation circumstance is set on the space-time path for occurring in 1994, the space-time path will deal with the task of modelling the boundary changes as investigated by the commissioners since 1944. Once the creation circumstance is dated, it cannot subsequently be modified since this would cause integrity problems within the STDM.

There exists a specific spatial relationship between a ground feature and a public boundary that has to be selected from a set of possibilities depending on the kind of ground feature being utilised. For example, ground features can be paths, ponds, rivers, railways, fences, roads and hedges. Therefore, some possible spatial relationships would be 'centre of' the road, 'face of' the fence, 'root of' the hedge, '1.00m from' the railway or '1.83m from' the path. Public boundaries have to be related to ground features, but, some landscapes do not have suitable ground features. In this case, a straight line between two mereing points on the ground determines a public boundary. Otherwise, an engineering work can be carried out to build the necessary ground feature to be a part of the public boundary. All boundary lines representing these spatial relationships are portrayed on Ordnance Survey maps at 1:1250, 1:2500 and 1:10000 scales.

Existence

Existence encapsulates the space-time path over which the historical *events* (allocation, delimitation, demarcation, administration) of a public boundary evolve over space and time. The existence of a boundary is also constituted by the occurrence of changes. These changes result from the effects of human activity. They are the *states* through which the public boundaries evolve. For example, a draft state is assigned to a public boundary which has not yet been confirmed by an act of Parliament. Once this act is promulgated, the boundary modifies its *draft* state to a different state named as *new*. The historical event to occur is the *delimitation* which generates the *new* boundary state (see Figure 5.1).

The operative date and the effective date represent the episode element of the space-time path for the delimitation event. An operative date is the actual date when a public boundary is issued by an act or order, generally 16 May, or the first Thursday in May for Scotland. An effective date assigns the date when an act or order has become effectual after the General Election following the operative date (Rackham, 1987). A coupling constraint for each public boundary is essential to guarantee that the date in the allocation event must be prior to both operative and effective dates of the delimitation event (Chapter 2 describes coupling, capability and authority constraints).

All public boundaries in Great Britain have been delimited by the issuing of an act of Parliament or an order of the boundary commission. Extensive archives containing maps at 1:10000 scale and the statutory documents, such as acts and orders, are held by the Ordnance Survey in order to preserve the legal records of the public boundaries. Thus, the new boundary state plays an important role in the STDM, which verifies the fact that each public boundary cannot exist without having a new boundary state (coupling constraint).

The significant characteristic encountered in the spatial relationship between draft and new states is spatial generalisation, which demands procedures for line simplification (capability constraint). The Ordnance Survey uses different scales for portraying a public boundary having draft and new states. A new boundary is usually portrayed at larger scales than a draft boundary. As a result, some points have to be eliminated from the set of points for the draft boundary line. However, turning points might have to be preserved as intact points representing the topological junctions, i.e. the line intersections between public boundaries.

At this point of the space-time path, a political boundary can be demarcated on the ground, thus the *demarcation* event occurs and the *old* state is set on the space-time path (see Figure 5.1). All old boundaries are portrayed on basic maps (1:1250, 1:2500 and 1:10 000 scales) by boundary lines and by symbols representing their respective demarcation descriptions. The spatial relationship between a new boundary state and an old boundary state plays an important role in detecting any controversy between the interpretation of the legal definition of a public boundary and its equivalent geographical position on the landscape. Sometimes this controversy can provoke boundary disputes over the actual location of a public boundary. This dichotomy can arise for several reasons, such as having more than one interpretation of terms used in the delimitation event, as well as having a contradictory demarcation of the turning points along the boundary line. Generally, the uncertainty of geographical interpretation is more likely to be the culprit.

The episode for the demarcation event has an interval representation that is required to date the start and end of the demarcation event for a public boundary. Since most of the disputes concerning the actual location of a public boundary occur during the demarcation events, it is fundamental to have the dates when the dispute began, as well as the dates when actions were taken to rectify the disagreement.

Finally, the administration event can take place in a space-time path. The main reason for having *administration* events within a space-time path is the fact that a public boundary can change its position on the ground (see Figure 5.1). As the boundary changes its position, a transfer of territory from one authority to another will occur, causing changes in sovereignty and, possibly, changes in the socio-economic development of the border landscapes. The updates on a position of a boundary can occur by three basic changes:

- *Natural changes* The most common example is the displacement of a boundary position with the displacement of the watercourse by rivers and streams.
- *Man-made alterations* Updates are due to changes in the position of a boundary by opencast mining, erosion or overthrow of the ground features.
- *Attachment* Attachment of new descriptions to an existing boundary can lead to its position being updated.

New descriptions can also occur at any time in the existence of a public boundary. However, they are more likely to appear during the delimitation event when a boundary line is incorrectly portrayed on the original map in relation to its true position on the ground, and much later when the position of a boundary has been incorrectly demarcated on the ground.

The Ordnance Survey maintains regular perambulation measurements by which the surveyor confirms the displacement of the ground feature on the landscape. This survey plays an important role in the space-time path of the STDM since it represents the temporal relationship between the moment when the ground feature is updated on the map and the moment when the equivalent change is confirmed in the landscape. On the other hand, perambulation measurements also uncover mistakes and misinterpretations of the geographical terms used in the delimitation events. A common example is that a political boundary can be correctly portrayed on a map, but the actual boundary line was wrongly demarcated on the landscape.

Demise

Demise characterises the closure of a space-time path. It can occur on the space-time path at any time during the lifespan of a public boundary. However, this is more likely to occur when a public boundary is no longer operative or effective. In other words, when a public boundary reaches its obsolete state (see Figure 5.1).

5.2 EVOLUTION IN DEFINITION

The events and states from the space-time paths have been modelled as object classes. However, they play different roles within the STDM. Events are used to describe what happened, is happening, or will happen during the lifespans of public boundaries. On the other hand, states tell us what has changed, is changing, or will be changed during the lifespan of public boundaries. The main advantage of this modelling decision is that events can be modelled, exist within the database, and interact with the states of a public boundary without depending on the changes in the states themselves. Changes occur at the instance level (object level) in such a way that a public boundary has its space-time path depicted by different instances of different classes representing the events and states. All the instances are connected through the incremental modification mechanism based on the inheritance construct of object orientation (see Chapter 4, Section 4).

`GroundFeature` and `PublicBoundary` have been identified as generic classes within the STDM (Chapter 3, Section 3, explains system-defined classes). Each of them contains objects which embody some state, exhibit certain behaviour, and are uniquely identifiable. `GroundFeature` represents every physical feature in the landscape that has been assigned to be a public boundary object. Likewise, the `PublicBoundary` class denotes the political boundary itself. Considering the generic classes of the STDM, we can now associate them with their respective system-defined types in the following manner:

- *Generic classes*
  `PublicBoundary`
  `GroundFeature`
- *Versioned classes*
  `DraftBoundary` `ObsoleteBoundary`
  `NewBoundary` `GroundFeatureRevolutionaryState`
  `OldBoundary` `OldBoundaryRevolutionaryState`

