Concepts and System Architectures for the Management of Very Large Spatial Raster Objects in a Database Framework

Stephan Nebiker[1] and Lukas Relly[2]

[1]Institute of Geodesy and Photogrammetry
[2]Institute of Information Systems
Swiss Federal Institute of Technology Zurich
Switzerland

Abstract

The efficient management of very large spatial raster objects is becoming an important new feature of Geo-Information Systems. This paper investigates and summarises the main requirements that must be fulfilled by a spatial raster management solution. The investigations primarily focus on the management of very large raster mosaics, as a typical example for future requirements, both in terms of data volume and functionality. The aspects investigated include spatial objects access, spatial partitioning and partition indexing, multi-resolution, georeferencing and storage management.

The paper then presents two system architectures which approach the problem at different levels of abstraction. The first architecture, GrIdS, is a DBMS application which investigates spatial raster management concepts and techniques available at a middleware layer. The paper discusses some of the key features of the GrIdS project, including a tile-based multi-resolution concept for very large raster mosaics. The section on GrIdS is concluded by the presentation of results which demonstrate the capabilities and limitations of this approach. CONCERT, the second architecture presented, enables the investigation of extensible database concepts and techniques supporting the efficient management of large objects, in particular spatial raster objects.

Keywords

raster, DBMS, GIS, storage management, extensible DBMS

1 Introduction

The emergence of high-resolution remote sensors and the establishment of digital photogrammetry are leading to a rapid increase in the amount of spatial raster data being acquired. In addition to this large-volume raster imagery, a number of other spatial raster types, such as raster maps, regular matrix DTMs and thematic rasters or 'grids' have become an important information source for GIS. The integrated management of these raster types, which will subsequently be referred to as 'spatial raster data', is one of the major challenges for future GISs and their underlying DBMS technology.

The main characteristics of spatial raster data can be described as 'large to very large objects with spatial characteristics'. There are DBMS-based data management solutions for large objects on the one side and spatial objects on the other, e.g. video databases or vector-based spatial database extensions. However, there are hardly any solutions addressing and exploiting both characteristics simultaneously. Yet, DBMS support is the key to the integration of spatial raster data with other spatial and non-spatial information as well as the basis for ‘scaling to large numbers of users and large volumes of data’ [Sto96].

In this paper we first discuss a number of important requirements in the design of a spatial raster management solution. We then present two architectures and prototype systems which enable the investigation and development of concepts on different levels of abstraction. In Section 3, we present the GrIdS project and architecture, which represents a so-called middleware approach (see Fig. 1a). This project focuses on the investigation of spatial raster management concepts which can be integrated with existing state-of-the art GIS and DBMS technologies. In Section 4, we present the CONCERT project and architecture (see Fig. 1b), which investigates new DBMS server concepts and technologies supporting spatial raster data.

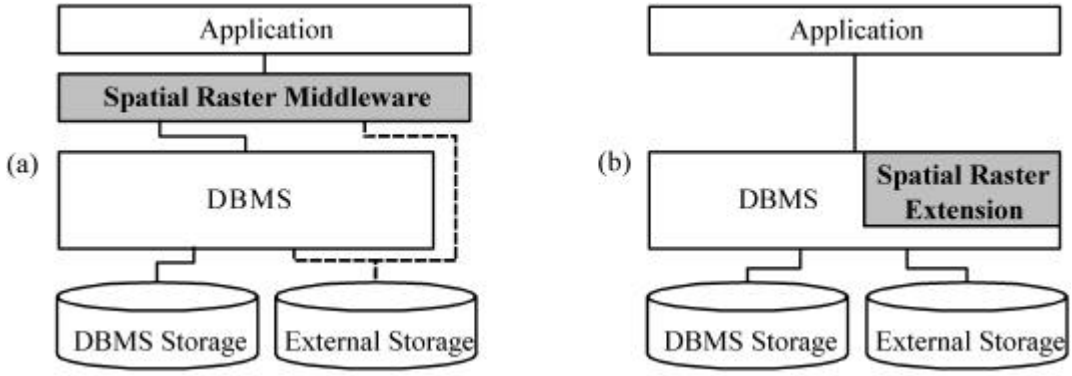

Figure 1 Middleware-based spatial raster management architecture (a) vs. spatial raster management architecture based on DBMS server extension (b).

