VLDB Journal, 3, 445-477 (1994), Ralf Hartmut Güting, Editor

A Semantic Modeling Approach for Image Retrieval by Content

Wesley W. Chu, Ion T. Ieong, and Ricky K. Taira

Received July 2, 1993; revised version received, March 20, 1994; accepted May 20, 1994.

Abstract. We introduce a semantic data model to capture the hierarchical, spatial, temporal, and evolutionary semantics of images in pictorial databases. This model mimics the user's conceptual view of the image content, providing the framework and guidelines for preprocessing to extract image features. Based on the model constructs, a spatial evolutionary query language (SEQL), which provides direct image object manipulation capabilities, is presented. With semantic information captured in the model, spatial evolutionary queries are answered efficiently. Using an object-oriented platform, a prototype medical-image management system was implemented at UCLA to demonstrate the feasibility of the proposed approach.

Key Words. Medical, image, and multimedia databases, spatial query processing, temporal evolutionary query processing.

1. Introduction

Advances in medical imaging systems have revolutionized the acquisition of human images, providing views of cross sections and physiological states using a variety of modalities, including x-ray, computed tomography (CT), and magnetic resonance imaging (MRI). A medical Picture Archival and Communication System (PACS) infrastructure has been developed at our institution to provide efficient archival, retrieval, communication, and display of the large repository of digital medical images (Huang et al., 1988; Taira and Huang, 1989). The digital image data were acquired automatically by the PACS, which contains two optical disk library units, each with a capacity of one terabyte. At present, image retrieval in these systems is based on file names or artificial keys such as patient hospital identification numbers, which limits querying capabilities. There is a need to retrieve the images by content. For example, in therapy treatment planning, the therapist is often interested in

Wesley W. Chu, Ph.D., is Professor, Department of Computer Science, University of California, Los Angeles, CA 90024; Ion T. Ieong, Ph.D., is Senior Principal Engineer, Advanced Development Group, AT&T Global Information Solutions, 100 N. Sepulveda Blvd., El Segundo, CA 90245; and Ricky K. Taira, Ph.D., is Assistant Professor, Department of Radiological Sciences, University of California, Los Angeles, CA 90024.

retrieving historical cases that demonstrate diagnostic image features (e.g., object density, texture, location, extent, clustering patterns, shape, and size.).

A knowledge of the spatial content of a medical image is especially important in surgical or radiation therapy of brain tumors because the location of a tumor has profound implications for a therapeutic decision. Functional mapping of the brain requires the correlation of physiologic function to anatomical locations in the brain. The evolutionary content of a sequence of images is important when studying the treatment-response effects of various therapies. Also, the temporal characteristics of image features help describe the behavior of disease processes. For example, what is the normal growth rate of tumors for a given classification of patients? How does this growth rate change with various regimens of drug/radiation therapy?

Several approaches have been proposed to retrieve images in pictorial databases. Feature indexation indexes and retrieves image objects on the basis of their features (Grosky and Mehrotra, 1990). Symbolic pictures use orthogonal relationships to encode the image objects into a 2-D string (Chang et al., 1987). As a result, user queries concerning the orthogonal relations of the objects can be answered. The information retrieval approach (Rabitti and Savino, 1991, 1992) transforms the image and the query into signatures. The query is answered by matching the signatures. These approaches are symbolic, and cannot be used to solve direct spatial queries that require point addressing capabilities.

To solve direct spatial queries, spatial data structures can be used to store the minimum bounding boxes of the image objects (Roussopoulos et al., 1988; Gupta et al., 1991). Since a bounding box is an approximation of the original image object, representing the image object with such structures may not allow the user to retrieve all the answers that satisfy the query specification. Image objects can also be represented in point sets (Orenstein and Manola, 1988). Spatial relations among the image objects are dynamically determined through pixel-level manipulation during query processing.

Images of biological objects are evolutionary in nature and can capture object states at different times. To study the spatial characteristics of an object or its growth pattern requires querying objects with spatial and evolutionary characteristics. Spatial queries deal with the spatial features and relations of image objects:

- *Symbolic spatial queries* express the query content of image objects with specific spatial characteristics such as type, shape, size (e.g., area, volume, length, diameter), spatial relations (e.g., `Separated`, `Contains`), and orthogonal relations (e.g., `SouthWest`, `EastNorth`). They allow the user to express the query content through alphanumeric descriptions. For example: *retrieve images with a microadenoma*[1] *with a diameter of 5 mm or larger,* or *retrieve images with a macroadenoma invading sphenoid sinus.*

1. Microadenoma is an adenoma (brain tumor) less than 10 mm in diameter which can evolve into a macroadenoma 10 mm or larger.

- *Direct spatial queries* require point addressing capabilities of the image objects. They allow the user to directly manipulate the points and non-zero size objects in the images, and to search for spatial objects over a certain area. For example: *retrieve images with an adenoma extending within a 20 mm radius from the center of the hypothalamus.*

Biological objects evolve, fuse, split, and change their spatial orientations over time. The therapist is often interested in retrieving historical cases that demonstrate diagnostic image features. Spatial evolutionary queries deal with the evolution of objects captured by images at different times. Consider the query, *retrieve the image frames demonstrating a microadenoma 5 mm or larger in diameter developing inside the pituitary gland, which has evolved into a macroadenoma invading the sphenoid sinus in one year's time.*

The existing image-retrieval approaches in a large medical database have the following problems:

1. *Inadequate query capabilities.* Previous research often focused on a specific application domain and supports only some forms of symbolic spatial query answering. However, few approaches support direct spatial query answering (Roussopoulos et al., 1988), and even fewer support temporal spatial query answering (Orenstein and Manola, 1988) or temporal evolutionary query answering (Chu et al., 1992). None support spatial evolutionary query answering.
2. *Lack of semantic information.* Due to the lack of definition and organization of image semantics, few approaches use semantic information. Data models in Mohan and Kashyap (1988) capture hierarchical and temporal information, but none capture high-level spatial evolutionary semantics or incorporate the features extracted from the images in the data model.

The semantic data model, VIMSYS (Gupta et al., 1991), consists of four layers: IR, IO, DO, and DE. IR represents the base type of the system. IO represents the images and their features. DO specifies domain entities, and DE accommodates events defined over time.

Our model consists of two layers. The lower layer represents the related images and object contours in a number of stacks (Chock et al., 1984). The upper layer abstracts the objects from the images. In addition, semantics of the images are captured by modeling the relations among objects, and are expressed by the hierarchical, spatial, temporal, and evolutionary constructs. Our model mimics the user's conceptual view of the image content, providing the framework and guidelines for image preprocessing to extract features and relations of image objects. Based on the model constructs, a spatial evolutionary query language (SEQL) is developed to express spatial evolutionary queries, providing direct manipulation capabilities of image objects. Using the proposed model, we can express spatial, temporal, and evolutionary objects captured by a set of snapshots taken over time. Thus, queries with temporal, evolutionary, and spatial predicates can be answered.

Using Gemstone,[2] a commercial object-oriented database with ObjectWorks/ VisualWorks as the development environment, a prototype medical image management system was implemented at UCLA to demonstrate the feasibility of the proposed approach.

In Section 2 of this article, we describe feature extraction from images. A conceptual data model is presented in Section 3. The SEQL for expressing user queries is introduced in Section 4, and the strategy for processing SEQL queries is addressed in Section 5.

2. Feature Extraction

The PACS stores a variety of medical images including X-rays and MRI. Each image type has its own specialized image segmentation methods which take advantage of its image characteristics. The methods range from fully automatic segmentation software to semi-automatic computer assisted methods. Additional information is available through diagnosis by physicians. The advances of image segmentation techniques greatly influenced the design strategy of the image management system (to selectively extract the image information, and to organize and store it to provide efficient content-based retrieval).

Preprocessing of the images consists of the following steps:

1. Identify the objects in the images.
2. Detect the boundaries of the objects.
3. Obtain the spatial features and relations of the image objects.

Current approaches can be broadly categorized, based on intensity discrimination (Brummer et al., 1989) or edge detection (Bart et al., 1991). The intensity approach assumes that a fundamental relationship exists between the pixel intensity and the physical substrate. However, numerous studies have shown that factors such as field inhomogeneity, instrumentation noise, and partial volume all contribute to pixel intensity fluctuation, and create uncertainty in discrimination. Current work in edge detection shows some degree of success (Marr and Hildreth, 1980) but, in general, user interaction is required. Visual interpretation plays an important part in correlating edges in different scales, as well as in making decisions on true or false edges. The lack of a rigorous means of performing both multiscale edge analysis and parsing has seriously limited the capability of automated segmentation. Most of the current work in segmentation of anatomic objects of the brain centers around wavelet transforms (Daubechies, 1988) and model-guided methods. However, because of

2. We selected Gemstone (Servio Corp.) because it has a richer type system than relational database management systems, which is essential to the development of the new modeling constructs proposed in this article. Furthermore, Gemstone has a gateway to Sybase, which allows us to retrieve the stored patient demographics at the UCLA medical center.

fewer objects and better contrast in X-ray images of the hand, feature extraction is accomplished automatically (Pietka et al., 1991*a*, 1991*b*).

To accommodate different types of images with selected segmentation software in a system, we need to provide guidelines for the types of information to be extracted from the images and the structures to organize them so that they can be effectively retrieved. The types of information extracted from the images for spatial query answering are *object contour, spatial features,* and *spatial relations.*

Object contours are stored as bit maps in the system. A number of spatial characteristics on the object can be derived from the contour, for example, area, volume, circumference, bounding box, and 3D-volume rendering (Angel, 1990).

Spatial features, such as type, shape, area, volume, diameter, length, and circumference, describe the spatial characteristics of an image object. For example, to measure the circumference on a cross section of a macroadenoma, the contour showing its boundary is detected as shown in Figure 1. The number of pixels of the contour is counted and the circumference of the macroadenoma is obtained.

