OBJECT-ORIENTED GEOGRAPHICAL DATABASE MODEL

Jet Propulsion Laboratory, Pasadena, California 91109

Mosc LaMark Johnson, hr. Nevin Bryant, Dr. Demetrios Sapounas

1. ABSTRACT

Terbase is an ob~cct-oriented" database system under development at the Jet Propulsion Laboratory (J])],). Terbase is designed for flexibility, reusability, maintenance case, multi-user collaboration and independence, and efficiency. This paper details the design and development of Terbase as a geographic data server, using examples proposed for the Tactical M ovement Anal yzer (TMA). TMA is a software tool, also developed at JPL, designed for predicting the mad and cross-country movement of vehicles such as Transportable Erector-Launchers (TEL.s). Terbase uscs Object-Store''N~ by Object Design inc. (ODI) for basic database persistent storage and query. Object-Store''M is used because it is an ob~cct-oriented" database system that runs on several platforms in a heterogeneous manner. Terbase is designed to provide hierarchical functionality. 1 'ach function or service provided by this system can be classified under a concept definition. The idea of concept definitions provides the system administrator with a scheme for adding new services with minimum coding. A service will inherit the concept definitions functionality under which it is declared. 1 'or instance, a new service declared as part of the indexing concept will automatically index any data generated by that service. Terbase allows the user to create spatial windows into the database for running geographically efficient queries. By using windows, a query will not search the entire database but only information in the area of interest. The window automatically keeps a legend and history of actions performed. The window and its information can be save.d as a database object. It can also be set to work in collaboration with other windows. The windowing service combine the concept definitions of automatic legends and indexing. Terrain data is usually accessed by location then accessed by data type. By default, the Terbase system usc data sets as a strategy for clustering data since the user provides the location indexing using windows. By clustering data on data type boundaries the user can more efficiently specify information for analytical study. The packaging structure of Terbase allows this system flexibility by providing only the components required for the task at hand. Libraries arc bounded by Concept 1)efiniti ens. 1 'or instance, Terbase dots not need to implement raster utilities if images will not be used in the final application. Terbasc dots this by maintaining a description file for each application or set of applications. Terbase was designed as an application programmers' tool. This system was designed to benefit the application developer as well as the end-user. It provides the basic services needed for terrain storage and querying.

2. INTRODUCTION

Today's military actions require fast and accurate responses by U.S. forces if operations are to be successful. Most information for an operation invol vcs geographical data. The acquisition and anal yst of this type data can create a speed barrier for the operation. Most of the information is readily available from various sources that can be stored in a database for later USC. However, one or more sources may require individual preprocessing to be valuable. The data format may not be compatiblc with the database system.

The data required by a mission planning system is both varied and volatile. The ongoing but irregular updates to the data mean that a flat-file data management approach is inadequate, while the different types of data the system uses -- raster and vector, topographic and mechanical, fixed and moving -- suggest a Relational Data Management System (RDMS) or an Object-oriented Data Management System (OODMS) is the best approach.

With a relational system we can create an application that satisfies conditions for a finite number of operations. The application would provide protection against data corruption through atomicity, consistency, backup, and concurrently. However, the overhead for an RDMS could degrade performance. A relational model would also consist of many tables and keys and a query could result in several queries whose output would need to be correlated. The structure of the database would dictate flexibility. The system's structure would be a static model unable to change and data may require preprocessing efforts to fit it into the system's model. Preprocessing could still introduce corruption and a loss of useful information. Most Relational models do not support complex data such as two and three dimensional images like terrain and weather data. This may be adequate for some scenarios, but many could require a more, flexible system.

over the past few years we have heard a great deal about the advantages of Object Oriented Development. Object-Oriented Data Management Systems reduce time and cost for production and maintenance of a complex system by providing models for developing reusable code. It also makes modeling real-world entities easier.

An Object-oriented approach would mean that a mission planning system could be versatile. Data would no longer require preprocessing. The system could be extended to suite new data. It could add new models for each data source acquired. Possible corruption would not be a problem because system extensions could maintain data integrity throughout the life of the data.