A specific case study was chosen to allow the two different approaches and solutions to be compared. The scenario of *raster mosaic management* represents a typical example for future data management requirements, both in terms of data volume and functionality. Raster mosaics consist of lateral combinations of multiple raster objects into 'seamless' raster databases. In the past, raster mosaics were limited in size to a few GB, mainly due to the unavailability of suitable data management solutions. However, very large raster mosaics are likely to play a key role in future commercial and public spatial information servers.

2 Spatial Raster Management Requirements

The nature of spatial raster data requires a number of aspects to be considered when designing a management solution which should support this data type. This section provides an overview and a brief discussion of some of the key requirements.

Spatial Object Access

A basic requirement in the management of spatial raster objects is the support for spatial object access. This involves the efficient processing of spatial queries on collections of raster objects with arbitrary position and extent. For this type of spatial queries, raster objects can be substituted by vector objects, e.g. by bounding polygons or minimum bounding rectangles (MBRs). Querying, finding and retrieving such spatial objects is a standard feature in any GIS. A comprehensive overview of existing spatial access methods is provided in [GG95]. Some of these methods have been incorporated in commercial spatial DBMS-extensions, which can be used for implementing this type of spatial access.

Spatial Partitioning and Partition Indexing

The size of spatial raster objects and in particular raster mosaics usually exceeds the amount of primary memory available on a computer system. Typically, these objects are accessed only partially by means of spatial queries and they are only rarely manipulated as a whole. A spatial raster data management solution should therefore support the transparent spatial partitioning and reconstruction of large raster objects. The efficient access to partitions or to individual pixels or cells of large spatial raster objects requires the use of specialised spatial access methods. Several of the methods described in [GG95] are also suitable for this type of intra-object or object partition access. An interesting approach combining the aspects of raster partitioning, spatial access and compression is presented in [PW96].

Multi-Resolution Support

Multi-resolution support is an important feature in spatial raster management. The determining factors which direct but also limit the use of multi-resolution concepts are:

- high costs of operations on full-resolution raster data
- potential access restrictions to the full-resolution data (e.g. due to commercial reasons)

Existing multi-resolution techniques which are applicable to raster data include resolution or image pyramids [Sha94], interlacing schemes, spatial multi-resolution encoding [Sam89] and simple concepts such as previews or 'thumbnails'. These concepts are well suited for individual raster objects with limited dimensions and size and they are often part of standard raster formats or compression schemes. However, they are not easily applicable to raster mosaics or seamless raster databases.

Spatial Raster Attributes

Descriptive attributes play an important role in the management of spatial raster data. Generic raster attributes include information on raster type, radiometric resolution, geometric resolution, object identification, description, etc. Data-type specific attribute information for spatial raster data includes:

- layer or channel information (e.g. topic or spectral band)
- value look-up tables and/or colour palettes
- information on series of raster objects (e.g. cartographic map sheet series)
- georeference information

Georeferencing or georegistration is a concept generally applicable to spatial data. However, it is particularly important in conjunction with raster data as it is the prerequisite for relating raster coordinates to real-world coordinates and for enabling the above-mentioned spatial access to individual raster objects. The major objective of a modern georeferencing concept should be its global usability. This requires the support for different spatial reference systems and the adherence to an international standard (see Section 3.1).

It should be noted that spatial raster data is predominantly accessed via its attribute information. This information consumes little space and should be clustered for optimal performance. The actual raster data, on the opposite, is generally large and variable in size. As a consequence, attribute and raster data should be stored separately, irrespective of the storage solution.