Spatial relations describe the relations between a pair of spatial objects, including:

- Orthogonal relations describe the directional relationships between objects, such as `East`, `South`, and `SouthEast`.
- Containment relations describe the relative position and the locations of contact between a pair of objects, such as `Invades` and `Contains`.

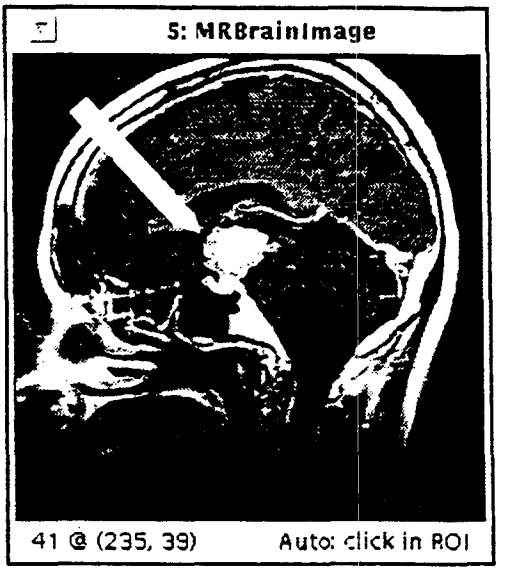
The spatial relations among objects such as containment can be determined by computation with object contours (Angel, 1990; Mantyla, 1988). Such information derived from the object contours is used for answering symbolic spatial queries efficiently. The object contour is also essential for answering the direct spatial query since it provides detailed object boundary information at the pixel level.

The number of spatial relations for n objects in an image is on the order of $O(n^2)$. Storing all the spatial relations not only increases the storage but also lengthens the time needed to search for the objects. To limit the size of storage of spatial features and relations, only those most frequently used are stored in the system. There is a storage/response time tradeoff as to what spatial features and relations need to be stored.

Based on anatomical knowledge and physician diagnosis, the spatial relations of an object with respect to its surrounding objects can be derived; we call them the primary spatial relations of the objects. We use primary spatial relations to represent the semantics of the images in the data model. In addition, the relationships of abnormal objects discovered in diagnosis with respect to their adjacent objects are also kept in the system. Due to the relatively small number of abnormal objects found, we only need to maintain a small fraction of spatial relations in the system.

To support the spatial queries that directly manipulate the image pixels and determine the infrequently referenced spatial relations, we need the object contours

Figure 1. Sagittal image on cross section of brain



The boundary of a macroadenoma (pointed by an arrow) is outlined.

information that describes the object boundary in pixels. Although this information requires a large amount of storage, it is essential to answer these queries. Not storing object contours has the following problems:

- Using an approximation of object contours such as the bounding box cannot provide accurate spatial query answers.
- Performing image segmentation on-the-fly is very time consuming. For example, to extract features on an X-ray image of the hand takes four minutes on a Sun Sparc II (Pietka et al., 1991*a*, 1991*b*). To scan through all the images and extract features during query execution is not feasible. Further, certain image segmentations require human expert guidance and interpretation.

Therefore, storing high-level semantic spatial information in the database enables us to efficiently answer symbolic spatial queries. Further, such symbolic spatial information and object contours can also improve the processing efficiency of direct spatial queries.

In our system, related images and object contours are stored in stacks (Chock et al., 1984) based on temporal or spatial correlation among images. For example, an MRI brain scan consisting of about 70 cross-sectional 2-D images for one patient is stored in one stack. A time series of X-ray hand images for the same patient taken at different points in time is stored in another stack. The organization of the semantic information will be addressed as follows.

3. Conceptual Data Model

A conceptual data model is needed for high-level representation of image objects to capture the hierarchical, spatial, temporal, and evolutionary semantics of the images; to provide the framework and guidelines for the image segmentation and feature extraction processes; and to allow direct manipulation of the image objects. There are several characteristics of the medical images considered in the data model:

- *Similarity among medical images.* Images of a body part are very similar for the same growth period. Therefore, object classes (object class and object type are used interchangeably in this article) are created to collect similar object instances on the same body part.
- *Differences of imaging modalities.* Different imaging modalities such as x-ray, CT, MRI, and PET scans can be applied to the same body part. A conceptual model can be used to relate different types of images of the same object that present different views and physical characteristics of the object.
- *Object evolution.* Biological objects grow and evolve with time. An object in a certain growth stage can evolve, fuse, or split into a set of different objects. For example, a brain tumor may grow from small to large and/or split into two segments. To reflect the trend of object development, image sequences of evolving objects should be related together in the data model.

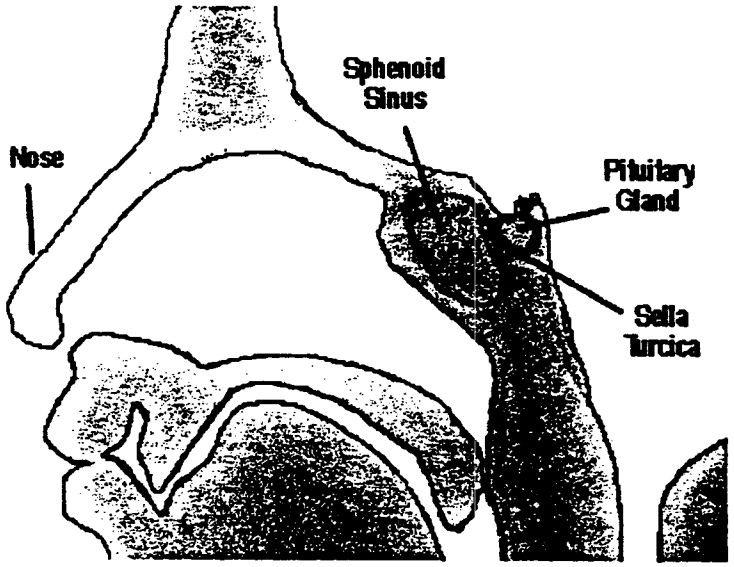
To illustrate the conceptual data model, we refer to the example of a pituitary gland, and use the object constructs to model the development of an adenoma (main abnormality of the pituitary gland) with respect to its surrounding objects. As shown in Figure 2, the pituitary gland consists of several constituent objects, which can be modeled by hierarchical constructs. The pituitary gland is encapsulated by the sella turcica, which can be modeled by spatial constructs. A microadenoma developing in the pituitary gland may evolve into a macroadenoma over time, and this development can be modeled by temporal and evolutionary constructs.

As shown in Figure 3, the conceptual data model consists of two layers. The lower layer represents the related images and object contours in various stacks (Chock et al., 1984). The upper layer abstracts the objects from the images. Semantics of the images are captured through the modeling of the relations among objects and expressed by the hierarchical, spatial, temporal, and evolutionary constructs (Section 3.2).

3.1 Abstraction of Image Objects

Depending on the image types, objects are identified from the images by segmentation software or expert identification, and represented in the data model. Key information that represents the object characteristics is extracted. Similar image object instances of the same body part are placed in the same class. For example, the object class defined for the microadenoma found in the images of our Gemstone implementation (Ullman, 1988) is:

Figure 2: Pituitary gland in the sella turcica



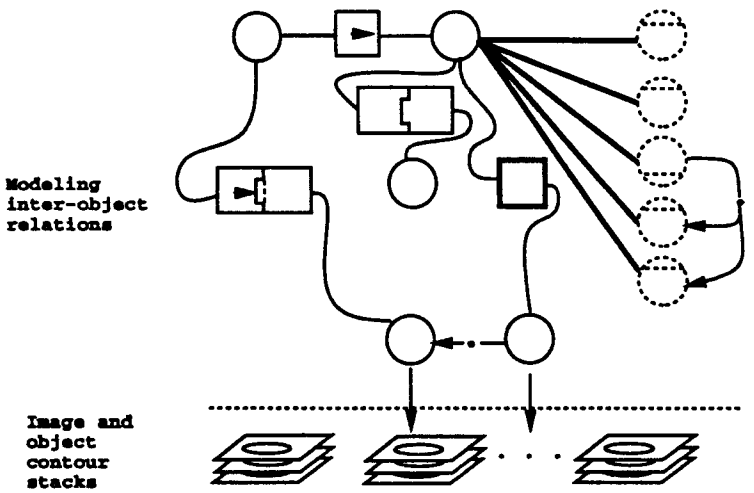

```
Object subclass 'Microadenoma'
    instVarNames: #['type,' 'shape,' 'diameter,'
        'volume,' 'eventTime,' 'contourPointer,' 'imagePointer']
    constraints: #[#[ #type, String],
        #[ #shape, String],
        #[ #diameter, Number],
        #[ #volume, Number],
        #[ #eventTime, DateTime],
        #[ #contourPointer, Contour],
        #[ #imagePointer, Image]].
```

`Microadenoma` is a subclass of `Object`. The `instVarNames` defines the attributes: `type`, `shape`, `diameter`, `volume`, `eventTime`, `contourPointer`, and `imagePointer` for object `Microadenoma`. The `constraints` defines the type of each attribute. For example, `volume` and `diameter` are of type `Number`.

`MicroadenomaSet` is an object class in which each instance is a `Microadenoma`, as shown in the following:

```
Set subclass 'MicroadenomaSet'
    constraints: Microadenoma.
```

Figure 3. Two-layered data model for modeling semantics of image data



The images and object contours are stored in the stacks. The semantics of the images are modeled in the upper layer.

If an image in the system contains a microadenoma, then there is an object instance in `MicroadenomaSet` corresponding to that microadenoma and representing its features. An object class is created for each type of identifiable object in all the images. Thus, such data structures provide a framework and guidelines to extract features from all images. The object instance maintains links that point to its contours and images. To reduce retrieval time, semantic information on the objects is kept in the corresponding object classes. The image and object contours are stored separately in stacks which can be invoked through object pointers for direct spatial query processing.

Our data model extends the conventional hierarchical object-oriented model by introducing additional new constructs to describe the spatial and evolutionary behavior of objects as described in the following sections.