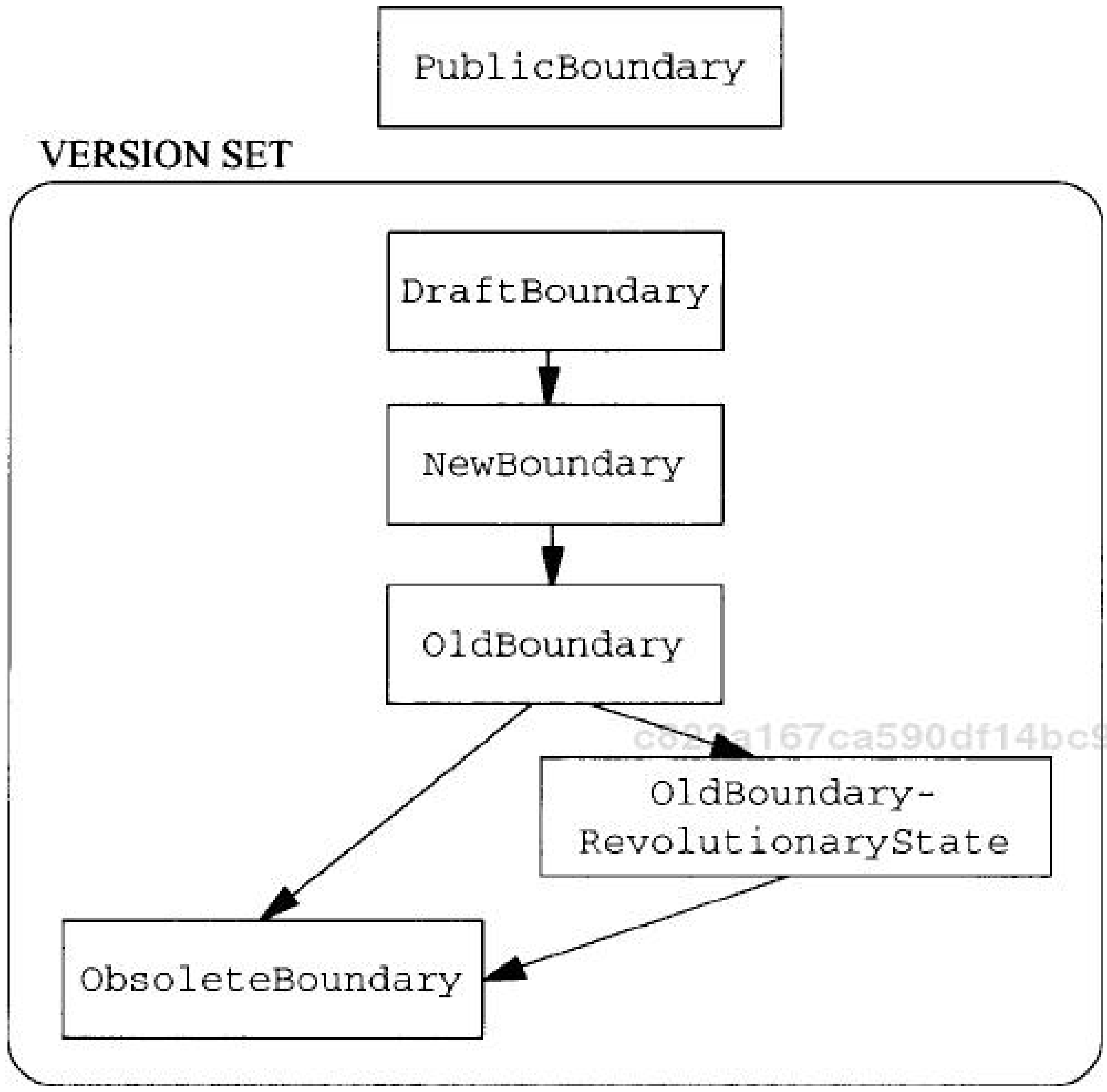

Figure 5.2 The version graph of the STDM

- *Unversioned classes*

  `Assumption` `Demarcation`
  `Allocation` `Perambulation`
  `Delimitation`

Versioned classes represent the different evolutionary states within the STDM. In essence the lifespan of a public boundary can be represented by these versioned classes. The instances of the versioned classes are the possible states of the space-time path of a political boundary. In other words, each state of a public boundary belongs to a different versioned class in order to create space-time paths within the system. Figure 5.2 illustrates the version graph that represents the link between versioned classes that contain the versions themselves in the STDM.

A version graph plays an important role in the temporal data management of the STDM because it helps to visualise the space-time path without depicting the events. Therefore it can be used as a modelling tool for designing the version management mechanism within the system. In defining what versioned classes are needed in the STDM, the next step is related to how the instances of these classes (i.e. the versions) should be identified and distinguished within the STDM. This is discussed in the following sections.

Unversioned classes represent the events over the space-time path, therefore they hold a time stamp which is the valid time corresponding to the lifespan of the state of a public boundary or the occurrence of an event. In the STDM, the time stamp is an attribute value associated with the valid time that can be nominal type (e.g. 27 May) or ratio type (e.g. event demarcation starts on 17 June and ends on 29 July).

The next section considers the four main scenarios devised as integrated subsystems within the STDM:

- *Public boundary entry scenario* Based on the creation circumstance of the space-time path, it is responsible for managing the allocation events in which a ground feature is assigned to be a public boundary.
- *Evolution tracking scenario* Based on the existence circumstance of the space-time path, it is in charge of managing all possible states of a public boundary (draft, proposed work, new, disputed, old, obsolete) and their respective historical events (allocation, delimitation, demarcation).
- *Update scenario* Based on the existence circumstance of the space-time path, it is responsible for updating public boundaries and managing all changes in the position of a public boundary.
- *Archiving scenario* Based on the demise circumstance of the space-time path, it is in charge of storing and retrieving the obsolete public boundaries.

5.3 FOUR MAIN SCENARIOS

Each of the four scenarios has been allocated differently in the physical view of the system. The process diagram (Figure 5.3) illustrates the different scenarios by allocating

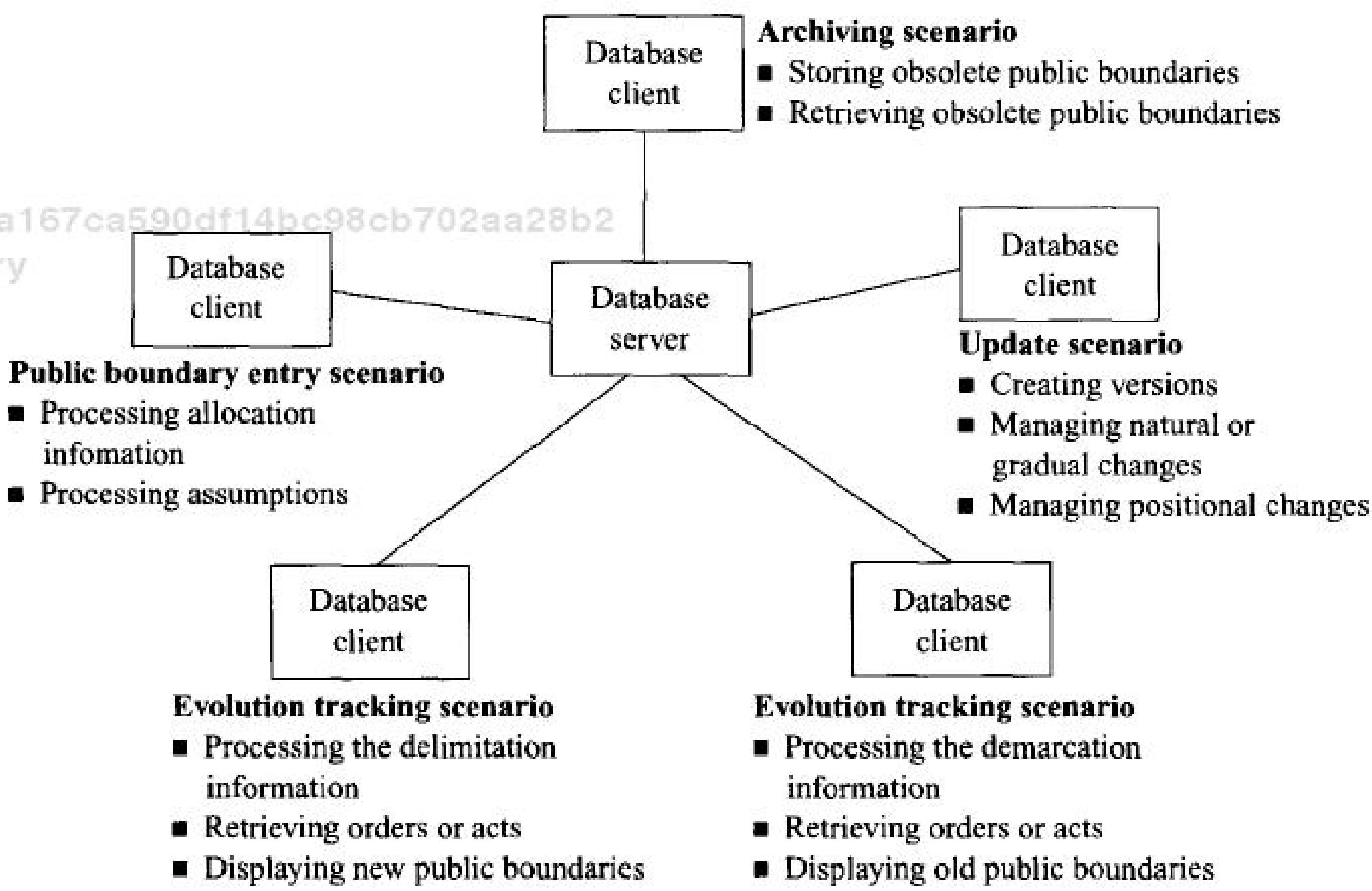

Figure 5.3 The process diagram

the appropriate processes to the processors in the physical view of the STDM. It can also be used to visualise the mapping of the architecture onto its realisation in code, and it shows the overlap between the analysis and design activities in developing the spatio-temporal model. Each scenario represents a potential path area at the analysis level as well as a client-server architecture of the system at the design level. This suggests a decentralised architecture, so most of the processing can be done by the client, reducing the bottleneck across the network.

5.3.1 Public boundary entry scenario

The public boundary entry scenario represents the creation circumstance of the space-time path within the STDM. It manages the states and events concerned with the allocation events in the STDM. And it involves the creation of the space-time path itself within the model. Four classes have been designed for the public boundary entry scenario: `DraftBoundary`, `GroundFeature`, `Assumption` and `Allocation`. The `GroundFeature` class represents the ground features on the landscape about which statements might be made. The `DraftBoundary` class comprises many `GroundFeature` objects, each of which might have different statements about them. The statements play an important role in the public boundary entry scenario. A statement can be related to an assumption which states that a ground feature can be regarded as a possible feature to be a public boundary. This denotes the mapping between an instance of the `GroundFeature` class and its corresponding instance of the `Assumption` class.