Raster Storage

Current DBMSs do not offer specific data types for raster data. However, there are two generic concepts for modelling and storing raster data in a DBMS environment: Binary Large Objects (BLOBs) and Abstract Data Types (ADTs). Their main characteristics are highlighted below:

- *BLOBs* offer a very generic representation of raster data in the form of variable-length byte strings. The concept is not bound to a specific database technology and has been supported by commercial DBMS products for several years. Fortunately, these mostly proprietary solutions will be standardised with the introduction of SQL3, where BLOBs will be defined as a new base type. Despite this standardisation effort, BLOBs will remain a data type with limited operations, which are restricted to the insertion, updating, selection and deletion of entire BLOBs or parts thereof.
- *Abstract Data Types* provide a great modelling power and flexibility which makes them particularly suitable for the representation of spatial data and of other complex data such as multimedia objects. With raster data, the main advantage of using ADTs over BLOBS is not so much the superior modelling power but much rather the possibility of supporting user-defined operations.

Storage Management

A storage management concept supporting spatial raster data needs to address two contradicting issues: *very high storage capacities* and *high performance*. Large spatial raster databases require storage capacities in the range of hundreds of gigabytes to terabytes. Today, these capacities can only be provided economically by collections of tertiary storage media such as magnetic tapes and optical disks. Spatial raster databases with simultaneous multi-user access additionally require data transfer bandwidths exceeding the I/O performance of individual hard disks.

Support for high-performance secondary storage subsystems (e.g. RAID) has been added to most commercial DBMSs. However, mass storage support in the form of near-line tertiary storage

subsystems (e.g. optical or tape jukeboxes) or even hierarchical storage solutions is only gradually becoming available. For middleware-based spatial raster management solutions (see below), application-controlled mass storage management offers an alternative solution. However, in this approach, several database services such as transaction, integrity and security control are no longer available for the actual raster data.

3 GrIdS - A Spatial Raster Middleware Architectu re

GrIdS (Geospatial Raster and Image Database management System) is a prototype system developed by the GIS and photogrammetry groups at the ETH Zurich. The objectives of the GrIdS project include the design and development of a spatial raster management prototype system on the basis of existing (object-)relational DBMS technology. This platform enables the implementation and evaluation of different concepts and techniques for the DBMS-based management of spatial raster data. GrIdS should particularly demonstrate the capabilities and limitations of a *DBMS application* complementing the efforts of the CONCERT project (see Section 4) which focuses on *DBMS server extensions*.

3.1 Main Features of the GrIdS Architecture

The architecture of GrIdS is divided into the following three sub-systems:

- client: spatial raster management application and user interface (primarily written in Tcl/Tk)
- middleware: spatial raster middleware (written in C and Tcl/Tk)
- server: (object-) relational DBMS (Oracle Universal Server)

The investigations and the development effort of the GrIdS project are currently focussing on the middleware subsystem. The client sub-system is relatively light-weight, which favours its future implementation as an Internet application. The middleware - server communication is based on the CLI (Call-Level Interface) mechanism, which allows the sub-systems to be operated in a local or distributed computing environment.

Tile-based Partitioning

A partitioning concept based on regular, square tiles has been chosen. This concept simplifies the process of decomposing the raster space and its reverse process. In particular, it enables the use of simple and efficient spatial access methods (see below). A potential drawback of regular tiles with fixed dimensions is incurred by the variable size of compressed tiles. This variability in objects size could introduce inefficiencies into the secondary storage management.

Tile Indexing based on Morton Ordering

The Morton ordering [OV96, Sam89] in its N-ordering variant has been selected as a spatial indexing and access mechanism for the potentially very large number of mosaic tiles[1]. The Morton ordering is ideally suited for indexing regular partitions and it provides an inexpensive code generation and evaluation by bit-interleaving the row and column positions of a tile. It supports direct spatial access as well as spatial range queries. It also results in the lowest oversearch for range queries in comparison to all other major space orderings [AM90]. The linear Morton codes can be stored and indexed in any DBMS by using standard data types and traditional built-in access methods (e.g. b-tree). A standard 32-bit Morton coding scheme is capable of indexing raster mosaics holding 2'500 terapixels[2] or more – depending on the selected tile dimensions [Neb97]. This exceeds the size of any raster mosaic conceivable today.