3.2 Modeling Inter-Object Relations

3.2.1 Hierarchical Object Constructs.

1. *Aggregation* (Figure 4; Hull and King, 1987; Kim and Chou, 1988): An object is composed of several constituent objects that form an "Is-part-of" hierarchy. For example, the growth of a pituitary gland can be described in two stages (Goodrich and Lee, 1987). Originally, the pituitary gland is composed of the infundibular process and Rathke's pouch. As it matures, it is composed of pituitary stalk, neurohypophysis, cleft, pars distalis, pars intermedia, and pars tuberalis (Figures 2 and 10).

Figure 4. Hierarchical object constructs

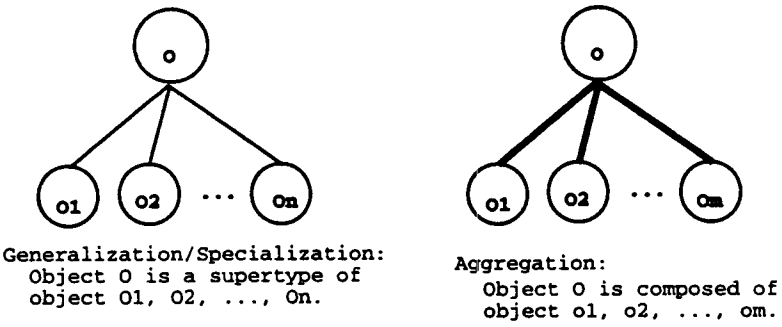
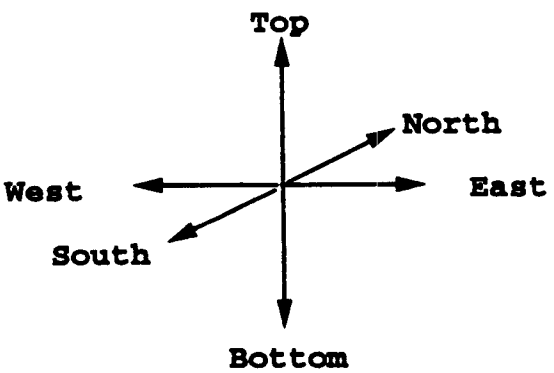


Figure 5. Directional convention for spatial objects



2. *Generalization/Specialization* (Hull and King, 1987): Objects form an "Is-a" hierarchy. The supertypes are a more generic representation of the objects, while the subtypes provide a more specialized representation. The subtypes inherit characteristics from their supertypes.

3.2.2 Spatial Constructs. Spatial object constructs are used to model the spatial relations among objects. Based on the anatomical knowledge and the physician's diagnosis, we model the primary spatial relations among image objects, which are the relations of an object with respect to its surrounding objects. The spatial relations include orthogonal and containment relations as follows:

Orthogonal relations describe the direction from one point to another:

```
East, South, West, North, Top, Bottom,
EastSouth, EastWest, EastNorth, EastTop, EastBottom,
SouthWest, SouthNorth, SouthTop, SouthBottom, WestNorth,
WestTop, WestBottom, NorthTop, NorthBottom, TopBottom,
```

```
EastSouthTop, EastSouthBottom, SouthWestTop, SouthWestBottom,
WestNorthTop, WestNorthBottom, NorthEastTop, NorthEastBottom
```

Containment relations describe the relative position and locations of contact among objects, including:

```
Separated, ExteriorContact, Connects,
Socketed, Invades, Contains, InteriorContact.
```

The spatial object constructs included in our model can be divided into three categories: *no containment, containment,* and *partial containment* as follows:

No containment

1. `Separated` (Figure 6*a*): This spatial construct is used to describe two objects that are separated from one another. The orthogonal relation is expressed inside a square between both objects. The Separated relation between a pair of objects is symmetric. For example, the pituitary gland lies in the back (East) of the sphenoid sinus, and is separated from it by the sellar floor (Figures 2 and 10).
2. `Connects`: One object connects with another at one or more locations. Connects is a symmetric relation that is expressed by two squares connected to each other (Figure 6*c*). The orthogonal relation between the corresponding objects is represented inside the square closer to O1. For example, the pituitary gland connects with the hypothalamus via a neural stalk (Figure 10).
3. `ExteriorContact`: Two objects touch each other at one or more locations. The ExteriorContact relation is symmetric and is modeled by two squares adjacent to each other as in Figure 6*b*. ExteriorContact is a significant event in the medical domain.

Containment

1. `Contains`: An object contains another object. Contains is asymmetric and is expressed by two squares with one containing the other, indicating that O1 contains O2. The orthogonal relation from O1 to O2 is indicated in the inside square (Figure 6*d*). For example, a microadenoma may develop inside the pituitary gland (Figure 10).
2. `InteriorContact`: An object touches the inside of the other. The orthogonal and containment relations are modeled by two squares with one touching the inside of the other (Figure 6*e*). For example, a microadenoma may lie in the inferolateral portion of the pituitary gland.

Partial Containment

1. `Invades`: This spatial construct is used to describe one object that invades another. The objects are represented by circles. The orthogonal relation is expressed inside the square closer to O1. Figure 6*g* indicates that O1 invades O2 in a West direction. The Invades relation between a pair of objects is asymmetric in the sense that O1 invades O2 and O2 is invaded by O1.

Figure 6. Spatial object constructs

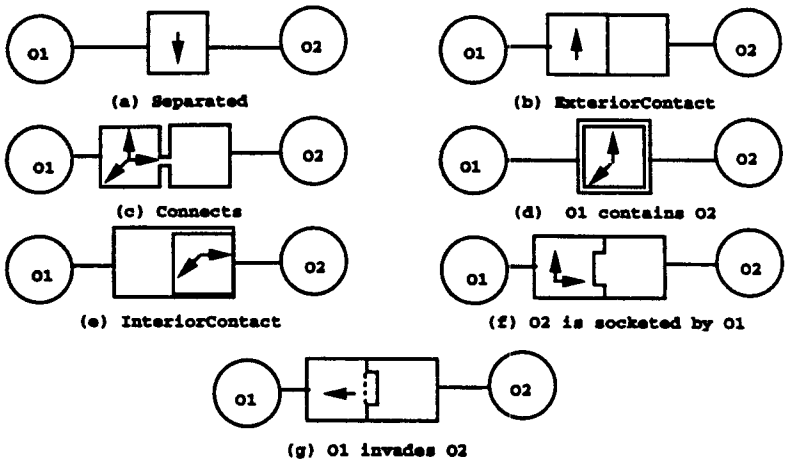
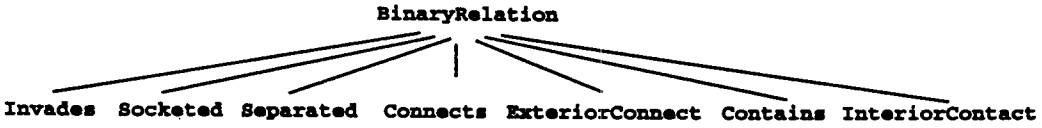


Figure 7. Spatial relation hierarchy



2. Socketed: To model one object encapsulating another, the orthogonal and containment relations need to be specified (Figure 6*f*) in the same way as the Invades relation. Socketed is also an asymmetric relation. For example, the pituitary gland lies in the sella turcica (Figure 10).

As shown in Figure 7, all the spatial relations in our system are subclasses of BinaryRelation. BinaryRelation has two attributes: source object and destination object as shown below:

```
Object subclass 'BinaryRelation'
    instVarNames: #['source,' 'destination']
    constraints: #[#[ #source, Object],
      #[ #destination, Object]].

BinaryRelation subclass 'Invades'
    instVarNames: #['orthogonalRelation']
```

```
constraints: #[#[ #orthogonalRelation, String]].
```

Invades is a subclass of the BinaryRelation with an additional attribute storing the orthogonal relation from source to destination. The point orthogonal relation is also asymmetric. For example, if O1 is on top of O2, then O2 is on the bottom of O1. The object class Invades stores only the orthogonal relation from source to destination. The orthogonal relation from destination to source is obtained by a method defined on the BinaryRelation class, which computes the inverse orthogonal relation given the source to destination relation. In our Gemstone implementation, the internal structure of the storage of relations and the computation of its inverse are transparent to the user. The user only sees two methods defined on the relation class: one returns the orthogonal relation from source to destination, and the other from destination to source. All other asymmetric spatial relations are handled in a similar way.

Notice that the orthogonal relation between a pair of objects can be determined by the centers of the masses, the centers of the bounding boxes of the corresponding objects, or by the interpretation of the domain expert.

An object class is defined in the system that collects similar object instances corresponding to each spatial construct in the data model. For example, a macroadenoma may invade the sphenoid sinus (Figure 10). The object class MacroadenomaInvades SphenoidSinus[3] is defined to store the invasion relation between the Macroadenoma and the Sphenoid Sinus as follows:

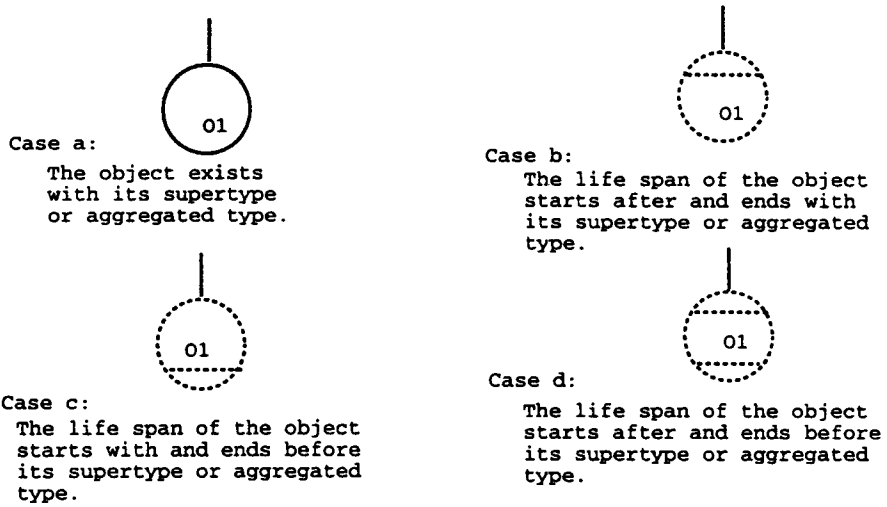
```
Set subclass 'MacroadenomaInvadesSphenoidSinus'
    constraints: Invades.
```

Each instance in this class represents an invasion relation between the corresponding pair of macroadenoma and sphenoid sinus. All the other spatial relations are defined and modeled in a similar manner.