Once a ground feature has been selected to be a public boundary, this denotes the mapping between an instance of the `GroundFeature` class and its corresponding instance of the `Allocation` class. In this case the mapping is definite, hence unchangeable, because it depicts the creation of a space-time path within the system. As a result, an instance of the `DraftBoundary` class can be created. The class diagram in Figure 5.4 illustrates the design decisions regarding these object classes (Appendix C gives the entire class diagram of the STDM).

An inheritance relationship exists between the `Allocation` and `Assumption` classes. The `Assumption` class represents a superclass having the generalised statements about the allocation event. The `Allocation` class is a subclass representing a specialisation in which are added properties and methods from the superclass `Assumption`.

The aggregation relationship is assigned between the `Allocation` class and the `GroundFeature` class. This abstraction permits different instances of the `GroundFeature` class to be allocated as a possible public boundary. Allocating ground features to be a public boundary is deemed to occur in such a way that selecting a ground feature to be an instance of the `GroundFeature` class does not interfere with the properties of its corresponding instance of the `Allocation` class as a whole. And removing an instance of the `GroundFeature` class does not necessarily delete all its corresponding instances of the `Allocation` class.

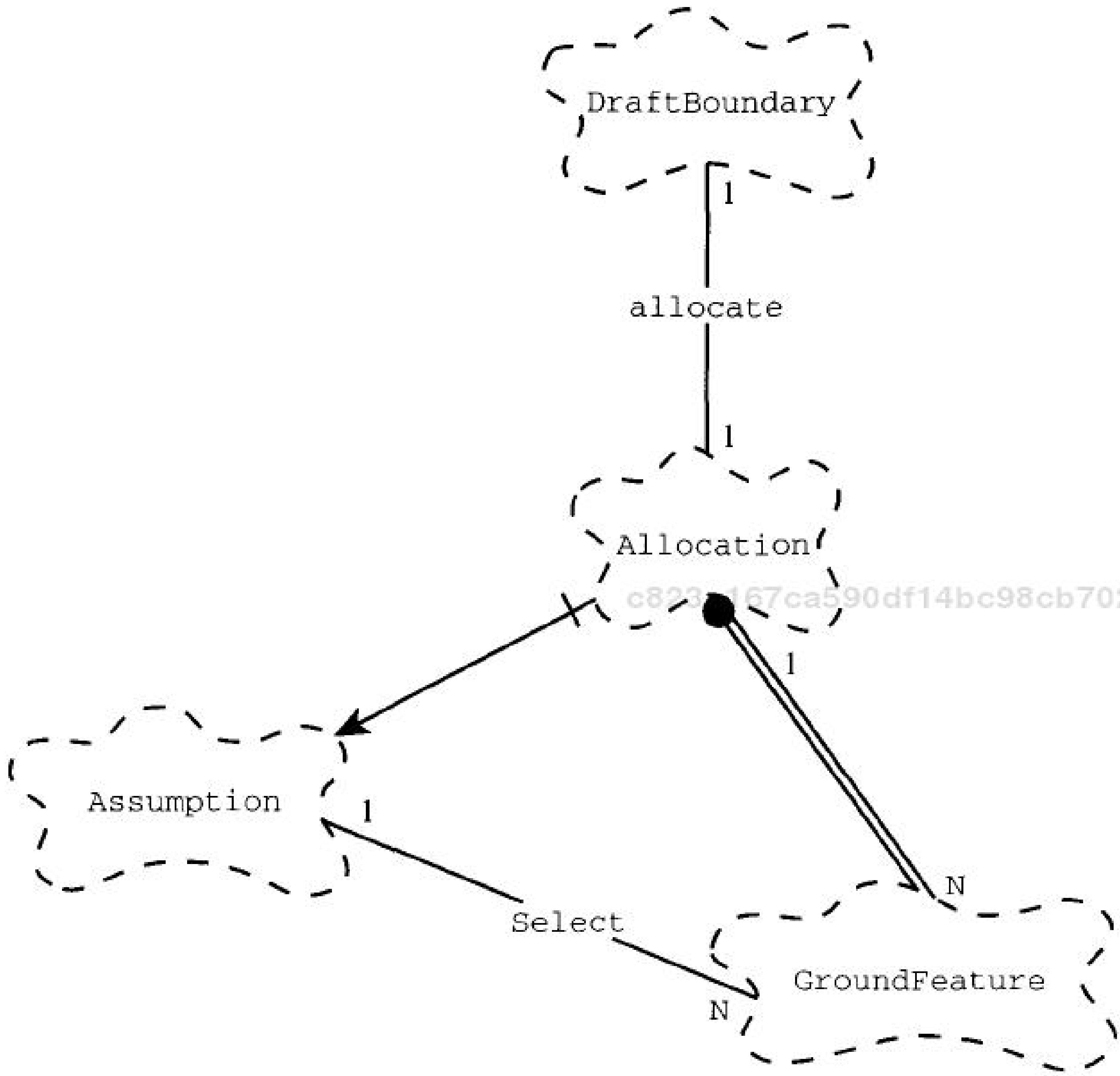

Figure 5.4 The class diagram for the public boundary entry scenario

The `allocate` and `select` associations in the class diagram denote a semantic dependency and suggest a bidirectional navigation between classes. For example, given an instance of `GroundFeature`, we should be able to locate the respective object denoting the `DraftBoundary`, and vice versa. Besides, these associations represent the independent incremental modifications in the STDM.

In this case, `GroundFeature` is the parent class having `Allocation` and `Assumption` as modifiers into a resultant `DraftBoundary` class. Table 5.1 shows the properties defined for each of the classes involved in the public boundary entry scenario. Although these classes are interrelated through an incremental mechanism, each one has its own properties. The `HistoricalView` class in the public boundary entry scenario represents this independent incremental mechanism of the STDM. It denotes the union of all properties which belong to the parent, modifier and resultant classes in the scenario (Chapter 4, Section 4 explains the independent incremental modification of space-time paths). As a design decision, the `HistoricalView` class has been defined for visualising the space-time path involved in the public boundary entry scenario. The historical view provides a snapshot of all the properties

Table 5.1 Class properties.

Class	Properties
`DraftBoundary`	line length mereing_point[a] type {*draft creation not confirmed, draft creation confirmed, proposed works*} description {*base of, centre of, foot of, 1.00 m from, not applicable*}
`GroundFeature`	point line area type_ground_feature (p. 102) {*stream, fence, hedge, wall, pond, undefined, defaced*}
`Allocation`	map_scale_used_for description date [*year, month, day*]
`Assumption`	statement validated {*yes, no*} valid [*from, to*]

[a] See Appendix B.

within the system, which have been involved in the assumption and allocation events of a specific public boundary selected by the user up to a certain point in time.

During the execution of the public boundary entry scenario, many statements mapped through the `Assumption` class will be made about the feasibility of certain ground features becoming a public boundary. As the scenario moves closer to a final decision, these statements eventually become `Allocation` objects. The execution of this scenario is illustrated in Figure 5.5; this interaction diagram provides a global perspective of the various operations involved in the public boundary entry scenario, and it shows the behaviour of the system in terms of the interaction between instances of classes. These are the operations designated for the following classes:

- `GroundFeature`
  `draftBoundary`
- `DraftBoundary`
  `groundFeature`
- `Assumption`
  `make` `select`
  `value` `become`
  `mostRecent`
- `Allocation`
  `select`
  `assign`