Through encapsulation, the planning system could present the user with a common interface for all the data. The source of the data would not need presentation because the user would only concern themselves with operations on the data. Implementation of the data tools and structure of the data would be hidden from the user.

object-oriented" Data Management Systems are ideal for complex applications, such as mission planning, that require data that is not record oriented. The object-oriented system provides several methods of accessing data. Data can be retrieved though pointers to related objects. This is the most efficient form of access. However, most systems require thes ystem's second form of access, lookup through attribute values.

3. SYSTEM PURPOSE

Object-Oriented Data Management Systems represent younger technology then the better known relational database systems such as Oracle and Sybase. Nevertheless, there are a number of commercial products for genuinely object-based data management available [Cattell1991]. Terbase is a set of classes that are used to extend existing object-based data management systems. It acts as a modular interface. Application programmers use the classes as the main interface to the Object-Oriented Data Management System. However, the programmer may still use the direct interface to the OODMS.

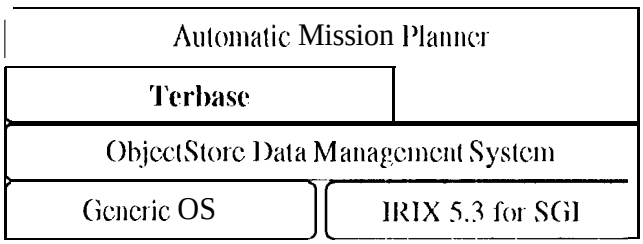

Figure 1: Terbase is an interface between the application and the ODMS.

The overall purpose for Terbase is to provide a set of rules and agents that will become the standard solution for development of an Automatic Mission Planner to used by next generation Autonomous Unmanned Vehicles. The challenge in developing such an interface is to provide complex data model support, including graphic data, spatial data, and attribute data as well as machine heterogeneous support. The system must be flexible and efficient. The Terbase Model will define a set of management system independent rules that could provide greater support for any environment an autonomous vehicle may display. in the case of Figure 1, ObjectStore™, by object Design Inc., was selected because of the control it gives the program mer. As the Terbase Model is perfected, requirement for ObjectStore™ will be abolished and any object Based System will work.

4. EXAMPLE APPLICATION

Jet]'repulsion Laboratory and Naval Surface Warfare Center (NSWC) are currently involved in a project encompassing the development of' the Tactical Movement Anal yzer and the Object Oriented 'J'errain Dat abase, known as Terbase. The office of Naval Research has tasked NSWC to develop an integrated prototype mission planning system by the year 2000. The prototype system will be based on Route Planner, 'l'MA, a Smart Search Planner, and Image Processing for targeting.

Project planning from this date to the year 2000 involves a number of interim releases of Terbase and "l'MA. Each release will provide the necessary functionality to aide in the integration and progress of the other tools.

The tools under development, for the Automated Mission Planning system, arc independent standalone prototypes that use flat-files for their data needs. This makes is difficult to operate the too] on different geographic areas without first pre-processing data to meet the requirements of each tool. To make each tool more flexible and enhance its case of USC, the development of a database was started, to be eventually used as the data repository for all the prototypes.