3.2.3 Temporal relation object constructs. Now let us consider the modeling of inter-object temporal relations. The constructs that represent the temporal relations between an object and its supertype or its aggregated type are shown in Figure 8. These temporal relationships define how the characteristics of the supertypes are temporally inherited. *Temporal inheritance* deals with the way time-dependent

3. An alternative approach is to define a generic invasion class to represent the invasion relations among all the objects. This design decision affects system performance, depending on the number of object classes and the number of instances in the object classes in the applications. However, invasion among different pairs of objects also requires different features to be represented in the system. As a result, the generic invasion class needs to be specialized. The total number of object classes representing the spatial relationships among objects in the pituitary gland area in our system such as MacroadenomaInvadesSphenoidSinus is around 20. Since we capture more objects in hand images, and each object has many evolutionary stages (Tanner et al., 1975), the total number of object classes representing the spatial and evolutionary relationships for hand is around 180.

Figure 8. Temporal object constructs



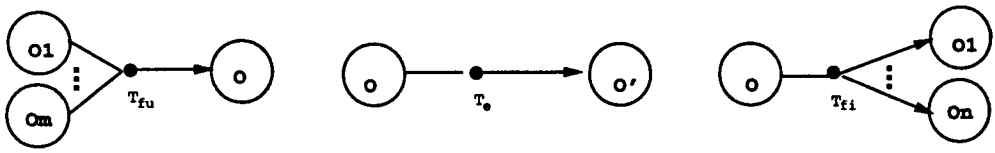
characteristics of a supertype are inherited by its subtypes. The general rule is that an object may only inherit characteristics from other objects that exist in its own space-time domain (Chu et al., 1992).

To illustrate the use of the temporal relation object constructs, we apply them to the model describing the growth of the pituitary gland (Figure 10). The pituitary gland is the result of two separate developmental processes that culminate in the formation of the adenohypophysis and the neurohypophysis. At the third week of gestation, the pituitary gland is composed of Rathke's pouch and the infundibular process (Figure 8*c*). By the end of the 20th week of gestation, the infundibular process develops into the neurohypophysis and the neural stalk (Figure 8*b*; Goodrich and Lee, 1987).

3.2.4 Evolutionary Object Constructs. The existing approach models versions of objects (Kim and Chou, 1988). We have developed a mechanism to model the fission and fusion of objects (Figure 9).

1. *Evolution*: The characteristics of an object may evolve with time. For example, a microadenoma may evolve into a macroadenoma over a period of time (Figure 10).
2. *Fusion*: An object may fuse with other objects to form a new object with different characteristics than either of the constituent objects.
3. *Fission*: An object may split into two or more independent objects. For example, the pituitary gland is composed of two parts: neurohypophysis and

Figure 9. Evolutionary object constructs

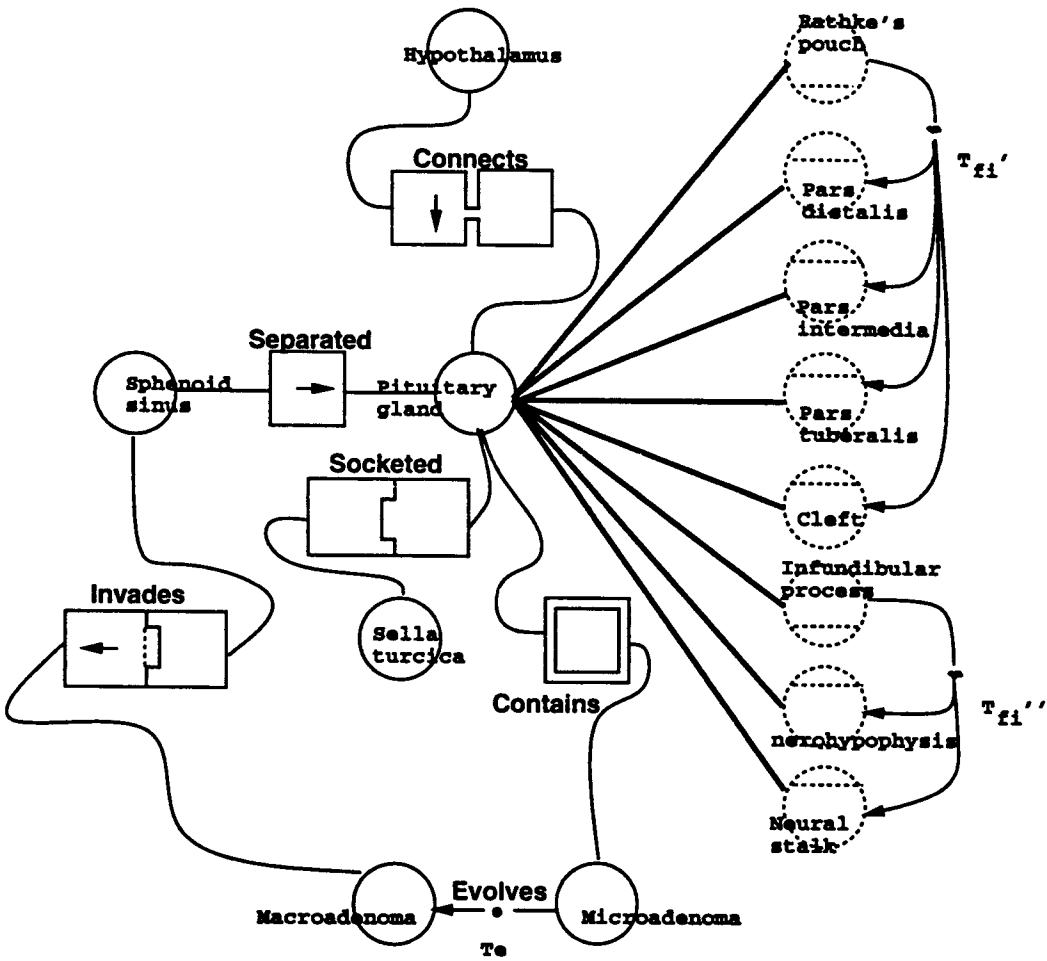


Fusion: Objects O1, ..., Om fuse into a new object at time T_{fu}

Evolution: Object O evolves into a new object O' at time Te

Fission: Object O splits into objects O1, ..., On at time T_{fi}

Figure 10. Growth of the pituitary gland



The evolution and the spatial relations of the adenomas with respect to their surrounding brain structures are modeled.

adenohypophysis. As shown in Figures 2 and 10, Rathke's pouch, the origin of the adenohypophysis, develops in the third week of gestation and splits into the pars distalis; the pars tuberalis; the pars intermedia; and the cleft.

As an example, adenomas in the pituitary gland are modeled with the hierarchical, temporal, evolutionary, and spatial object constructs as shown in Figure 10. The pituitary gland has long been considered the master endocrine organ. It lies at the base of the brain, *beneath* the hypothalamus, to which it is *connected* (Figure 6*c*) via the neural stalk, in a bony cavity known as the sella turcica (or sella). The pituitary gland lies *behind* (East; Figure 6*a*) the sphenoid sinus, as shown in Figures 2 and 10. It *fills* (Figure 6*f*) approximately three-fourths of the sellar cavity.

The symptoms of an adenoma depend on its position with respect to its surrounding brain structures. As shown in Figure 10, microadenomas develop *inside* (Figure 6*d*) the pituitary gland, and may cause endocrinological disorders. Microadenomas may also *evolve* (Figure 9) into infrasellar macroadenomas which *invade* (Figure 6*g*) the sphenoid sinus.

Based on the model constructs, we present a query language that can express the spatial evolutionary content of the images. The model shown in Figure 10 helps formulate the user queries in the following section.

4. Spatial Evolutionary Query Language

Conventional query languages lack the capability to query the spatial evolutionary nature of medical images (Snodgrass, 1987; Roussopoulos et al., 1988; Kim, 1989; Navathe and Ahmed, 1989). To remedy this, we propose SEQL, which operates on the spatial evolutionary domains of medical images. In addition to alphanumeric predicates, SEQL contains constructs to specify spatial, temporal, and evolutionary conditions. The spatial operators specify the spatial relations, the temporal operators specify the data at a specific point in time, and the evolutionary operators specify the evolutionary object sequences of interest. A SEQL query consists of the following seven optional clauses:

```
[CONTEXT a_view]
[CONSTRUCT a_view]
[WHERE clauses]
[WHICH clauses]
[WHEN clauses]
[SELECT clauses]
[Operations]
```

`CONTEXT` references the view created by the user. `CONSTRUCT` creates a view customized to the interests of the individual user. `WHERE` clauses describe the selection criteria using spatial and traditional arithmetic predicates. `WHICH` clauses describe the evolutionary processes among object types. `WHEN` clauses select the appropriate snapshot of the database. `SELECT` selects the desired data items. Operations specify

the system or user-defined operations, such as display, rotate, and superimpose on the selected data. The full text of the query language in Backus-Naur form is given in the appendix.