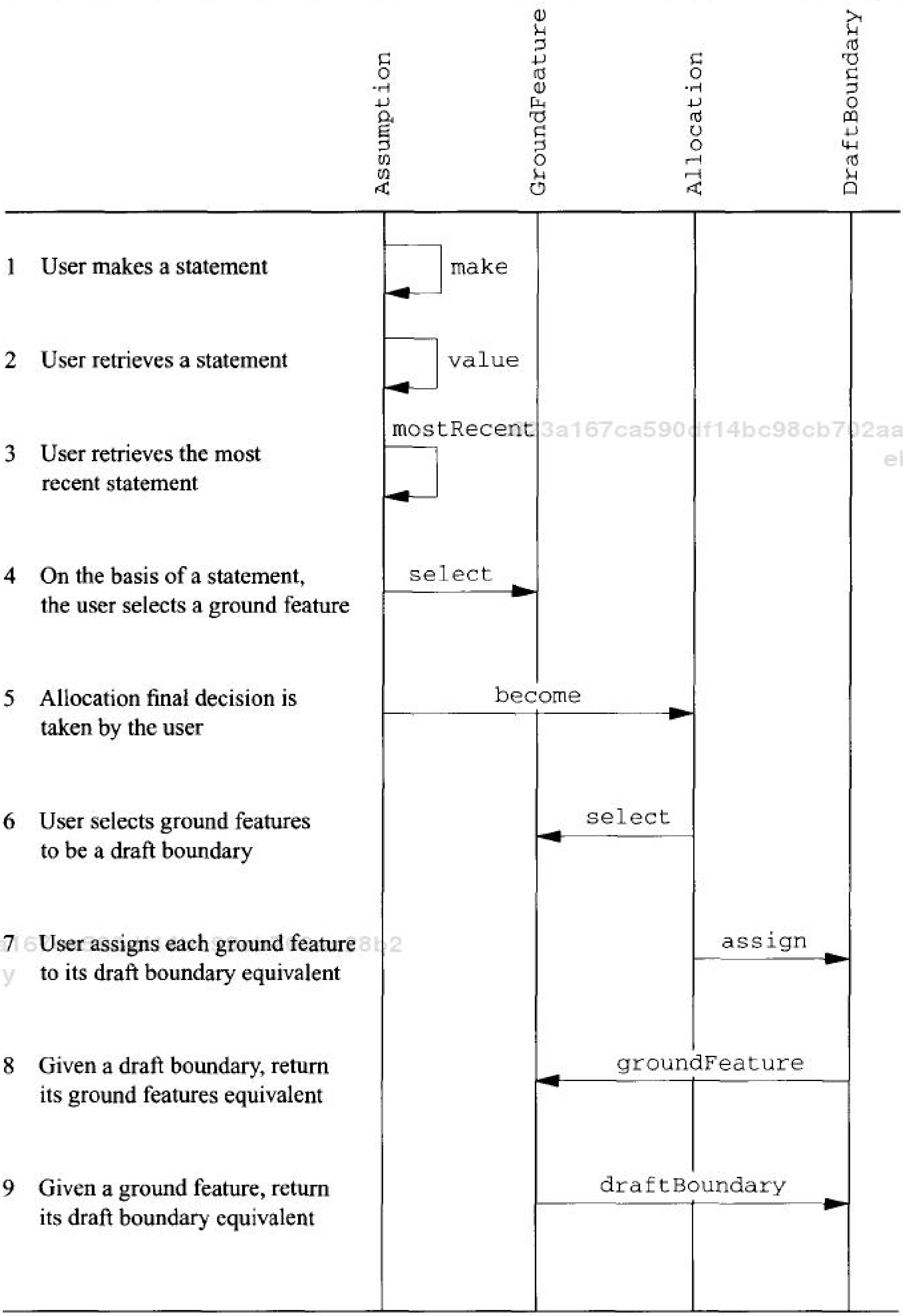

Figure 5.5 The interaction diagram for the public boundary entry scenario

5.3.2 Evolution tracking scenario

The main role of the evolution tracking scenario is to assign an order among events as well as the precise dates for the boundary-making process. Therefore, both delimitation and demarcation events from the space-time path are described in the evolution tracking scenario (Appendix C gives the entire class diagram of the model). In the delimitation event, an act or order confirms a draft boundary as a public boundary, so the boundary assumes a new evolutionary state in the STDM, called a new boundary state. The focus is on the design decisions regarding the structure and behaviour presented by the instances of the classes `DraftBoundary`, `Delimitation` and `NewBoundary`.

For the demarcation event, the position of a new boundary is ratified on the ground by surveyors, on the basis of the delimitation documents. In the STDM, the emphasis is on modelling the controversial aspects between the delimitation and demarcation of a public boundary over time. This involves the classes `NewBoundary`, `Demarcation`, and `OldBoundary`.

The `Delimitation` and `Demarcation` classes characterise the delimitation and demarcation events. `NewBoundary` and `OldBoundary` are classes defined to represent the new and old evolutionary states of a public boundary. On the basis of the analysis carried out for modelling the delimitation events, a prerequisite arises to carry out a line generalisation of a public boundary. This imposes different scales for portraying the same line that belongs to the `DraftBoundary` and later on when it has been assigned to be a `NewBoundary`. For example, consider displaying a specific line of a public boundary having a draft status at 1:2500 scale. Once this line has been elected to be a new boundary, its display might appear at one unique scale, i.e. at 1:10000 scale. And for clarity, some previous points belonging to the line in its draft state have to be eliminated.

There are two ways to produce a visual representation of the different states of a public boundary. An external class can be defined in order to query each object in terms of each kind of graphical display to be employed. Alternatively, each object can encapsulate the knowledge of how to display itself. The external class has been chosen as the better solution due to its closeness to the object-oriented concepts. For example, an object will be a line representing the public boundary. Having a draft state, this line will be displayed at 1:1250, 1:2500 or 1:10000 scale with the following graphical elements (Appendix B):

- Points portraying the change of a ground feature description (mereing points).
- Text properties portraying the description of the line, e.g. CR (centre of road), CS (centre of stream) and FF (face of fence).

In contrast, by having a new state, the same line will be displayed at 1:10 000 scale with the following graphical elements (Appendix B):

- Turning points portraying the line intersections between public boundaries.

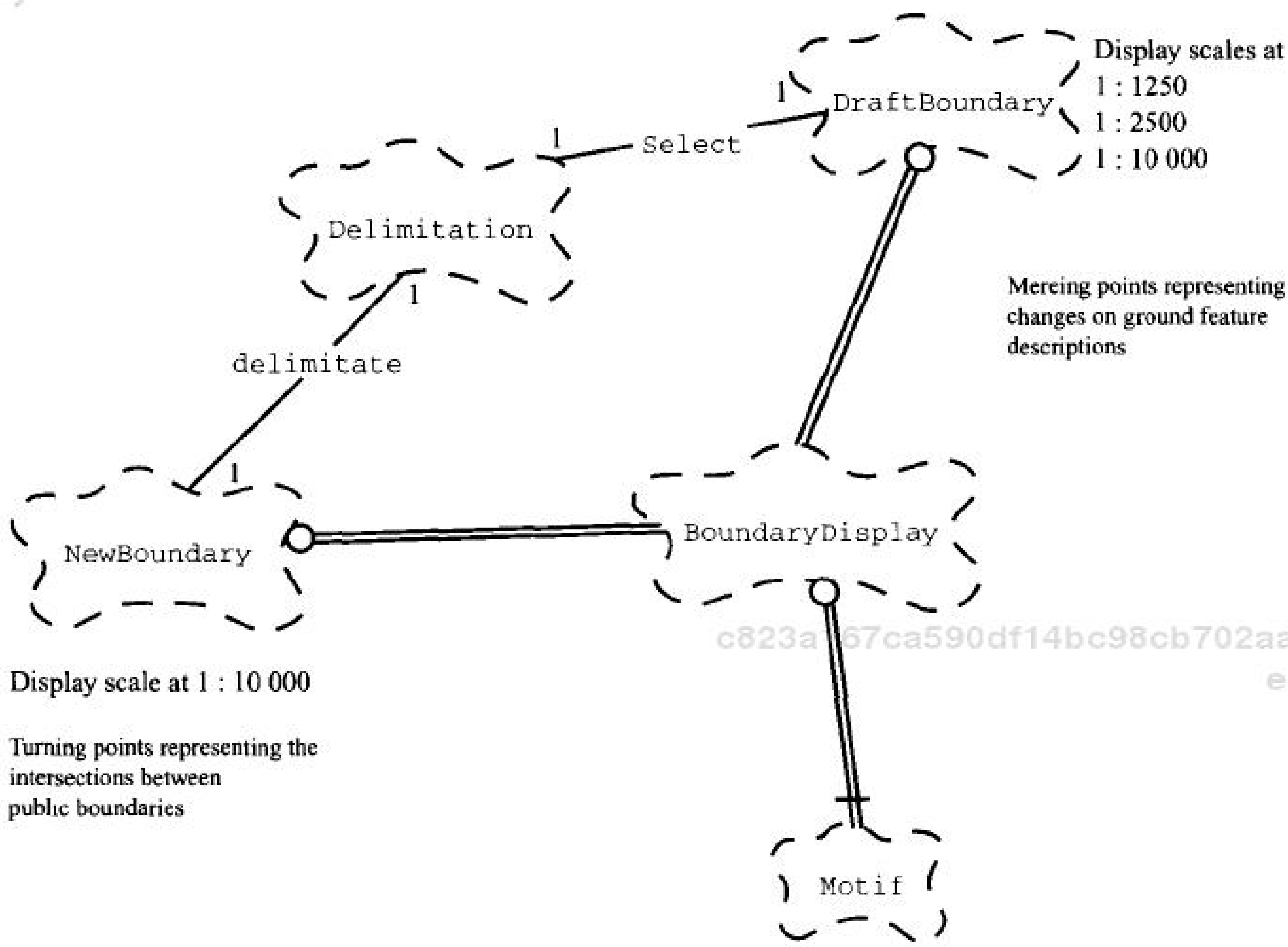

Figure 5.6 The class diagram for the evolution tracking scenario—delimitation event

Figure 5.6 illustrates the `DraftBoundary` and `NewBoundary` classes sharing a common class `BoundaryDisplay` through a *using* relationship within the model. Both `DraftBoundary` and `NewBoundary` are the clients of the display interface of the supplier `BoundaryDisplay`. The `BoundaryDisplay` provides the display routines for the graphical elements that each client requires. In implementation terms, the `BoundaryDisplay` is built upon a `Motif` class which deals with the off-the-shelf graphics package. The advantage of this design is that it allows a future replacement of the available display software with, for example, a hypermedia coordination which would display the objects in a dynamic manner. This would require the replacement of the display routines in the `BoundaryDisplay` class without the need to modify the implementation of every displayable object of `DraftBoundary` and `NewBoundary`.