---

[1] 1-meter resolution raster mosaic for area of Switzerland (40'000km2) with 512x512-pixel tiles ~ 150'000 tiles

[2] Max. mosaic size with 32-bit Morton codes (4 Giga codes) and 768x768 pixel tiles = $2.5 * 10^{15}$ pixels (equivalent to an area of 50'331 km x 50'331 km at 1 meter pixel resolution)

Spatial Partition Queries and Query Optimisation

Raster mosaics are typically accessed by means of spatial range queries which are defined by rectangular search windows. These range queries are used to fetch candidate tiles which are then subjected to further tests. The Morton ordering has the property that any tiles intersecting the search window also lie within the code range which is bounded by the codes of the tiles holding the upper left corner (minimum code) and the lower right corner (maximum code). Depending on the location and dimension of the query window, this simple range query concept can result in a considerable oversearch. An effective query optimisation procedure evaluates and splits the search window along major sub-quadrant boundaries [AS84]. It then substitutes the code range of the original query by multiple ranges. This approach was incorporated in GrIdS and it is shown in [Neb97] that this optimisation step dramatically reduces the number of excess candidate objects returned by the region query.

Multi-Resolution Concept for Raster Mosaics

The GrIdS architecture supports a multi-resolution concept for raster mosaics which is based on a resolution pyramid with constant-dimension tiles. The resolution access has been integrated with the spatial tile indexing scheme described above. The main features of the solution chosen are discussed below. A full description of the concept can be found in [Neb97].

- *Resolution Pyramid based on Constant Dimension Tiles.* In traditional resolution pyramids for individual raster objects, each level represents the same 'scene' or spatial extent. This has the desired effect that size and retrieval time of reduced resolution 'images' decrease roughly by the square of the reduction or scale factor. With raster mosaics, however, a typical operation consists of zooming a constant-dimension window in and out of the database. For this scenario, multi-resolution concepts for raster mosaics should keep the amount of data AND the number of objects to be retrieved roughly constant throughout the entire scale range. A resolution pyramid based on constant-dimension tiles fulfils both requirements.
- *Multi-Resolution Tile Indexing and Access.* The investigations of an efficient indexing and access mechanism for multi-resolution mosaic data resulted in a concept which can easily be integrated with the tile indexing method discussed above. The concept makes use of the duality between the Morton ordering and the region quadtree [OV96, Sam89]. Full-resolution tiles can be considered as leaf nodes of the region quadtree. Reduced-resolution tiles correspond to higher-level nodes within the quadtree structure, each occupying the spatial extents of four child nodes. These properties allow the spatial and the resolution access to be integrated. Reduced-resolution tiles are accessed by means of a range query at full resolution and a subsequent filtering process using the algorithm shown below. This algorithm can be implemented in the DBMS server using Persistently Stored Modules (PSM), which avoids sending the full list of candidate tiles to the client application.

$$sc_{(level\ n)} = sc_{(level\ 0)} - \mathrm{mod}\left(sc_{(level\ 0)}, 2^{2n}\right) \quad \text{(level 0 = full resolution, n = reduction level)}$$

Spatial Reference Model and Database

The requirement for a globally applicable georeferencing concept was fulfilled by adapting the georeferencing mechanism incorporated in the GeoTIFF format. It makes use of the POSC/EPSG (Petrotechnical Open Software Corporation / European Petroleum Survey Group) model and database. This is probably the most comprehensive publicly available list and it provides information on almost any geodetic datum, ellipsoid and map projection known in the world. The POSC/EPSG standard can easily be implemented in a DBMS environment and has recently been adapted by the Open Geodata Committee (OGC) as part of their ‘OpenGIS Simple Features Specification for SQL’.

Raster Storage and Compression

The current version of GrIdS supports BLOB-based DBMS-internal and file-based raster storage. GrIdS incorporates the standard compression schemes of the TIFF format for the DBMS-internal and -external storage. The current middleware-controlled file storage option (dotted lines in Figure 1a) will

be replaced by DBMS-controlled file storage concepts, such as the LOB/BFILE mechanism provided by Oracle8.