4.1 Construct Clauses

CONSTRUCT creates a database view customized to the interest of the individual user. The view can be specified by its intentional pattern and extensional pattern types. For example, to construct a view for patients of age 18 or older, we have:

```
CONSTRUCT
    ViewAdultPatient = ( Patient [ age >= 18 ])
```

We can refer to the above view by using the following clause:

```
CONTEXT ViewAdultPatient
```

4.2 Where Clauses

WHERE clauses restrict the extensional patterns that satisfy certain conditions. Spatial conditions and conventional alphanumeric predicates can be specified in WHERE clauses. Comparison conditions involving aggregation functions such as COUNT and AVG are also allowed. The alphanumeric predicates used in our model are similar to those in the relational model. For example, to select the Asian patients from the database, we have:

```
WHERE Patient ethnicGroup=''Asian''
```

The WHERE clauses allow the user to directly manipulate the spatial objects derived from the images in the system. Some constant spatial objects such as points, line segments, circles, spheres, and rectangles with certain sizes can also be constructed in the system to help formulate the query.

4.2.1 Symbolic Spatial Feature Predicate. Spatial feature queries involve the spatial characteristics of the image objects, while spatial relation queries deal with the orthogonal and containment relations.

By inspecting the object classes, the user knows what spatial features are available for querying and, hence, if the user wants to retrieve the images with a microadenoma of 5mm or larger in diameter, the user writes the following predicate:

```
WHERE Microadenoma diameter >= 5 mm
```

4.2.2 Symbolic Spatial Relation Predicate. The user can also query on the orthogonal relations among objects. For example, to retrieve images of the macroadenoma that is on top of the sphenoid sinus, the user writes:

```
WHERE Macroadenoma center ON_TOP_OF SphenoidSinus center
```

Querying can also be performed on the containment relations of the image objects. To retrieve the image of a macroadenoma that invades the sphenoid sinus, the user writes:

```
WHERE Macroadenoma INVADES SphenoidSinus
```

4.2.3 Direct Spatial Predicate. Direct spatial queries requiring point addressing capabilities that directly search for objects over a certain area of the image can also be described in SEQL. For example, to retrieve the images in which the macroadenoma extends within a 20 mm radius from the center of the hypothalamus, the user writes:

```
WHERE Macroadenoma INTERSECTS SPHERE( Hypothalamus center, 20 mm)
```

SPHERE(point, radius) is used to construct a sphere in the 3-D space from a certain point with a certain radius. The WHERE clauses may be combined with other clauses for querying database contents.

4.3 When Clauses

The WHEN clause selects the appropriate snapshot of the data of interest at a particular point in time. We now discuss the temporal operators.

Temporal functions manipulate time points: START_TIME, END_TIME, EVENT_TIME, and RECORD_TIME. They are the operators used in the query language to retrieve the start time, end time, event time, and record time of objects.

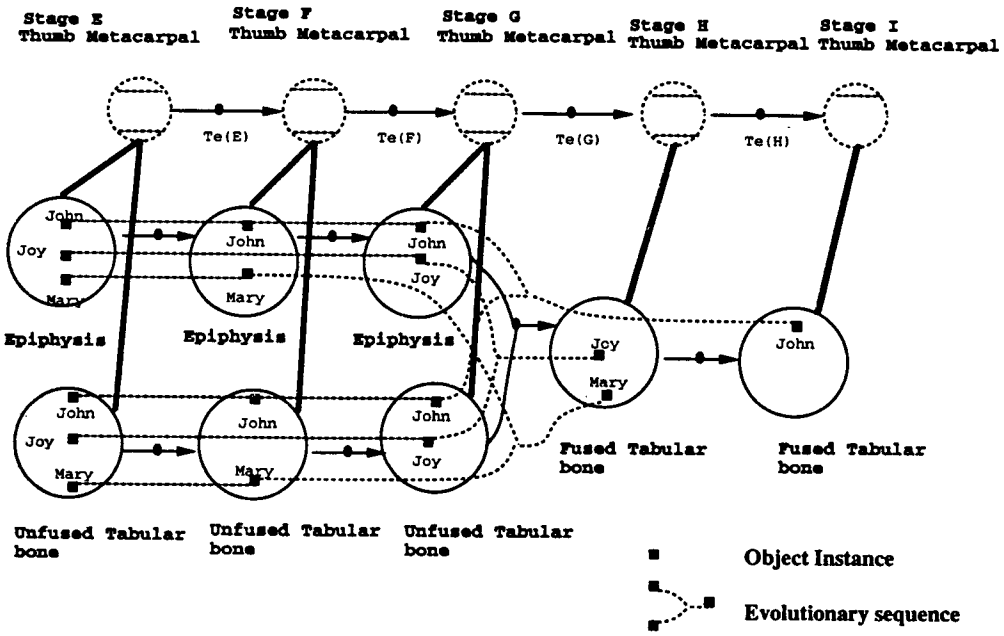
Temporal ordering functions. Temporal ordering of an object history sorts the object versions in ascending order, based on their time stamps so that retrieval of object versions in a specific order can be specified, including FIRST, LAST, N_th, PRIOR, and NEXT.

Temporal interval comparison operators. To specify a more complex temporal condition, the interval specified in the WHEN clause may be subjected to temporal interval comparison operators (Snodgrass, 1987; Navathe and Ahmed, 1989), such as PRECEDES, FOLLOWS, and DURING. These operators specify how the intervals following the WHEN clauses are related to some other time intervals. The time point comparison operators such as BEFORE and AFTER are also included.

4.4 Which Clauses

The WHICH clause describes various evolutionary processes on a set of evolving objects. The evolution of hand bones, such as the metacarpal, provides a good example on the use of evolutionary operators. The evolution of the thumb metacarpal can be

Figure 11. Fused tabular bone for the thumb metacarpal



Fusing the epiphysis and unfused tabular bone into a fused tabular bone.

described in eight stages from stage B to stage I (Figure 11).[4] The thumb metacarpal is fused from two pieces of bones, the epiphysis and the unfused tabular bone, into a fused tabular bone in Stages H and I. There are three kinds of evolutionary processes in general: evolution, fusion, and fission. A discussion of each evolutionary process follows.

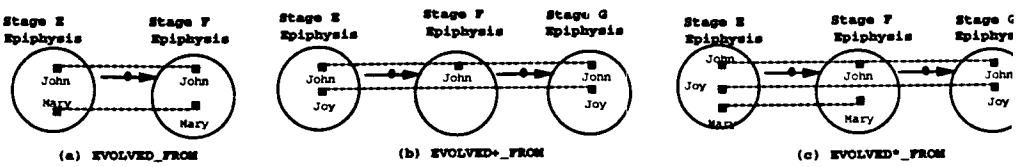
4.4.1 Evolution. The condition *object_type*$_1$ `EVOLVED_FROM` *object_type*$_2$ selects all single-step evolutionary sequences in which an object instance of *object_type*$_2$ evolves into an object instance of *object_type*$_1$. For example, applying the single-step evolution operator `EVOLVED_FROM` in the following `WHICH` clause

```
WHICH   ThumbMetacarpalEpiphysisStageF
        EVOLVED_FROM ThumbMetacarpalEpiphysisStageE
```

onto the objects in Figure 11 selects the evolutionary sequences for John and Mary

4. The development of hand bones goes through several stages. We model each stage as an object class to allow us retrieve and manipulate information at different stages. For more detailed discussions, the interested reader is referred to Tanner et al. (1975).

Figure 12. Evolution operators for thumb metacarpal



from stage E to stage F for the epiphysis of the thumb metacarpal as shown in Figure 12*a*.

The condition *object_type*$_1$ EVOLVED+_FROM *object_type*$_2$ selects all multiple step evolutionary sequences in which an object instance of starting *object_type*$_2$ evolves into another object instance of *object_type*$_1$ in the evolutionary net. For example, an X-ray taken of Patient Joy at age 8 is an object instance in stage E. The fact that she did not have another one age 12, is an object instance in stage G. Using the EVOLVED+_FROM in the following WHICH clause

```
WHICH   ThumbMetacarpalEpiphysisStageG
        EVOLVED+_FROM ThumbMetacarpalEpiphysisStageE
```

selects the evolutionary sequences for John and Joy on the evolutionary net from stage E to stage G as shown in Figure 12*b*.

The condition "*object_type*$_1$ EVOLVED*_FROM *object_type*$_2$" selects all single- and multiple-step evolutionary sequences in which an object instance evolves into another object instance in the evolutionary net where *object_type*$_2$ evolves into *object_type*$_1$ as follows:

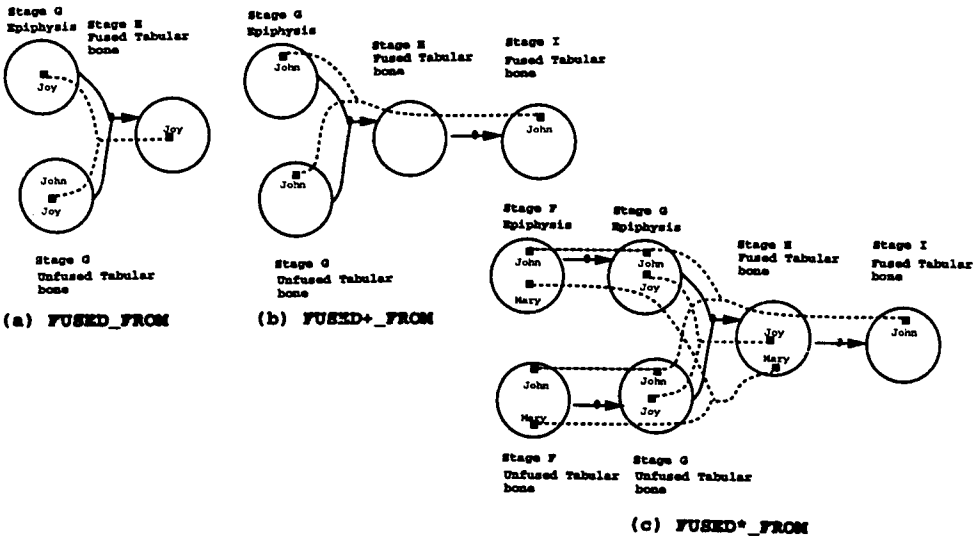
```
WHICH   ThumbMetacarpalEpiphysisStageG
        EVOLVED*_FROM ThumbMetacarpalEpiphysisStageE
```

selects the sequences for John, Mary, and Joy from stage E to stage G, as shown in Figure 12*c*.