The execution of this scenario is illustrated in Figure 5.7. The interaction diagram illustrates the behaviour of the system in terms of the interaction between instances of classes. The following operations have been identified for the respective classes given below:

- `DraftBoundary`
  - `select`
  - `notify`
  - `length`

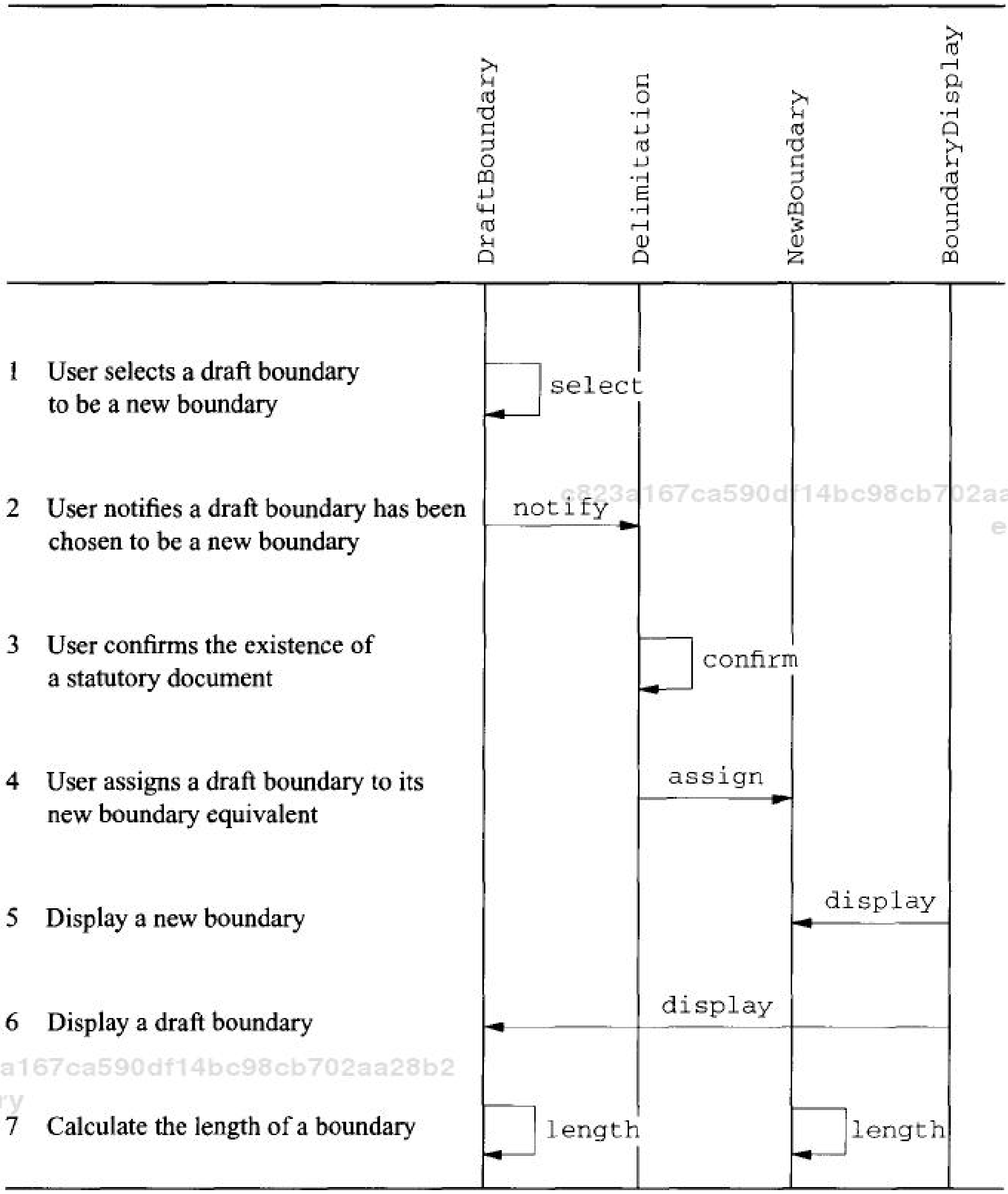

Figure 5.7 The interaction diagram for the evolution tracking scenario—delimitation event

- `Delimitation`
  `confirm`
  `assign`
- `NewBoundary`
  `length`
- `BoundaryDisplay`
  `display`

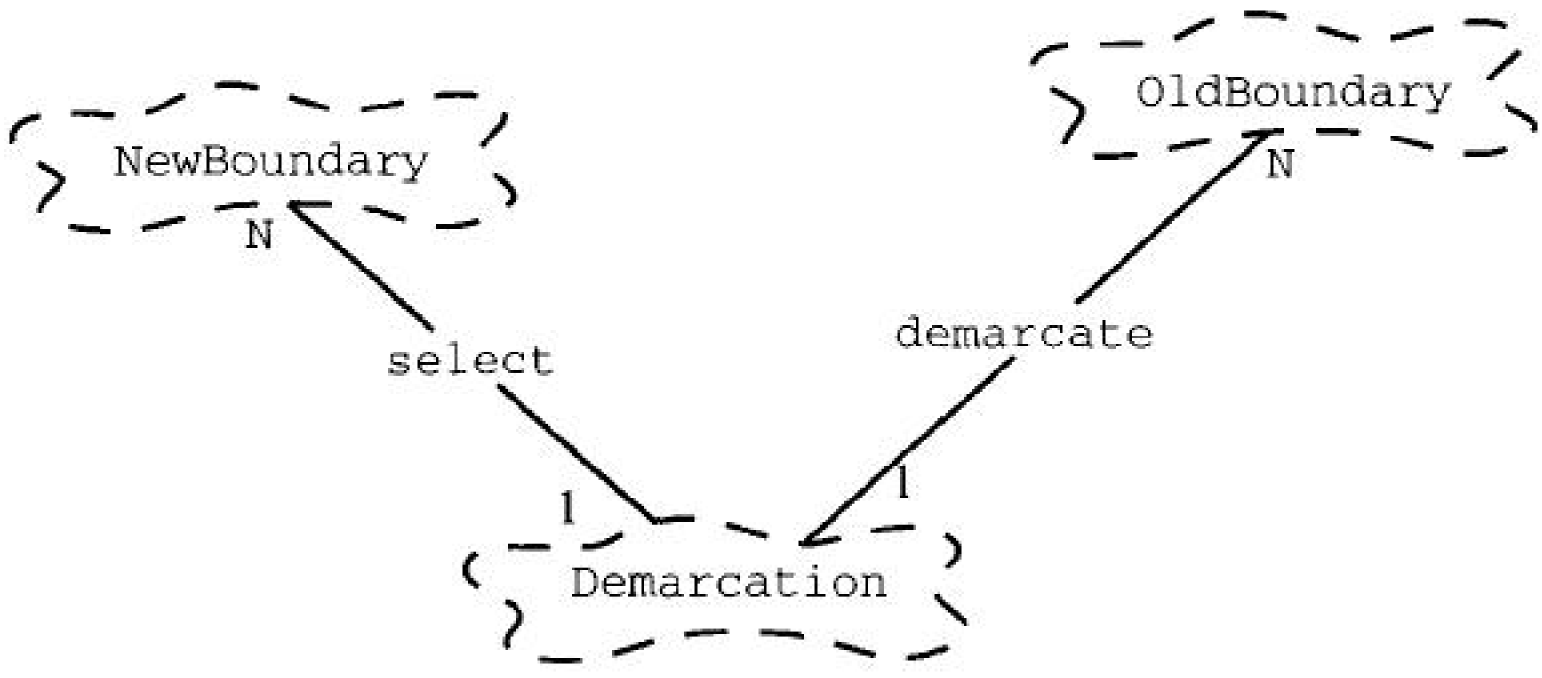

Figure 5.8 The class diagram for the evolution tracking scenario—demarcation event

In the evolution tracking scenario, the demarcation event involves the `NewBoundary` class, the `Demarcation` class and the `OldBoundary` class (Figure 5.8). `NewBoundary` represents the legal location of a public boundary. `OldBoundary` represents the old status of a public boundary. In other words, it characterises the actual location of a public boundary after the demarcation. The `Demarcation` class plays an important role within the STDM. It controls the flow of operations between `NewBoundary` and `OldBoundary`.

The interaction diagram in Figure 5.9 uses a box to represent the relative time that the flow of control is focused in a `Demarcation` class. Each instance of `Demarcation` is the ultimate focus of control, and its behaviour of carrying out a demarcation event invokes different methods over the instances of `NewBoundary` and `OldBoundary`. This is achieved by defining the following operations for the `Demarcation` class: `select`, `confirm`, `assign`.