3.2 System Functionality

GrIdS incorporates modules for handling individual raster objects, raster mosaics, series of raster objects and a number of utilities, such as an interactive SQL database interface. For individual raster objects, the supported functionality includes import, georeferencing and export as well as comprehensive metadata management, such as the assignment of layer or series information. The raster mosaic management functionality includes the insertion and retrieval of raster objects or 'windows' of arbitrary dimension and size. It also supports the management and fusion of multiple mosaic layers.

Window Retrieval Operation

The window retrieval operation is the most frequently used operation on raster mosaics. It is subsequently used to illustrate the operational principle of GrIdS and consists of the following steps (see Figure 2a):

- *Tile Identification* – In the first step, the minimum and maximum Morton codes are determined in middleware. Following an optional optimisation step, a standard query with one or multiple code range predicates is used to retrieve a list of candidate tiles from the server. This list is further evaluated to reject any remaining tiles outside the query window.
- *Tile Retrieval and Data Projection* – In the second step, the data of the requested tiles is retrieved from the server and decompressed in middleware. The tile data is subsequently mapped to the structure of the export raster object. This mapping operation involves geometric and radiometric transformations, such as clipping and layer fusion. In order to support the export of arbitrary size raster objects, the export is performed on a sequential 'strip-by-strip' basis, whereby each completed strip is compressed and flushed to file or transmitted over the network.

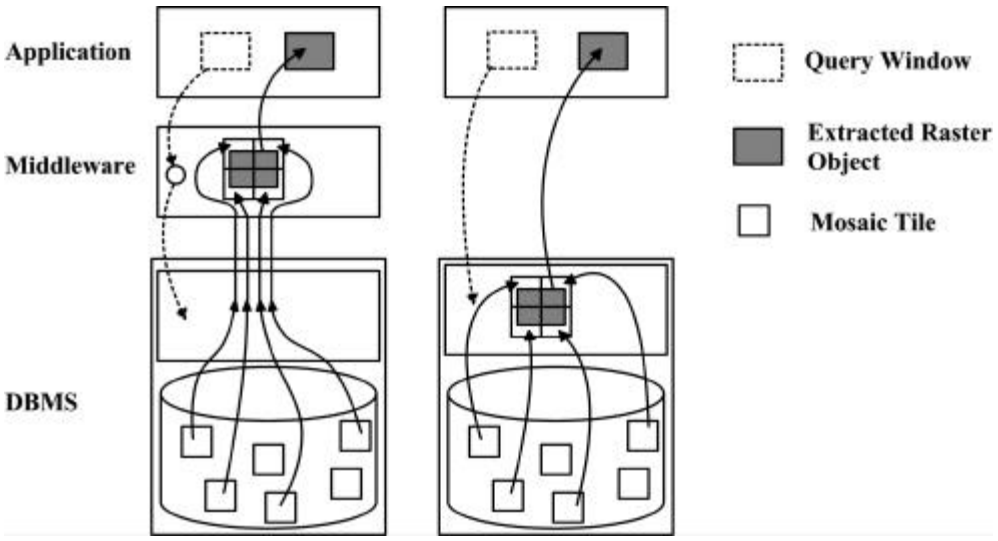

Figure 2 Mosaic window retrieval: (a) middleware solution (left) and (b) server extension (right)

3.3 Tests and Results

The prototype system was tested with raster maps and aerial orthoimages. The raster map data set consisted of 16 PixelMaps™ provided by the Swiss Federal Office of Topography. Each PixelMap was provided as 6 separately scanned and geocoded binary layers of 14'000 x 9'600 pixels, resulting in a total of 96 raster objects. The second data set consisted of 28 colour orthoimages provided by SwissPhoto Surveys, Regensdorf. The orthoimage dimensions were 4'000 x 4'000 pixels with a 24-bit RGB radiometric resolution. From these raster objects a number of mosaics with different tile

dimensions and compression schemes were generated. The size of individual mosaics was in excess of 1 GB and 20'000 tiles.

The test objectives included the evaluation of implemented concepts and the monitoring of the system performance. Particular attention was paid to the influence of tile dimension, tile size, compression scheme and raster type. A detailed description and discussion can be found in [Neb97].