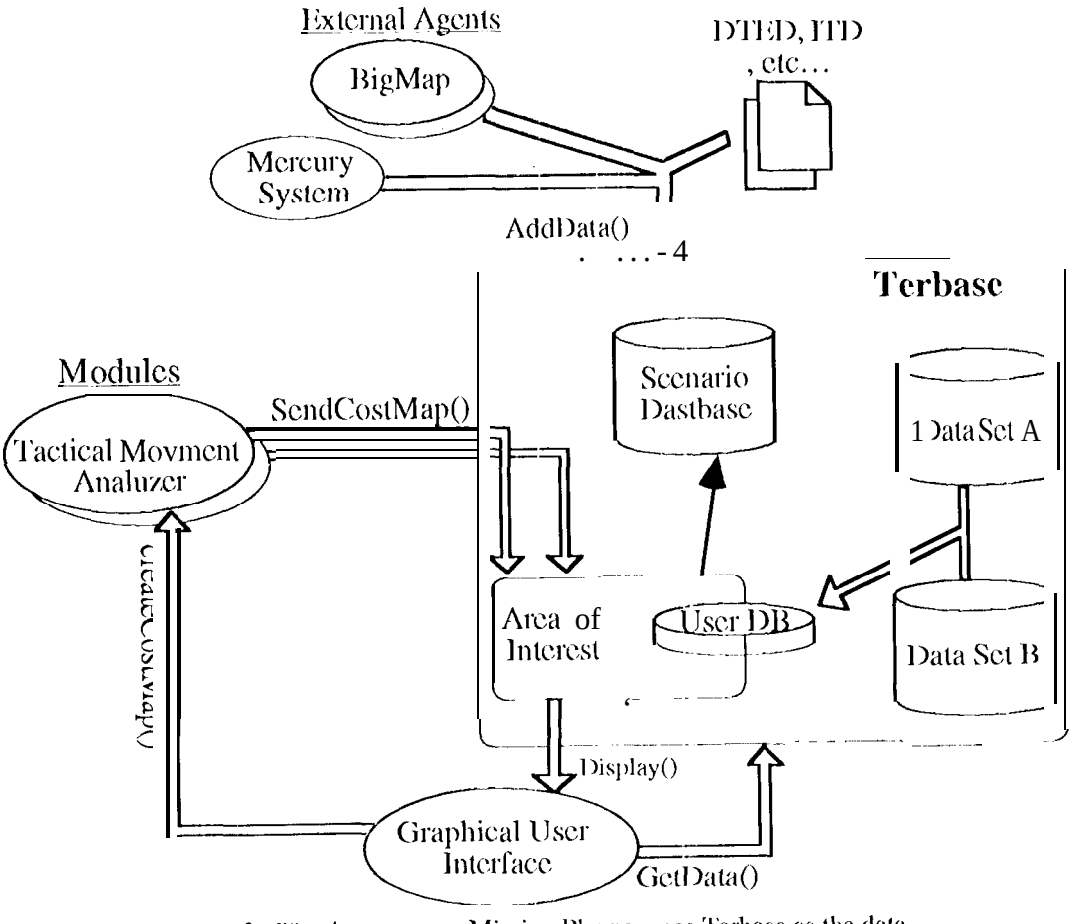

Figure 2: The Autonomous Mission Planner uses Terbase as the data and system integrator

Route Planner

The route planning application will provide mutes for aerial vehicles over terrain. The data needs, for the module include terrain elevation, threat models, and weather. It provides tbc routes from a departure location to a destination location. Multiple mutes arc available, ranging from optimal to sub- optim al, defined by corridors.

The Router combines the data into two-dimensional and three-dimensional cost surfaces. The three-dimensional cost surface is optimized for routing on the horizontal plane, where the three-dimensional cost surface incorporates vehicle flight constraints for better route/flight modeling in three dimensions.

Tactical Movement Analyzer (TMA)

The Tactical Movement Analyzer [Sapounas and Kreitzberg 1994] is a tactical decision aid that

provides information on movement capabilities for a generalized set of vehicles over arbitrary terrain. To achieve good modeling and simulation the database cannot be static but must evolve with time to present the most accurate and up-to-date information. A high accuracy, representing the latest operational profile, will result in better and more accurate predictions. Most of the data required for movement prediction will not change over time, but certain features like bridges and roads can change. These changes can greatly impact the movement capability of the ground forces.

Terbase

Terbase will be interfaced to all the Automated Mission Planning tools requiring persistent data. Most of the modules will require up-to-date information on the area of interest to plan an effective mission. Terbase will serve as the information repository for GIS-type data. It will also be responsible for rapidly returning information in response to a query. Data that must be included are geographic information, threat locations and characteristics, target locations and characteristics, keep-out zones, missile inventories, and missile characteristics.