The `delimitate`, `demarcate` and `select` associations in the class diagrams (Figures 5.6 and 5.8) denote a semantic dependency and suggest a bidirectional navigation between classes. They also represent the independent incremental modifications in the STDM. In the delimitation event, `DraftBoundary` is the parent class having `Delimitation` as the modifier into a resultant `NewBoundary` class. Moreover, in the demarcation event, `NewBoundary` becomes the parent class having `Demarcation` as the modifier into a resultant `OldBoundary` class.

Consequently, a user will be able to select a draft boundary to be a new boundary, and afterwards, to be an old boundary using the independent incremental mechanism. This allows users to enter values for the properties of classes involved in the evolution tracking scenario, taking into account the evolutionary aspects of the scenario. Classes are interrelated through the incremental mechanism in order to support space-time paths. Their properties do not depend on the existence of such a liaison between the classes. The independent incremental modification ensures that each of these classes has an independent lifespan. `HistoricalView` classes have been defined for visualising the space-time paths in the evolution tracking scenario, and Table 5.2 shows the properties attached to some of the other classes.

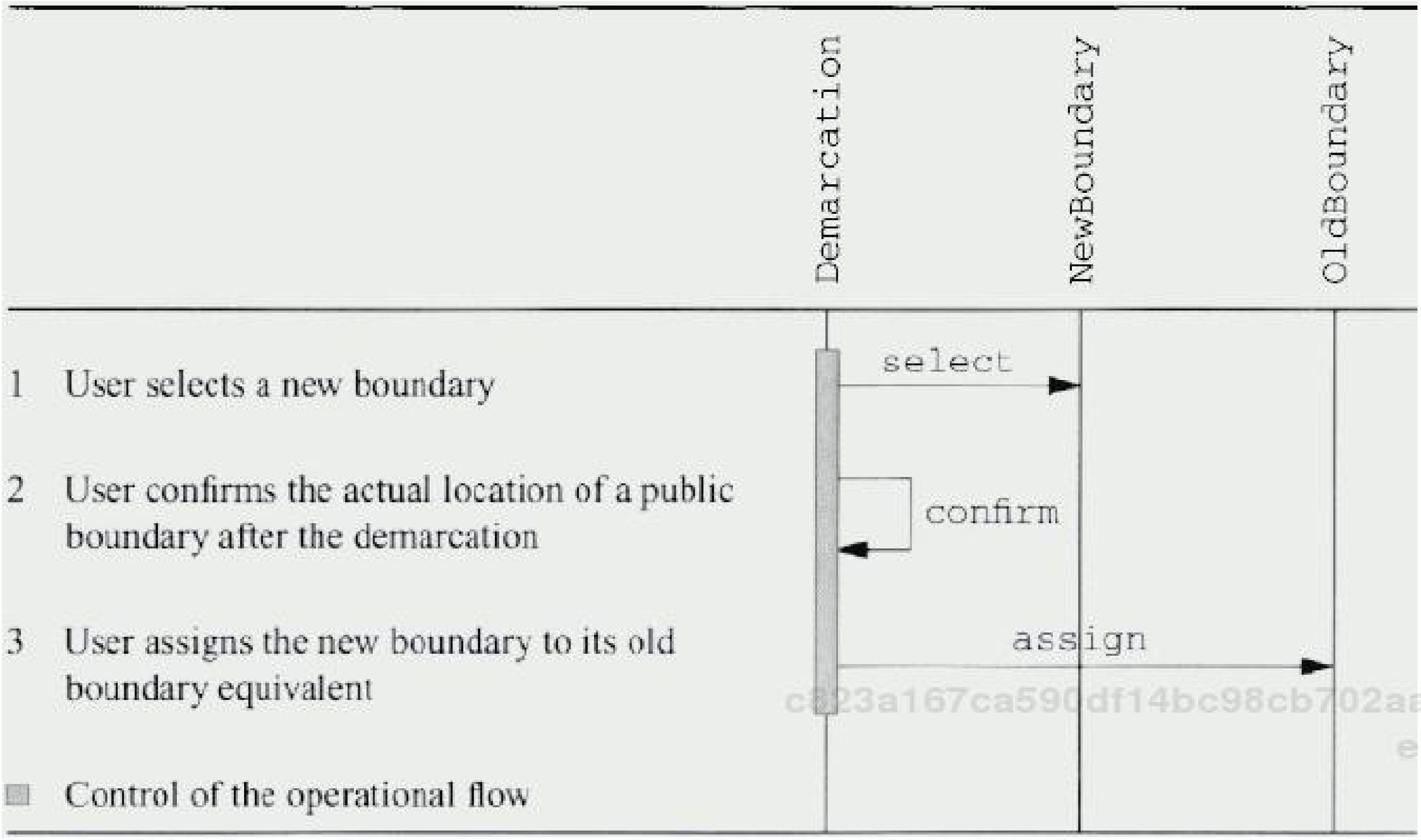

Figure 5.9 The interaction diagram for the evolution tracking scenario—demarcation event

Table 5.2 Class properties.

Class	Properties
Delimitation	statutory_document {*act of parliament* *boundary commission order*} map_scale_used_for operative_date [*year, month, day*] effective_date [*year, month, day*] actual_date [*year, month, day*]
Demarcation	surveyor_name map_scale_used_for valid {*from, to*}
NewBoundary	line turning-point length description {*operative*[a] *operative not effective*[b] *operative effective*[c]}
OldBoundary	line turning-point length type {*old, current, disputed*}

[a] Assigned by an act or order.
[b] Assigned by an act or order which has not yet become effective.
[c] Assigned by an act or order already effective.

Table 5.3 Vector representation: main update procedures.

	Update procedure	Change
1	Creation of a new object	nothing
2	Creation of a new object from an existing object	geometry, topology, theme
3	Status updating of an object	theme
4	Relocation of an existing object	geometry
5	Alteration of the spatial relationships among objects	topology
6	Relocation of an existing object with an alteration of its spatial relationship with other objects	geometry, topology
7	Status updating of an object with alteration of its spatial relationship with other objects	theme, topology
8	Status updating and relocation of an existing object	theme, geometry

5.3.3 Update scenario

In the update scenario every change is due to an update procedure, but not every update causes a change of state. When an update procedure generates changes, they are generated according to the spatial data representation being used in the GIS context. Among the spatial data representations available within GIS, the vector representation is the most complete because the geometric, topological and thematic properties of an object class can all be employed to describe changes. In contrast, a grid cell (raster) representation allows changes to be described using only thematic characteristics. For a vector representation, Armstrong (1988) defines eight possible update procedures and the changes that accompany them (Table 5.3).

The update scenario is deemed to handle the changes due to natural changes and new demarcation descriptions to public boundaries. Natural changes can be linked to update procedures 2, 4, 5 and 6 (Table 5.3). On the other hand, new demarcation descriptions in the update scenario are associated with update procedures 1, 2 and 4. So update procedures 1, 2 and 4 have been taken to illustrate the update scenario of a public boundary within the STDM. These update procedures have been designed to operate over the `GroundFeature` and `OldBoundary` classes in such a way that states are created for a public boundary. The class diagram in Figure 5.10 illustrates the design decisions regarding these object classes (Appendix C gives the entire class diagram of the STDM).

Both states on the diagram present an inheritance relationship with their corresponding instances. For example, an instance of the `OldBoundaryRevolutionaryState` class is created by inheriting some properties from its corresponding `OldBoundary` class. In this case the STDM is deemed to manage the overlapping incremental modification. New properties can be added to an instance of the `OldBoundaryRevolutionaryState` class whose names do not occur in its corresponding instance of the `OldBoundary` class. Table