In a first test series, the import and export performance for large objects was measured. This scenario is typical for the 'bulk' insertion of orthorectified satellite images or raster map sheets. The system demonstrated its capability to handle arbitrary size raster objects. However, the results showed a relatively low I/O performance of approx. 0.5 MB/s. The main limiting factors were identified as the DBMS's rollback overhead and the CPU performance[3].

In a second test scenario, the mosaic window retrieval performance was monitored. The window dimensions are often guided by dimension and resolution of display or printing devices. The window dimensions tested ranged from 512x512 to 1024x1024 pixels. The measurements included the time required to perform the tile query, to access, retrieve and decompress the tiles, and to compose a single contiguous raster object in main memory. The times measured did not include the subsequent display or export operations. The test results yielded window retrieval times from raster map mosaics from 0.6 to 1.5 seconds and from orthoimage mosaics in the range of 2.8 to 7.8 seconds. The results of this example compare well with those presented in [Lam94]. First tests on a modern UNIX workstation indicate that performance improvements by a factor of 5 to 10 can be expected.

4 CONCERT - A Database Architecture supporting Spatial Raster Data

Instead of implementing a raster data management component as a database application, as we presented in the previous section, such a component could be integrated into the DBMS server as an extension requiring the DBMS to be extensible. Many extensible research prototype systems such as ADT-Ingres [OFS84], DASDBS [SPS90], Exodus [CaD87], Genesis [BBG88], Postgres [RoS87] and Starburst [HCL90] have been developed and have started to influence today's commercial database systems (e.g. ORACLE 8, Informix Universal Server). Under the term "object-relational", these systems allow the database engine to be extended with additional functionality. This can be exploited to integrate operations such as the tile identification, the composition of individual tiles and the final clipping directly in the server instead of in the middleware. In the following we first discuss some general issues relevant for raster data management in an extensible DBMS. We then focus on CONCERT, an extensible database prototype system developed in the database research group at ETH providing extensibility concepts that are particularly useful for implementing a raster data storage system.

4.1 General System Architecture of an Extensible DBMS

Extensible DBMS are based on the idea of abstract data types. While traditional DBMS implement a fixed set of data types at their interfaces, for which they provide data definition, query capabilities, physical design and query optimisation, extensible DBMS allow the user to define new (abstract) data types together with specialised operations that enhance the query and optimisation possibilities of the system.

Raster image data types are not supported in traditional DBMS and there is little hope that in near future, there will be adequate support in these systems. Too many different raster image formats exist and the access requirements are too manifold. Rather, in order to gain good database support for raster data, a tailored raster extension of a standard extensible DBMS looks more promising. This approach allows all the operations that we implemented in GrIdS, to be integrated into the database server (as shown in Fig. 1) resulting in the following advantages and therefore potentially improving the results shown in Section 3.3 even further.

---

[3] Test platform: IBM RS/6000 530 workstation (model 1990) with a CPU performance of approx. 10% of a 1998 high-performance PC workstation.

Efficient Tile Index Access

In the GrIdS approach, identifying the necessary tiles to satisfy a window query involves calculating the minimal and maximal Morton code for the query in the middleware component, transforming these into an SQL range query to be evaluated on the DBMS server. The range query result is subsequently brought to the middleware, where it is filtered before finally the appropriate tiles can be requested from the server.

Calculating the Morton codes, performing the range query and filtering its result can be implemented as an abstract database server operation avoiding any unnecessary movement of data between the server and the middleware. Furthermore, instead of using Morton codes combined with a range query and a filtering step, a spatial data structure such as an R-Tree could be used to improve the tile identification even further.

Composition of Tiles and Clipping on the Database Server

Once the tiles are identified, they are retrieved into the GrIdS middleware, where they are composed and finally clipped to the requested size. This difference is illustrated in Figure 2. Especially for large tiles, the amount of pixels transferred to the middleware that are subsequently clipped and therefore discarded, can be substantial. In an extensible DBMS architecture environment, the composition and clipping of the tiles can be performed on the server side avoiding the shipment of excess pixels.