4.4.2 Fusion. The operator FUSED_FROM selects the evolutionary sequences in an evolutionary net where a fusion process of several objects occurs. For example,

```
WHICH   ThumbMetacarpalFusedTabularBoneStageH
        FUSED_FROM
        ThumbMetacarpalEpiphysisStageG,
        ThumbMetacarpalUnfusedTabularBoneStageG
```

Figure 13. Fusion operators



Epiphysis and unfused tabular bone are fused into fused tabular bone for thumb metacarpal.

selects the single-step evolutionary sequence for Joy (Figure 13a).

We can define the multiple step fusion operators, FUSED+_FROM and FUSED*_FROM in the same manner as the multiple step evolution operators. For example,

```
WHICH   ThumbMetacarpalFusedTabularBoneStageI
        FUSED+_FROM
        ThumbMetacarpalEpiphysisStageG,
        ThumbMetacarpalUnfusedTabularBoneStageG
```

selects the multiple-step evolutionary sequence for John (Figure 13*b*).

The condition "*object_type* FUSED*_FROM *object_type*$_1$, *object_type*$_2$, ..., *object_type*$_k$" selects all the evolutionary sequences with a fusion process in the evolutionary net from *object_type*$_1$, *object_type*$_2$, ..., and *object_type*$_k$ to *object_type*. For example,

```
WHICH   ThumbMetacarpalFusedTabularBoneStageI
        FUSED*_FROM
        ThumbMetacarpalEpiphysisStageF,
        ThumbMetacarpalUnfusedTabularBoneStageF
```

selects the evolutionary sequences with a fusion for John, Mary, and Joy as shown in Figure 13*c*.

4.4.3 Fission. Similarly, there are three operators to describe the fission processes: SPLIT_FROM, SPLIT+_FROM, and SPLIT*_FROM that are used to select the evolutionary sequences where an object splits into several objects. The fission condition "*object_type*$_1$, *object_type*$_2$, ..., *object_type*$_k$ SPLIT_FROM *object_type*" selects all single-step evolutionary sequences with a fission in the evolutionary net from *object_type* to *object_type*$_1$, *object_type*$_2$, ..., and *object_type*$_k$. SPLIT+_FROM and SPLIT*_FROM are defined similarly.

4.5 Select Clauses

The SELECT clause identifies the attributes and object types that are to be operated on by the specified operations. It eliminates attributes and classes that are not relevant to the operations.

The operations in the Operations clauses can be either system-defined or user-defined data, image manipulation, or visualization operations (Chock et al., 1984), such as movie_loop, display, contour, rotate, and superimpose.

4.6 Sample Query

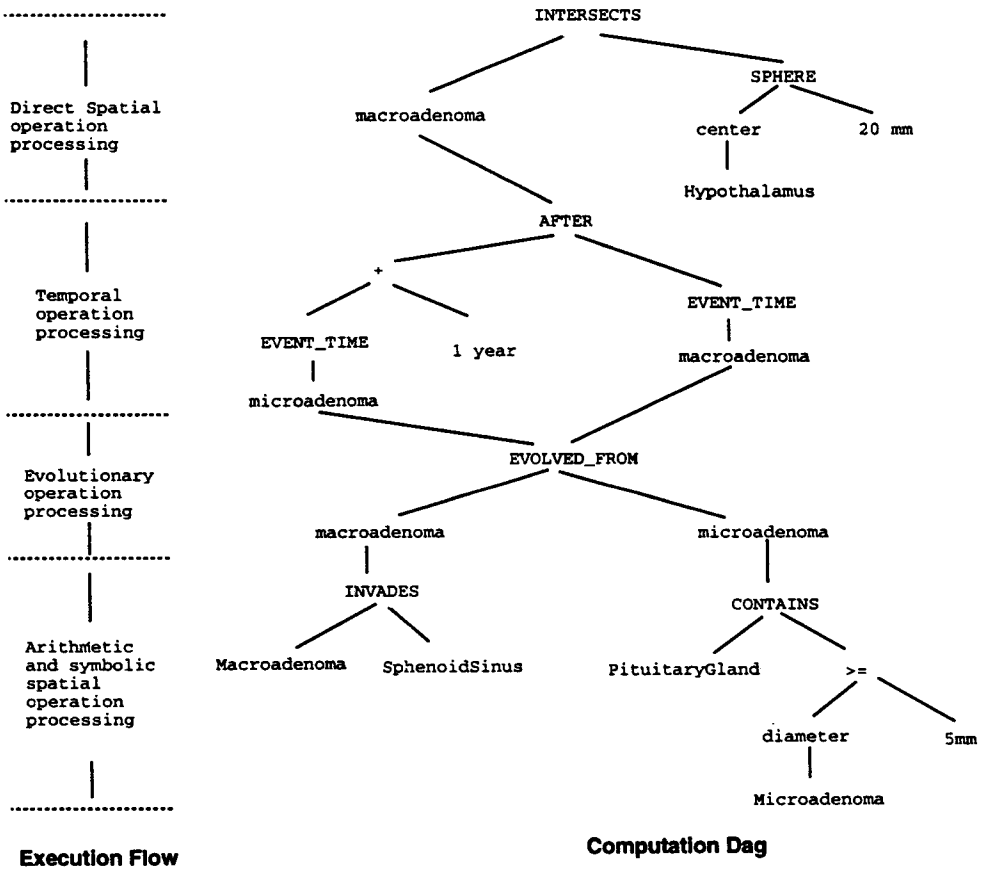
In this section, we present a sample query to illustrate the use of SEQL constructs to express certain clinical queries associated with radiographic findings in diagnostic images.

Sample query: *Retrieve the image frames showing a microadenoma 5 mm or larger in diameter which is developing inside the pituitary gland, which has evolved into a macroadenoma invading the sphenoid sinus, and which has extended within a 20 mm radius of the hypothalamus in one year's time.*

This query involves an evolving adenoma and its adjacent objects. Based on the object relations captured in the data model (Figure 10), as well as the object classes and their relations defined for each object, the sample query can be translated into the following SEQL query by using the AFTER, EVOLVED_FROM, CONTAINS, and INVADES operators:

```
WHEN     Microadenoma EVENT_TIME + 1 year AFTER
         Macroadenoma EVENT_TIME
WHICH    Macroadenoma EVOLVED_FROM Microadenoma
WHERE    PituitaryGland CONTAINS Microadenoma diameter >= 5 mm AND
         Macroadenoma INVADES SphenoidSinus AND
         Macroadenoma INTERSECTS SPHERE(Hypothalamus center,20 mm)
SELECT   Image
DISPLAY
```

Figure 14. Computation DAG for the sample query



5. Processing SEQL Queries

The major challenge of processing SEQL queries in large pictorial databases is maintaining efficiency while providing spatial and evolutionary querying capabilities. A SEQL query is first parsed and transformed into a computation directed acyclic graph (DAG; Fischer and LeBlanc, 1988) which represents the execution sequence of the query. The computation DAG for the sample query is shown in Figure 14.

The inner nodes in the DAG represent the operators of the query such as the `INTERSECTS`, `SPHERE`, `AFTER`, `EVOLVED_FROM`, `INVADES`, `CONTAINS`, and `>=`, while the leaf nodes represent the data input to those operators. The operators in the query are classified into four types: arithmetic and symbolic spatial operators, evolutionary operators, temporal operators, and direct spatial operators,

which correspond to their respective clauses in the query. There is a method defined for each operator in the SEQL query. Therefore, to generate the software code for the DAG, the query processor associates the operators in the DAG with their corresponding methods. The processing of the query starts from the terminal nodes and gradually moves upward.

5.1 Arithmetic and Symbolic Spatial Predicate Processing

Processing alphanumeric operators such as >= is made very straightforward by translating them into the corresponding operators in Gemstone. To process the symbolic spatial operators, a method performs table lookup through alphanumeric comparisons on the corresponding object classes that store the symbolic spatial information. For example, to search for the microadenoma 5mm or larger in diameter, the object class `MicroadenomaSet` (defined in Section 3) is searched, the diameter of each microadenoma instance in `MicroadenomaSet` is compared, and qualified object instances are retained for further processing. The operator `INVADES` is mapped to a search operation on object classes representing the invasion among objects, if the invasion relations exist. Otherwise, the operator is mapped to a direct spatial search on the image object. Therefore, to process the symbolic spatial relation `INVADES`, the object class `MacroadenomaInvadesSphenoidSinus` is searched. Since every instance in `MacroadenomaInvadesSphenoidSinus` represents a macroadenoma invading the sphenoid sinus, all of the instances are returned for evolutionary predicate processing.

5.2 Evolutionary Predicate Processing

A method is defined for each evolutionary operator. Each operator takes instances from two or more object classes as inputs, and returns the evolving object instances. For example, to locate a microadenoma evolving into a macroadenoma over time using the `EVOLVED_FROM` operator (Figure 14) after applying the arithmetic and symbolic spatial operators, the corresponding method will examine the qualified object instances in both object classes `Microadenoma` and `Macroadenoma` and verify whether there is an evolutionary link (object pointer) among them. Next, the qualified evolving object instances will be returned. For more discussion on the evolutionary query processing, the interested reader is referred to Chu et al (1992).

5.3 Temporal Predicate Processing

Once the evolutionary object instances are returned, temporal operators are applied to select the appropriate temporal objects and compare their time intervals. For example, in the sample query, the event time of the microadenoma and the macroadenoma are compared to determine whether the evolution took place in less than one year's time.

5.4 Direct Spatial Predicate Processing

The direct spatial search is performed through the manipulation of the image object at pixel level. We use a spatial indexing technique, the R^*-tree, to speed up the retrieval of image objects.[5] The R^*-tree (Beckmann et al., 1990) is an enhanced version of the R-tree. It is a page-based spatial indexing scheme that stores the bounding boxes of the image objects. An indexing tree is built for all the instances in an object class that require spatial indexing. At present, there are two R^*-trees in our system, one for hand images and one for brain images. We have more than 200 hand images and approximately 20 identified objects in each image. The total storage of the R^*-tree for hand images is less than 100 KB, with a utilization around 70%. Because we have fewer brain images stored in the system, the total storage of the R^*-tree for brain images is around 50 KB. The R^*-tree is implemented in C and is fully integrated with Gemstone.