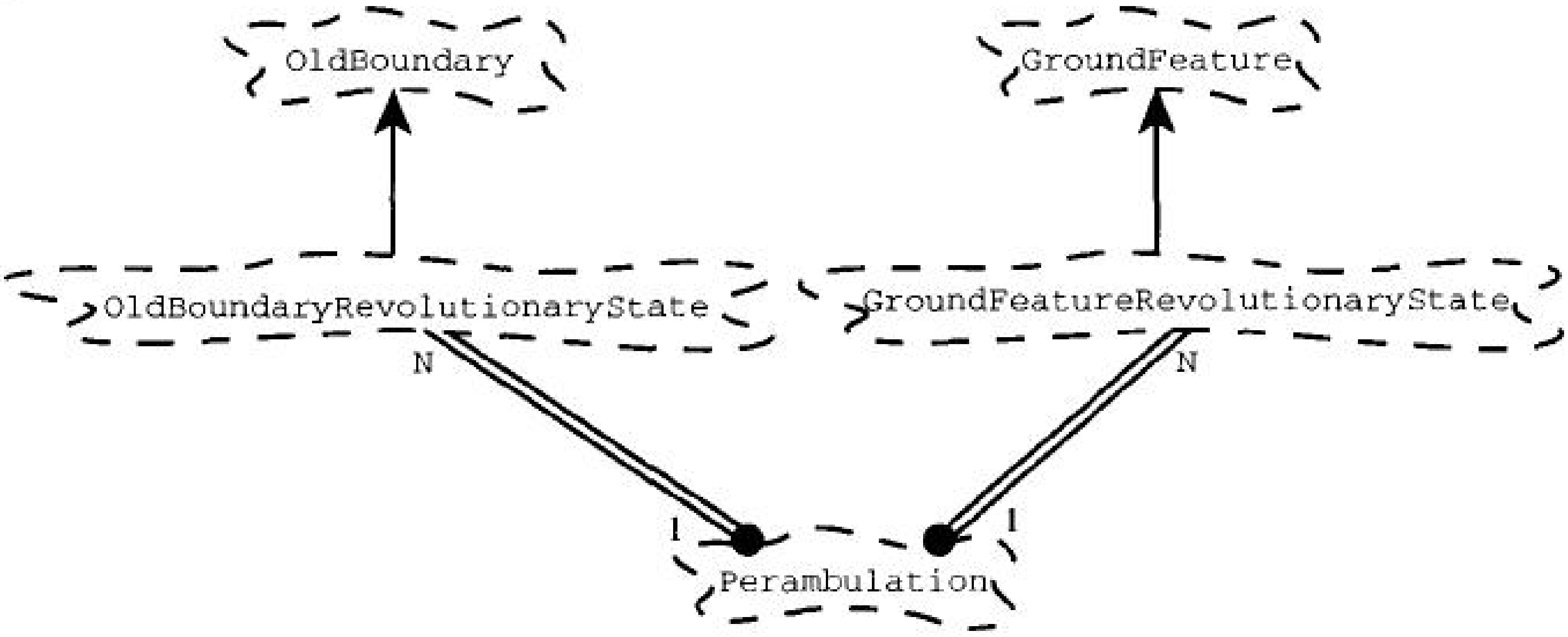

Figure 5.10 The class diagram for the update scenario

Table 5.4 Properties of GroundFeatureRevolutionaryState and OldBoundaryRevolutionary State classes.

Class	Properties
GroundFeatureRevolutionaryState	type[a]_ground_feature updated_point updated_line updated_area
OldBoundaryRevolutionaryState	type[a] updated_line updated_turning_point length

[a] Inherited properties from their respective parent classes.

5.4 illustrates the properties of the GroundFeatureRevolutionaryState class and the OldBoundaryRevolutionaryState class.

The exploratory nature of this design embraces two main aspects in creating discrete states using the overlapping incremental modification. First, the instances of the GroundFeature and OldBoundary classes are subject to the update procedures previously described. Second, there can exist several states for the same GroundFeature and OldBoundary classes. The update procedures play an important role in the update scenario. Figure 5.11 illustrates an example with three update procedures: creation of a new object, creation of a new object from an existing object, and relocation of an existing object.

5.3.4 Archiving scenario

Two classes have been defined in this scenario: OldBoundary represents the old state of a public boundary and ObsoleteBoundary represents the obsolete state

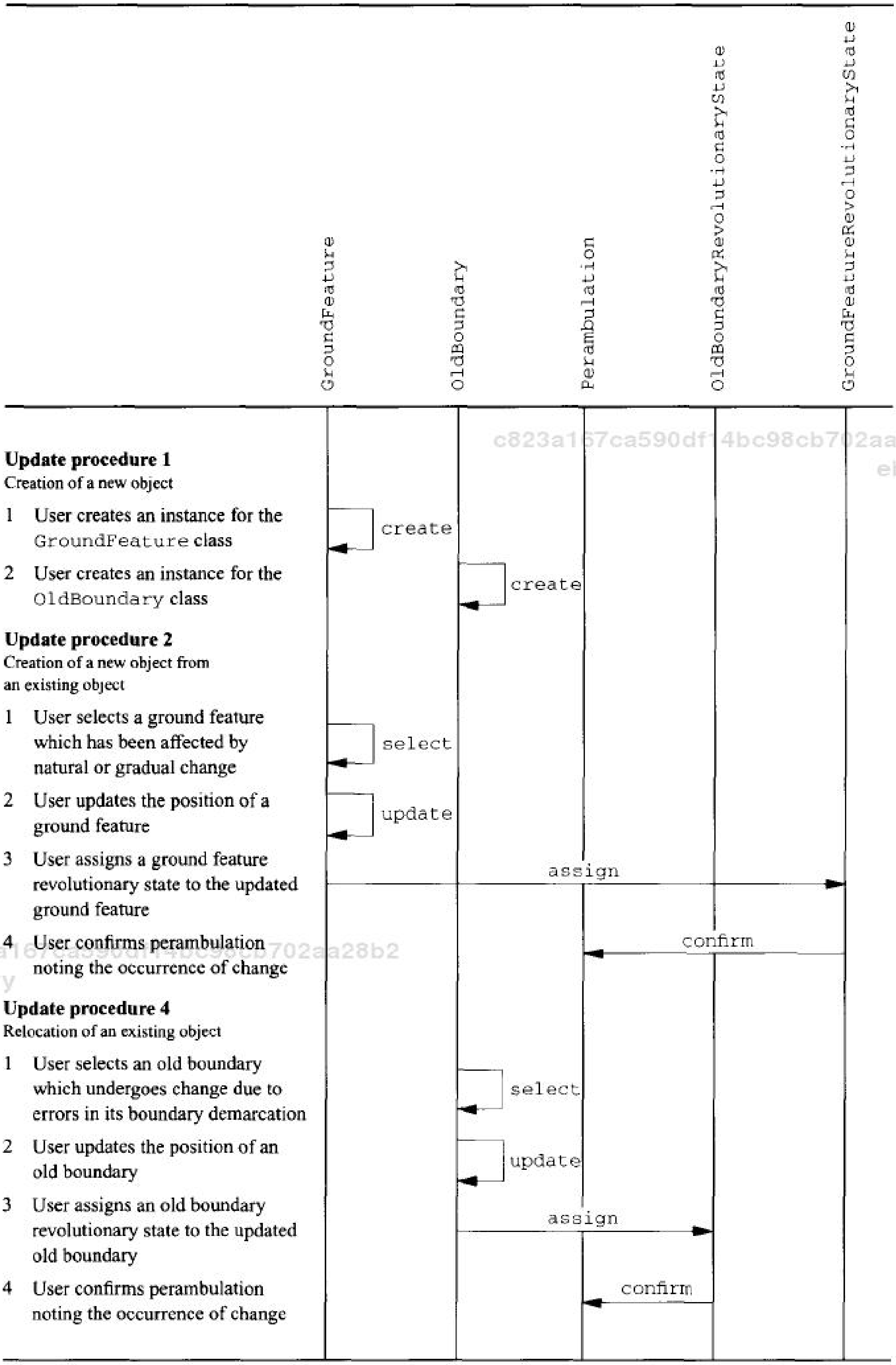

Figure 5.11 The interaction diagram for the update scenario

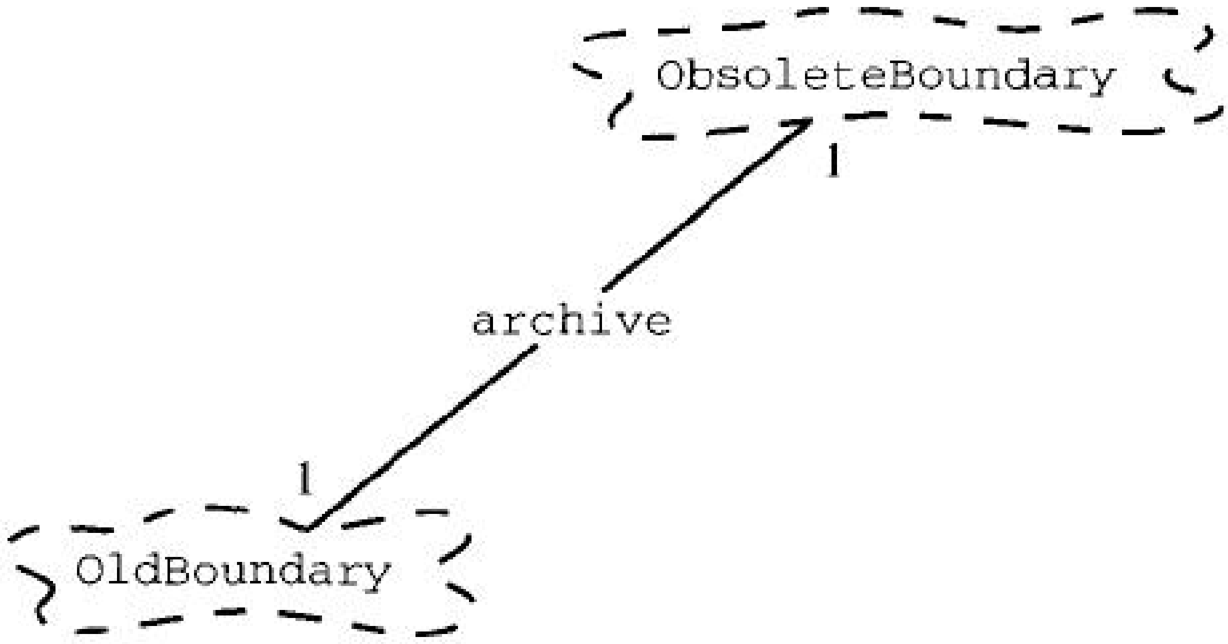

Figure 5.12 The class diagram for the archiving scenario

of a public boundary (Figure 5.12). This assumes the current data are associated to the `OldBoundary` class. Meanwhile `ObsoleteBoundary` represents the historical data. Current data are likely to have a higher query access frequency. Historical data archives will tend to become larger, hence they are most likely to be stored on cheaper optical disk. The execution of this scenario can be illustrated on an interaction diagram (Figure 5.13). The trigger method append assumes the current data are stored separately from the historical data, each kind having its own access method and possibly its own storage medium.