4.2 The CONCERT Architecture and its Specialities

Raster data collections are large, involving gigabytes or even terabytes of data. In order to efficiently provide access to such large amounts of data, good physical database design is crucial. While most extensible DBMS focus on functionality by allowing the user to integrate new operations into the kernel DBMS in a flexible way, the extensible prototype DBMS CONCERT [RB95, BRS96] we are developing at the ETH primarily focuses on aspects of physical database design. This makes it particularly useful as a platform for a raster management system. It is beyond the scope of this paper to give the full details of the CONCERT architecture and implementation. Details can be found in [BRS96,RSHN97]. However, by sketching the important issues, we want to give an idea of what an appropriate architecture for raster data management could look like in the future.

As opposed to other extensible DBMS, the CONCERT storage component does not accept arbitrary ADT extensions, but only such that are relevant for physical database design. It identifies the four fundamental concepts of physical design being

- the SCALAR concept: physical design can be performed ordering the objects according to some scalar property such as numbers or lexicographic order;
- the RECORD concept: objects might be composed out of individual components that can be stored and accessed separately;
- the LIST concept: objects might contain a set of properties (such as a list of words contained in a document), upon which they are to be searched;
- the SPATIAL concept: spatially located objects might be retrieved based on their spatial position and therefore stored in spatial indexes organising the data space into hierarchies of subspaces.

For each of these four concepts, CONCERT defines the ADT operations relevant for physical design (the ordering predicate for SCALAR, the partitioning into components for RECORD, the iteration over the LIST elements and the subspace relationships for SPATIAL objects). These operations have to be implemented by the ADT designer.

Physical database design is then performed solely based on these ADT operations, enabling the database to use standard physical design techniques for arbitrary ADT objects. A particularly relevant aspect for raster image management is the fact, that the CONCERT techniques allow data objects to be stored outside the database, but still provide physical database design possibilities for them. This is similar to the Informix Universal Server's virtual table interface, which provides query capabilities for data stored outside the database in an external storage as shown in Figure 1(b).

Exploiting the SPATIAL concept, spatial partition queries over raster image mosaics can be directly supported in the CONCERT kernel via ADT operations. Furthermore, standard physical database design supporting such spatial partition queries via precomputed result sets can be performed.

We identified tertiary storage integration as an important issue in a raster data management system that is difficult to achieve in traditional systems. The CONCERT concepts provide an elegant solution for this problem. Large raster image collections require most of the images to reside on tertiary storage caching only a few images or mosaic parts in secondary store.

CONCERT can view these image objects as consisting of two parts, one being its tertiary storage representation and the other being the secondary storage cache. Using the RECORD concept, these two views can be supported via the component selection ADT operation making tertiary storage access for the database look like an ordinary record projection operation.

5 Conclusions

Most of the spatial raster management concepts presented in this paper can either be implemented in middleware or as server extensions. The following comparison summarises the advantages and limitations of each approach. This comparison is followed by an outlook to some of the ongoing and future work.

5.1 Comparison: Middleware versus Server Extension

The presented middleware-based spatial raster management solution has proven its capability to manage very large raster objects and raster mosaics. It can be implemented economically on standard DBMS and hardware technology and offers a good scalability and performance – if operated on a modern platform. Middleware-based solutions will benefit from the improved external and tertiary storage management support and the spatial support which are gradually being added to commercial DBMSs. The concepts presented could relatively easily be integrated with existing middleware-based spatial data management solutions for GIS.

The server extension concepts presented offer a performance and functionality superior to that available on today's standard (object-) relational DBMS technology. As such, server extensions clearly represent the technology of the future. However, it should be noted that the implementation of server extensions, such as type specific access methods, is more complex and expensive than a middleware-based solution.

5.2 Future Work

The investigations presented have been carried out as part of the RasterGIS project. The RasterGIS project is a joint effort of the Database group, the GIS group and the Photogrammetry/Remote Sensing group at the ETH Zurich. The research objectives of the project include the investigation of integrated management of spatial raster and 3D-objects in GIS. Ongoing and future work on the basis of the prototype platforms presented includes the modelling and integration of 3D-objects, the closer integration of GIS and photogrammetric workstations and processes, and Internet/Intranet integration.