Using the spatial indexing technique, we speed up the retrieval of image object candidates for further examination with direct spatial operators. Similar to the other operators, a method is defined for each direct spatial operator on the appropriate image object classes.

For example, to determine whether a macroadenoma is within a 20 mm radius from the hypothalamus, the center of the hypothalamus is retrieved, and a sphere with a 20 mm radius is created at the hypothalamus center. Then, all macroadenomas with a bounding box overlapping that of the sphere are retrieved. To confirm an intersection, a pixel-level operation determines whether any point on the boundary of the macroadenoma occupies the same position as the sphere.

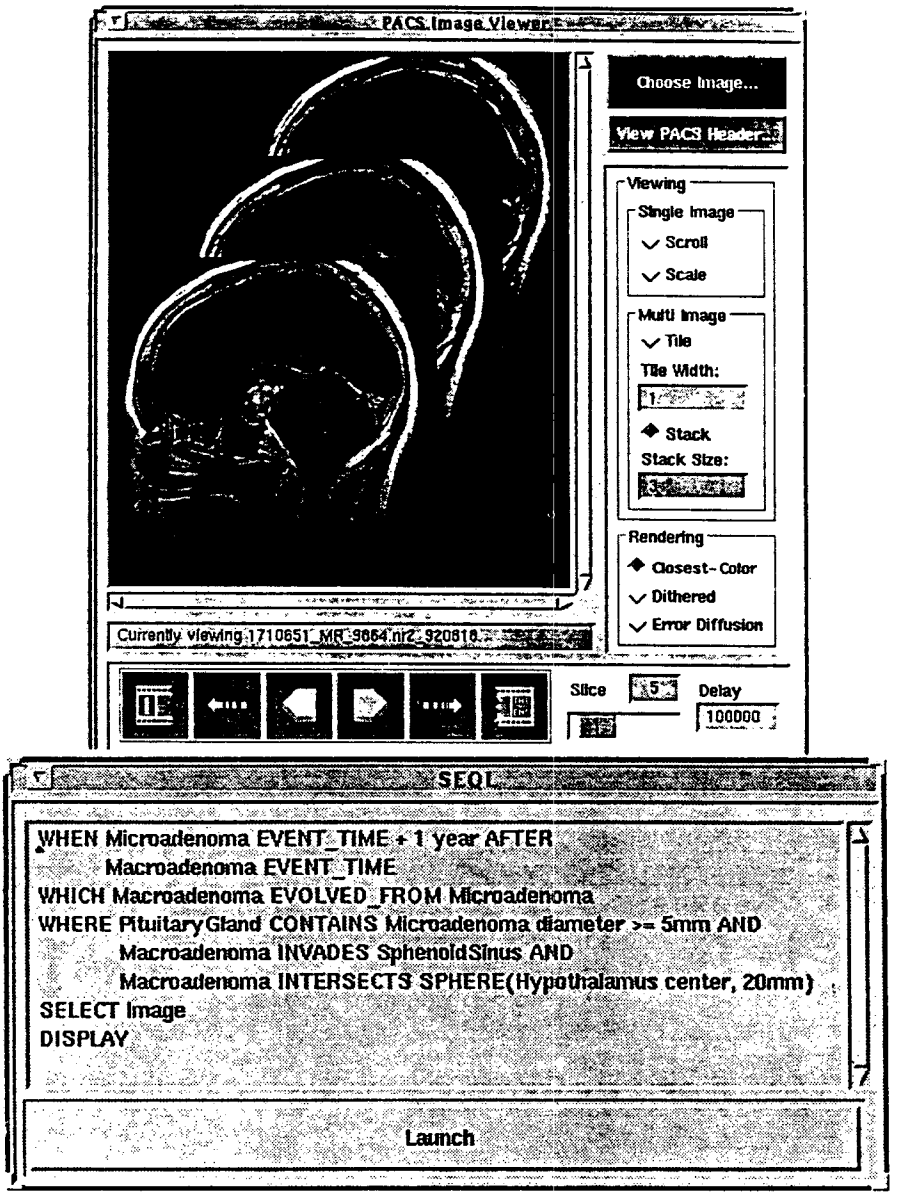
At the end of the query execution, all the images that satisfy the query predicates are returned. Figure 15 portrays the returned image frames (actual screen output) from our system after launching the sample query.

6. Implementation

A prototype image management system has been implemented to demonstrate the feasibility of the proposed approach. Since 1991, we have been building the prototype with a commercial object-oriented database, Gemstone, with VisualWorks as the data modeling and user interface development environment. Currently, the system runs on a Sparc 10 with 4 GB disk storage. Using VisualWorks, we have also developed a graphical interface, PICQUERY+ (Cardenas et al., 1993), with pull-down menu and "point and click" features, for querying the database in a user-friendly manner. PICQUERY+ uses a tabular user interface. It also employs a hierarchical window system and graphical representation of objects (e.g., icons).

5. If the image objects are not properly registered, then certain registration calibration of the images is needed to assure the accurate use of the R^*-tree.

Figure 15. Returned image frames



Macroadenoma invading sphenoid sinus for sample query

Buttons are included in the query window to perform such functions as schema and object display and query confirmation.

At present, the prototype system has a few hundred hand X-ray and MRI brain scan image studies. Feature extraction for hand X-rays was performed automatically for patients between the ages of 3 and 11 years (Pietka et al., 1991*b*). To extract features from brain MRI images, we use a commercially available MRI image segmentation and rendering system from ISG Corporation on many routine interventional brain procedures. Contours of objects showing good signal-to-noise ratios can often be automatically acquired using available minimum/maximum thresholding methods. The system also includes a semi-automatic region growing program after specification of an initial seed point, and manual contouring by hand outlining object boundaries. A comprehensive set of tools exists to edit (i.e., remove and splice) existing contours.

The time to process a query depends on the complexity of the query and the size of the database involved. In our system, all the images reside on the disk. The response time ranges from 2 seconds to 15 seconds, with an average of 5 seconds. Due to the small size of our current database, we did not observe significant improvement on response time when using the R^*-tree, as compared to sequential scanning. The display of images requires an additional 5 to 30 seconds, depending on the size and number of images.

The response time for a much larger system will depend on the size of the database and the query optimization in use. The scalability of our system is still under investigation.

7. Future Work

We plan to apply the proposed approach to model breast cancer and to manage mamogram databases. Our approach can also be used to represent evolutionary spatial objects in geographical information system (GIS) applications.

Further, medical queries are often conceptual and expressed in imprecise medical terms, for example, "retrieve the images demonstrating a brain tumor one gyrus away from the central sulcus." Current query processing accepts only precisely specified queries and only provides exact answers. This requires users to fully understand the problem domain and the database contents. The system returns null information if the exact answer is not available. To remedy these shortcomings, it is necessary to employ cooperative query answering (Chu and Chen, 1992), which provides approximate and associative answers that are relevant to the original query, even though not explicitly asked for by the user.

8. Conclusion

In this article, a semantic data model was introduced to capture the hierarchical, spatial, temporal, and evolutionary semantics of the images. The model mimics the

user's conceptual view of the image content, providing the framework and guidelines for image preprocessing to extract the features and relations of image objects. Based on the model constructs, a new query language, SEQL, is presented to express spatial evolutionary queries, providing direct manipulation capabilities of image objects. With semantic information captured in the model, spatial evolutionary queries are answered efficiently. The model constructs proposed are general, and also can be applied to other domains such as breast cancer modeling and geographical information systems.

A prototype image management system has been implemented at UCLA to demonstrate the feasibility of the proposed approach. The system has been built with a commercial object-oriented database, Gemstone, with ObjectWorks/VisualWorks as the data modeling and user interface development environment. The system runs on Sun Sparc 10 workstations. Our preliminary experience suggests that the proposed approach can be a feasible and effective way to retrieve images by content for large pictorial databases.

Acknowledgments

This work was supported by the NSF Scientific Database Initiative under grant numbers IRI9116849 and IRI9224559. The authors wish to thank the reviewers for their comments and suggestions, which helped improve the presentation of this article.

References

Angel, E. *Computer Graphics,* Reading, MA: Addison-Wesley, 1990.

Bart, M., Romeny, t.H., Florack, L., Koenderink, J., and Viergever, M. Scale space: Its natural operators and differential invariants. *Proceedings of the Twelfth International Conference of the Information Processing in Medical Imaging,* Berlin, Germany, 1991.

Beckmann, N., Kriegel, H., Scneider, R., and Seeger, B. The R^*-tree: An efficient and robust access method for points and rectangles. *Proceedings of the ACM SIGMOD,* Atlantic City, NJ, 1990.

Brummer, M., Van Est, A., and Menhardt, W. The accuracy of volume measurements from MR imaging data. *Proceedings of the Eighth Annual Meeting of the SMRM,* Amsterdam, 1989.

Cardenas, A.F., Ieong, I.T., Taira, R.K., Barker, R., and Breat, C.M. The knowledge-based object-oriented PICQUERY+ language. *IEEE Transactions on Knowledge and Data Engineering,* 5(4):644-656, 1993.

Chang, S.K., Shi, Q., and Yan, C. Iconic indexing by 2-D strings. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 9(3):413-428, 1987.

Chock, M., et al.?? Database structure and manipulation capabilities of a picture database management system (PICDMS). *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 6(4):484-492, 1984.

Chu, W.W. and Chen, Q. Neighborhood and associative query answering. *Journal of Intelligent Information Systems,* 1:355-382, 1992.

Chu, W.W., Ieong, I.T., Taira, R.K., and Breant, C.M. A temporal evolutionary object-oriented data model and its query language for medical image management. *Proceedings of the Eighteenth VLDB,* Vancouver, BC, 1992.