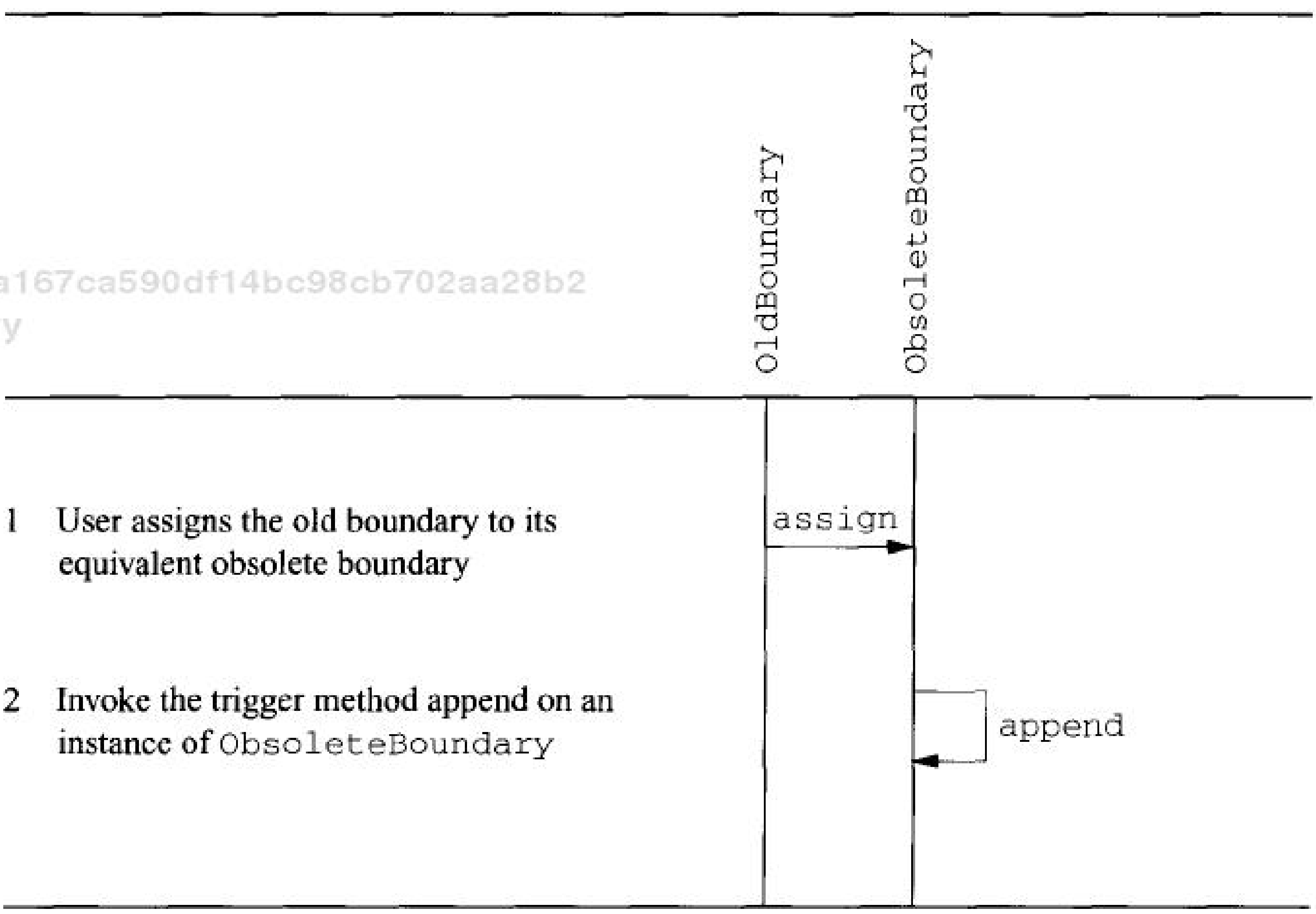

Figure 5.13 The interaction diagram of the archiving scenario

Table 5.5 Versioned and generic classes: classification of attributes.

`DraftBoundary`	
Version significant	none
Non-version significant	line, length, mereing_point, type, description
Invariant	none
`NewBoundary`	
Version significant	none
Non-version significant	line, turning_point, length, description
Invariant	none
`OldBoundary`	
Version significant	line
Non-version significant	none
Invariant	type
`ObsoleteBoundary`	
Version significant	none
Non-version significant	none
Invariant	line, turning_point, length, type
`OldBoundaryRevolutionaryState`	
Version significant	none
Non-version significant	none
Invariant	updated_line, type
`GroundFeature`	
Version significant	point, line, area
Non-version significant	none
Invariant	type_ground_feature
`GroundFeatureRevolutionaryState`	
Version significant	none
Non-version significant	none
Invariant	updated_point, updated_line, updated_area, type_ground_feature

5.4 VERSION MANAGEMENT

A different identifier (OID) has been associated with each version (state) in the STDM. This strategy has been taken in the STDM because it emerges as a more suitable way to track versions. The incremental modification approach designed in the STDM is based on an inheritance mechanism for the properties of an object, so it would be unfeasible to attach the versions at the attribute level. Versioning needs to be done at the instance level, i.e. at the object level. The properties of the generic and versioned classes of the STDM have therefore been divided into the three main categories described in Chapter 3 (invariant, version significant, non-version significant).

This classification is not deemed to embrace the infinite range of valid changes which can occur over public boundaries. It is based on the valid changes which have been associated with the update procedures defined for the STDM. They are deemed to handle the valid changes due to natural changes and due to new demarcation descriptions of public boundaries (see Chapter 3). Table 5.5 illustrates the classification, bearing in mind the update procedures previously defined in the STDM: creation of a new object, creation of a new object from an existing object, and relocation of an existing object. These update procedures are due to valid changes that have occurred over a public boundary. In the STDM they are applied to the `GroundFeature` and `OldBoundary` classes. Consequently, the classification shows both classes having version significant attributes. These version significant attributes must be updated in a structured manner. An update in one of these attributes pertaining to the `GroundFeature` class and the `OldBoundary` class creates versions which are respectively the instances of the `GroundFeatureRevolutionaryState` class and the `OldBoundaryRevolutionaryState` class (see Figure 5.10).

This classification has provided a unifying solution for effective management of versions within the STDM. It means that maintenance of versions within the STDM can be implemented using a GIS. Hence the abstractions developed in the STDM, such as space-time paths involving events and states, can also be implemented using a GIS. Version management will provide a mechanism for maintaining consistent update propagation that prevents unnecessary version proliferation within the system. And beyond the evolution of public boundaries, it may have much to offer other applications, both conceptually and practically.

5.5 CONCLUSIONS

Developing the STDM (Appendix C gives an overview of the spatio-temporal data model) using the object-oriented method proposed by Booch has elucidated several aspects concerning the time geography framework. First of all, the design decision to model events and states as object classes may not be appropriate in another application. Instead it might require events to be modelled as methods rather than object classes. This would certainly affect the design of the space-time paths within the model. But having decided to model events and states as object classes, the class diagrams in this chapter are valuable tools for analysis and design.

Second, the design of a space-time path has been defined at the instance level (object level). Different states, as well as events, have been defined as instances of classes having their respective relationships. However, the space-time path is only generated by connecting the instances of these object classes. This connection has been designed using an incremental modification mechanism. Unfortunately, Booch's method does not provide a diagram for modelling different scenarios at the instance level of object classes. However, the interaction diagrams have been used to illustrate how these scenarios are executed, as well as to visualise the process involved in 'making space-time paths' within the STDM.

Finally, the analysis and design of the STDM using an object-oriented method has raised issues in version management. Having a system with several instances of several object classes connected to space-time paths has definitely raised a question mark about versioning. What is the best approach for managing versions in order to avoid inconsistency in the representation of space-time paths within the STDM? In investigating the possibilities of designing a version management mechanism in the STDM, one of the main findings was the need to understand the meaning of 'change'. Change is responsible for the existence of states on a space-time path, and their interrelations with events. In other words, change carries out actions that create different instances in the space-time paths within the system. Therefore, change has to be well understood as an element of the version management approach of the STDM.

The next chapter describes the implementation aspects for incorporating the STDM into a GIS. The prototype implementation has been undertaken mainly as a 'proof-of-concept' for the ideas developed in the STDM.