References

[AM90] D. J. Abel and D. M. Mark. A comparative analysis of some two-dimensional orderings. International Journal of Geographical Information Systems, 4: 21-23, 1990.

[AS84] D. J. Abel and J. L. Smith. A data structure and query algorithm for a database of areal entities. The Australian Computer Journal, 16:147-154, 1984.

[BBG88] D. Batory, J. Barnett, J. Garza, K. Smith, K. Tsukuda, B. Twichell, and T. Wise. GENESIS: An extensible database management system. IEEE Transactions on Software Engineering, 14(11), November 1988.

[BRS96] Stephen Blott, Lukas Relly, and Hans-Jörg Schek. An open abstract-object storage system. In Proceedings of the 1996 ACM SIGMOD Conference on Management of Data, June 1996.

[CaD87] Michael J. Carey and David J. DeWitt. An overview of the EXODUS project. Data Engineering Bulletin, 10(2):47-54, 1987.

[GG95] V. Gaede and O. Günther. Multidimensional Access Methods. Humboldt Universität zu Berlin, Wirtschaftswissenschaftliche Fakultät , Institut für Wirtschaftsinformatik, Berlin ISS-16, October 1995.

[HCL90] L. M. Haas, W. Chang, G. M. Lohman, J. McPherson, P. F. Wilms, G. Lapis, B. Lindsay, H. Pirahesh, M. Carey, and E. Shekita. Starburst mid-flight: As the dust clears. IEEE Transactions on Knowledge and Data Engineering, 2(1):143-160, March 1990.

[Lam94] P. Lamb. Tiling Very Large Rasters. Sixth International Symposium on Spatial Data Handling, London, 1994.

[Neb97] S. Nebiker. Spatial Raster Data Management for Geo-Information Systems – A Database Perspective. PhD thesis. Blue Series Report No. 63, Institute of Geodesy and Photogrammetry, ETH Zürich, November 1997.

[OFS84] James Ong, Dennis Fogg, and Michael Stonebraker. Implementation of data abstraction in the relational database system ingres. ACM SIGMOD Record, 14(1):1-14, March 1984.

[OV96] P. V. Oosterom and T. Vijlbrief. The Spatial Location Code. 7th International Symposium on Spatial Data Handling, Delft, 1996.

[PW96] R. Pajarola and P. Widmayer. Spatial Indexing into Compressed Raster Images: How to Answer Range Queries Without Decompression. Int. Workshop on Multimedia DBMS, 1996.

[RB95] Lukas Relly and Stephen Blott. Ein Speichersystem für abstrakte Objekte. In Proceedings of the 6th German Conference on Database Systems for Office, Engineering and Scientific Applications (Büro, Technik, Wissenschaft, BTW), Lecture Notes in Computer Science, pages 338-347. Springer Verlag Berlin Heidelberg New York, March 1995.

[RoS87] L. Rowe and M. Stonebraker. The POSTGRES data model. In Proceedings of the Thirteenth International Conference on Very Large Databases, Brighton, England, 1987.

[RSHN97] Lukas Relly, Hans-J. Schek, Olof Henricsson, and Stephan Nebiker. Physical database design for raster images in concert. In 5th International Symposium on Spatial Databases (SSD'97), Berlin, Germany, July 1997.

[Sam89] H. Samet. The Design and Analysis of Spatial Data Structures. Addison-Wesley Publishing Company, 1989.

[Sha94] F. Shahin. A Hierarchical Approach to Digital Image Compression. ASPRS/ACSM, 1994.

[SPS90] Hans-Jörg Schek, H.-B. Paul, M.H. Scholl, and G. Weikum. The DASDBS project: Objectives, experiences, and future prospects. IEEE Transactions on Knowledge and Data Engineering, 2(1):25- 43, March 1990.

[Sto96] M. Stonebraker and D. Moore. Object-Relational DBMSs - The Next Great Wave. San Francisco, California: Morgan Kaufmann Publishers, Inc., 1996.