Daubechies, I. Orthonormal bases of compactly supported wavelets. *Communications of Pure and Applied Mathematics,* XLI:909, 1988.

Fischer, C.N. and LeBlanc, R.J., Jr. *Crafting a Compiler.* Menlo Park, CA: The Benjamin/Cummings Publishing Company, Inc. 1988.

Goodrich, I. and Lee, K. *The Pituitary: Clinical Aspects of Normal and Abnormal Function.* Amsterdam: Elsevier Science Publishers B.V., 1987.

Grosky, W.I. and Mehrotra, R. Index-based object recognition in pictorial data management. *Computer Vision, Graphics, and Image Processing,* 52(3):416-436, 1990.

Gupta, A., Weymouth, T., and Jain, R. Semantic queries with pictures: The VIMSYS model. *Proceedings of the Seventeenth VLDB,* Barcelona, Spain, 1991.

Huang, H.K., Mankovich, N.J., and Taira, R.K. Picture archiving and communication systems (PACS) for radiological images: State of the art. *CRC Critical Reviews in Diagnostic Imaging,* 28(4):383-427, 1988.

Hull, R. and King, R. Semantic database modeling: Survey, applications, and research issues. *ACM Computing Survey,* 19(3):201-261, 1987.

Kim, W. A model of queries for object-oriented databases. *Proceedings of the Fifteenth VLDB,* Amsterdam, the Netherlands, 1989.

Kim, W. and Chou, H.T. Versions of schema for object-oriented databases. *Proceedings of the Fourteenth VLDB,* Los Angeles, CA, 1988.

Mantyla, M. *An Introduction to Solid Modeling.* Rockville, MD: Computer Science Press, 1988.

Marr, D. and Hildreth, E. Theory of edge detection. *Proceedings of the Royal Society,* 1980.

Mohan, L. and Kashyap, R.L. An object-oriented knowledge representation for spatial information. *IEEE Transactions on Software Engineering,* 14(5):675-681, 1988.

Navathe, S.B. and Ahmed, R. A temporal relational model and a query language. *International Journal of Information Science,* 48(2):57-73, 1989.

Orenstein, J.A. and Manola, F.A. PROBE spatial data modeling and query processing in an image database application. *IEEE Transactions on Software Engineering,* 14(5):611-629, 1988.

Pietka, E., Kaabi, L., Kuo, M.L., and Hunag, M.K. Feature extraction in carpal bone analysis. *IEEE Computer Graphics and Applications,* 12(1):44-49, 1991*a*.

Pietka, E., McNitt-Gray, M.F., Kuo, M.L., and Huang, H.K. Computer assisted phalangeal analysis in skeletal age assessment. *IEEE Transactions on Medical Imaging,* 10(4):616-620, 1991*b*.

Rabitti, F. and Savino, P. Image query processing based on multi-level signatures. *Proceedings of ACM Information Retrieval,* Chicago, IL, 1991.

Rabitti, F. and Savino, P. An information retrieval approach for image databases. *Proceedings of the Eighteenth VLDB Conference,* Vancouver, Canada, 1992.

Roussopoulos, N., Faloutsos, C., and Sellis, T. An efficient pictorial database system for PSQL. *IEEE Transactions on Software Engineering,* 14(5):634-650, 1988.

Snodgrass, R. The temporal query language for TQUEL. *ACM Transactions on Database Systems,* 1987.

Taira, R. and Huang, H. A picture archiving and communication system module for radiology. *Computer Methods and Programs in Biomedicine,* 30:229-237, 1989.

Tanner, J.M., Whitehouse, R.H., Marshall, W.A., Healy, M.J.R., and Goldstein, H. *Assessment of Skeletal Maturity and Prediction of Adult Height (TW2 Method).* London: Academic Press, 1975.

Ullman, J.D. *Principles of Database and Knowledge-Base Systems,* Vol. 1, Rockville, MD: Computer Science Press, 1988.

Appendix

Formal syntax of the SEQL in BNF:

```
query_block ::= [CONTEXT view_variable]
                [CONSTRUCT_view]
                [WHERE_clauses]
                [WHICH_clauses]
                [WHEN_clauses]
                [SELECT_clauses]
                [Operations]

CONSTRUCT_view ::= CONSTRUCT [view_variable =] subdatabase
subdatabase ::= ( object_and_its_attributes
   | [more_object_and_attributes] [subdatabase] )
more_object_and_attributes ::= more_object_and_attributes
   object_and_its_attributes | object_and_its_attributes
object_and_its_attributes ::= object_name[ '[' attributes ']' ]
attributes ::= [attributes,] selected_attributes
   | [attributes,] attribute_predicates
selected_attributes ::= attribute_name
   | selected_attributes, attribute_name
attribute_predicates ::= attribute_term
   | attribute_predicate OR attribute_term
attribute_term ::= attribute_term AND attribute_fac | attribute_fac
```

```
attribute_fac ::= [NOT] attribute_prim
attribute_prime ::= attribute_name comp_op primitive_constant

WHERE_clause ::= WHERE boolean1
boolean1 ::= bool_term1 | boolean1 OR bool_term1
bool_term1 ::= bool_term1 AND bool_fac1 | bool_fac1
bool_fac1 ::= [NOT] bool_prim1
bool_prim1 ::= spatial_predicates | arithematic_predicate | boolean1
arithematic_predicate ::= exp1 comp_op exp1
exp1 ::= arith_term | exp1 add_op arith_term
arith_term ::= arith_fac | arith_term mult_op arith_fac
arith_fac ::= [add_op] primary1
primary1 ::= numeric_constant | objects_and_attributes
    | aggregated_terms
comp_op ::= > | < | >= | <= | = |  !=
add_op ::= + | -
mult_op ::= * | /
spatial_predicates ::= spatial_objects
   spatial_relations spatial_objects
spatial_objects ::= objects | points | lines
    | spheres | rectangles
point ::= '(' numeric_constant ',' numeric_constant
    ','numeric_constant ')'
lines ::= LINE'(' point ',' point ')'
spheres ::= SPHERE '(' point', numeric_constant ')'
rectangles ::= RECTANGLE '(' point ',' point ')'
spatial_relations ::= directional_relation | distance_relation
    | containment_relation
directional_relation ::= 'ON_'direction'_OF'
direction ::= single_direction | double_direction | triple_direction
single_direction ::= EAST | SOUTH | WEST | NORTH | TOP | BOTTOM
double_direction ::= SOUTH_EAST | EAST_NORTH | EAST_TOP | EAST_BOTTOM
        | SOUTH_WEST | SOUTH_TOP | SOUTH_BOTTOM
        | NORTH_WEST | WEST_TOP | WEST_BOTTOM
        | NORTH_TOP | NORTH_BOTTOM
triple_direction ::= SOUTH_EAST_TOP | EAST_NORTH_TOP | SOUTH_WEST_TOP
    | NORTH_WEST_TOP | SOUTH_EAST_BOTTOM | EAST_NORTH_BOTTOM
    | SOUTH_WEST_BOTTOM | NORTH_WEST_BOTTOM
distance_relation ::= IS A DISTANCE OF numeric_constant FROM
containment_relation ::= IS passive_relation BY | active_relation
passive_relation ::= SEPARATED | SOCKETED | CONTAINED | INVADED
active_relation ::= EXTERIOR_CONTACTS | INTERIOR_CONTACTS
    | CONNECTS WITH | CONTAINS | INVADES | INTERSECTS
```

```
WHICH_clause ::= WHICH boolean2
boolean2 ::= bool_term2 | boolean2 OR bool_term2
bool_term2 ::= bool_term2 AND bool_fac2 | bool_fac2
bool_fac2 ::= [NOT] bool_prim2
bool_prim2 ::= boolean2 | evolutionary_predicate
evolutionary_predicate ::= objects evolutionary_operator objects
evolutionary_operator= EVOLVED_INTO | EVOLVED*_INTO | EVOLVED+_INTO
        | FUSED_FROM | FUSED*_FROM | FUSED+_FROM
        | SPLIT_INTO | SPLIT*_INTO | SPLIT+_INTO

WHEN_clause ::= WHEN boolean3
boolean3 ::= bool_term3 | boolean3 OR bool_term3
bool_term3 ::= bool_fac3 | bool_term3 AND bool_fac3
bool_fac3 ::= [NOT] bool_prim3
bool_prim3 ::= pred3 | boolean3
pred3 ::= exp3 tem_comp_op exp3
exp3 ::= [temporal_sequence] objects_and_attributes
    | temporal_constant
temporal_constant ::= time_point
    | '['time_point, time_point']' | numeric_constant
time_point ::= time_factor add_op temporal_constant
    time_granularity
time_factor:= objects_and_attributes | NOW
    | temporal_constant time_granularity
tem_comp_op ::= DURING | OVERLAPS | MEETS | EQUIVALENT
    | ADJACENT | FOLLOWS | PRECEDES | BEFORE | AFTER
time_granularity := years | months | weeks | days
    | hours | minutes | seconds
temporal_sequence ::= FIRST [N] | LAST [N] | Nth [N] | PRIOR | NEXT
time_point_functions ::= START_TIME | END_TIME | EVENT_TIME
    | RECORD_TIME

SELECT_clauses ::= SELECT terms
terms ::= [terms,] objects_and_attributes
    | [terms,] aggregated_terms
aggregated_terms ::= aggregation_function(objects_and_attributes)
aggregation_function ::= COUNT | MAX | MIN | AVG | SUM
objects_and_attributes ::= query_block | objects | attributes
objects ::= object | objects, object
attributes ::= attribute | attributes, attribute

Operations ::= image_operator['(' objects_and_attributes')']
    [parameter_list]
```

```
image_operator ::= PAN | ROTATE | ZOOM | SUPERIMPOSE | MASK
    | COLOR TRANSFORMATION | PROJECT | THRESHOLDING
    | BOUNDARY | MOVIE_LOOP | DISPLAY | user_defined_function
```