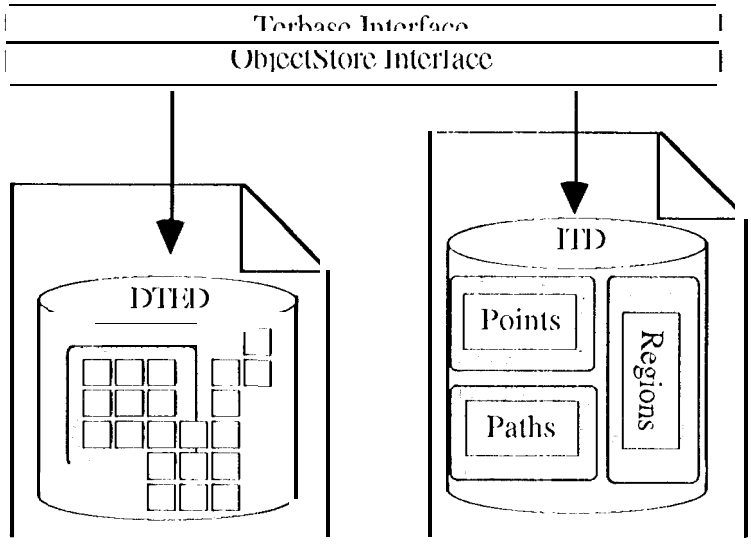

Figure 3: Each data set is stored in its own database. The Terbase user interface treats them as one.

5. STRUCTURE SPECIFICATIONS

Location Interface

For Terbase, the *Location object* is one of the most important structures. Most information in the database uses it. It provides positions within the environment in terms of a visual map. The object

represents common interface for the set of objects used to specify Cartesian coordinate space.

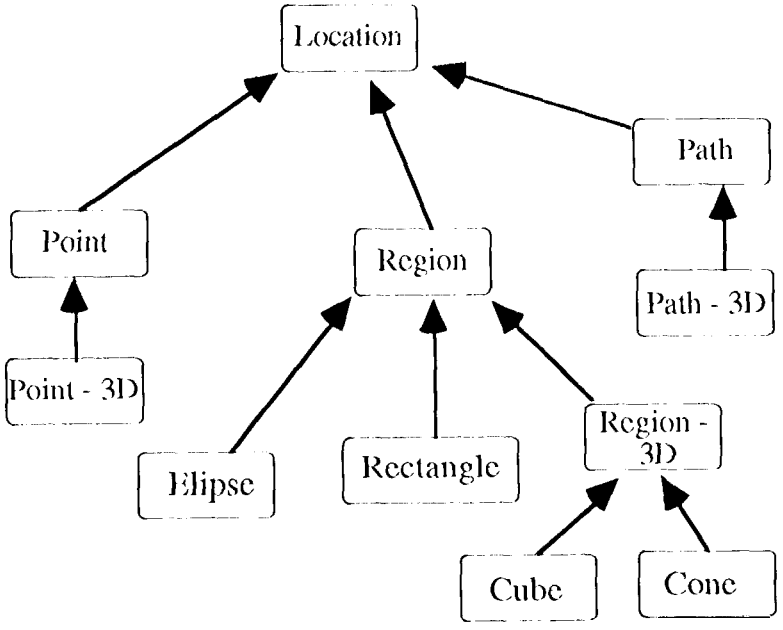

Figure 4: Location interface is used by a programmer to create rudimentary structures.

Asset Interface

The asset interface object represents an individual military unit. Assets are stored in a database as a set of attributes without a geographic location. To provide dynamic information about the asset, an *AssetDynamic object is* created. One of the important attributes of an asset is its alliance. Each asset, therefore, has a friend or foe status.

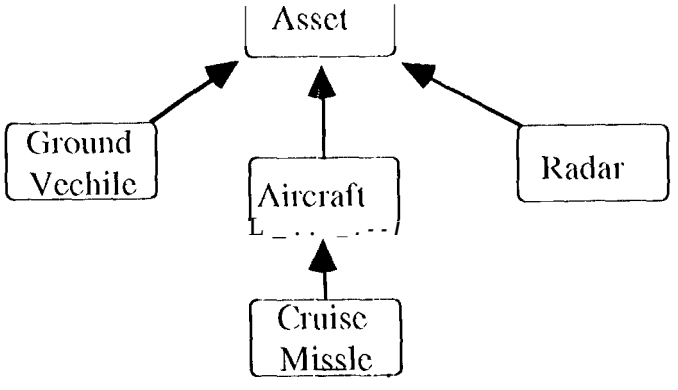

Figure S: The Asset Interface is the user i!ucl-face to all objects that can be targets or move.rs.

Feature Class

A *feature object* consists of a location, associated attributes, and a list of neighboring features. The feature class is the super class of most other vector classes. It simply provides the attribute functions for any vector relation required. The *point object* is a feature object represented as a single geographic coordinate. A point object can be used to represent entities like buildings on large scale maps or locations of interest. A *path object* is a feature object represented by an ordered list of geographic coordinates. The first point in the ordered list is the starting point and the last point is the ending point, therefore, a path has direction. These objects usually represent roads, rivers, and path transversals by moving entities. A *region object* is represented by an ordered list of geographic coordinates that defines the boundary of the region (including any islands). These features are most used to represent lakes, land use areas, and the like. The *group object* is a set of feature objects including group objects. These objects represent a set of entities such as cities and TOWNS.

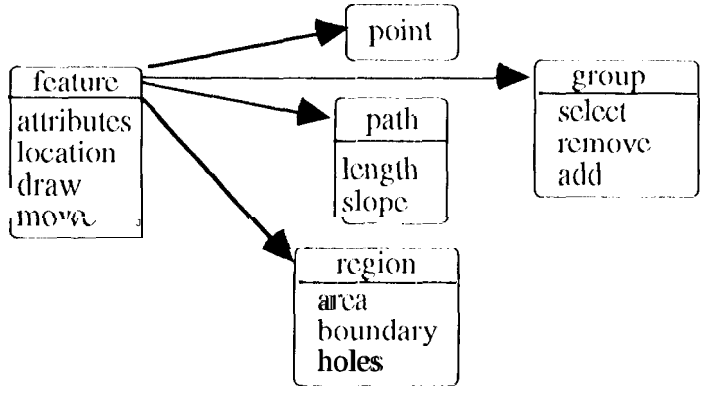

Figure 6: Features provide basic GIS-type data structures.

Image Class

An *image object* consists of a two-dimensional matrix of data values plus associated information. The *Georeferenced image* subclass adds a geographic location and projection to an image. A GeoTIFF image is a type of *Georeferenced image*. There are some subclasses of the Georeferenced image. The *elevation image* is a Georeferenced image whose pixel values are evaluations. It contains such attributes as units of elevation. The *map image* is a georeferenced image with a translation table for associating the matrix data values with a character code and an informational text string.

Neighboring georeferenced images can be combined to produce an arbitrarily large coverage area. The *mosaic object* has been implemented as set of georeferenced images to support this feature in a manner transparent to the end user. In particular, the elevation data for a region is stored as a mosaic of one-degree square elevation images.

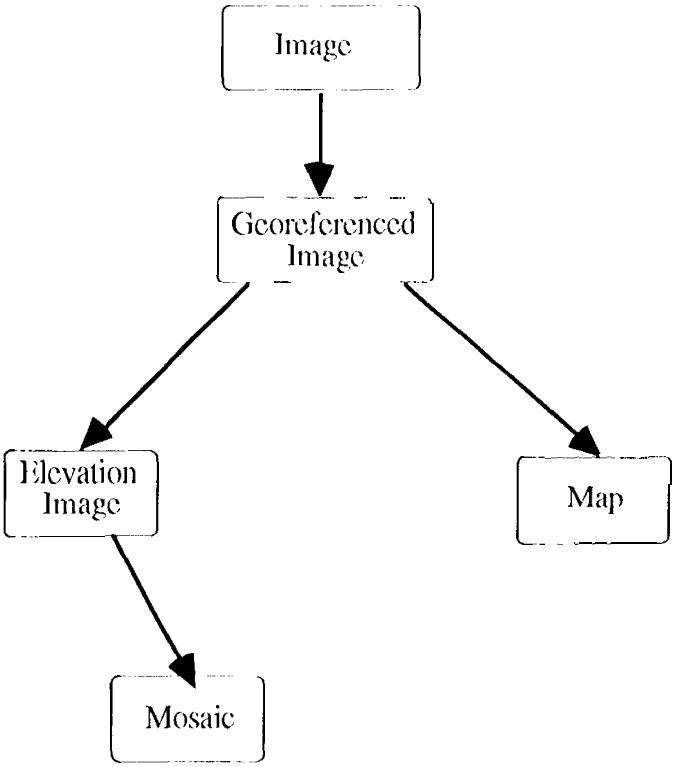

Figure 7: Image Structures provide the matrix support for the Automated Mission Planner.

AOI Class

Because a Terbase database size is limited only by hardware resources available, a database can become enormous. Even though an ODMS is designed to handle multi-gigabyte data sets, the applacations can be (and are usually) far less robust. Terbase defines an *A OI object as means* of organizing data regarding a specific geographic area of interest. The object maintains all the information and the battlefield situation such as the weather, the terrain and the development of both friendly and enemy assets; about intermediate calculations that have already been done as part of a process; and about the histories of changes made within the AOI. An A()] class contains a unique name. This is the identification the system uses to make reports on the scenario. The specification of the area of coverage is an attribute that is necessary when limiting the domain of the scenario. When the A()] object is created the area of coverage is used to retrieve all the data that is within its boundary. The user is limited to this data when performing queries and other database related functions. In this way, an instance of this object acts like a geographical window into the database. The data available. to the user in the initial prototype is a (possibly empty) set of images containing raster data lying within the boundary of the A()] and a history list describing the decisions made because of the A()] object. once an A()] object has been created with the desired set of features and images, it may be used directly by the application without searching the entire database again. The user can extend or reduce the area of coverage of the AOI. This would automatically add or remove data from the scenario as necessary.

6. 1 MPLEMENTATION

Backup

The primary operation required from a database is safe long-term storage. If a database file that is in constant use is to remain safe it must be backed-up often. The database backup procedure occurs at the end of a predetermined interval. This interval can be *transactio n-b ased* or *time-based.* In a transaction-based backup procedure, changes to the database are recorded in a log. This allows the backup to be up-lo-date at the expense of performance. A time-based backup procedure occurs when a timer reaches reset time. It is up to the user to determine the tinlc-interval or even the number of transactions that cause the backup procedure to initiate. The backup process in its most simple implementation copies the database at the operating system level and restores it by replacing the current database file with the copy. Note that this is an implementation and not a model definition.

Integrity

The Terbase model defines rules that rely on the control given it by the underlying ()]) MS. Data Integrity is maintained on a per-instance bases, that is when a new object is created, its attributes are checked for value bounds and placed in the appropriate context.

1 :ach onc of the general classes, feature, image, A()], etc., has an *indexing object* associated with it. The indexing object manages creation, examination, and (instruction of an instance of the general class. When an A()] object is created, the index checks the database to ensure the uniqueness of its name. For feature objects the index simpl y assures the availability of the object for later USC. in all the cases of an index, the spatial bounds are automatically updated to reflect a rectangular area that all the objects cover.

Terbase will manage a number of databases. The *System database* will contain all the data required for any mission, including session environments. "'his database will be read-only to a *Common-User* and require an administrator or *Super-User* to keep it updated. The *User-data base can* be read or written by the initiator of the session or the Super-lJscr.

Clustering

Clustering is a term that refers to the strategy of placing data commonly used together near each other on the disk. Due to the way Object oriented Data Management systems access database memory, an intelligent clustering scheme is required for adequate performance, especially in a nlulli-user environment. ODMS are usually client-server models. The information in a database is stored non-locally. The client application typically runs on a different machine. When the application requests an object from the ODMS for the first time, tbc object is transferred from the database to the application's memory.

The time it takes to establish a communication path and then send data across is usually long com-

pared to memory access. in RDMS this is a large percentage of the efficiency loss. in ODMS, however, once the transfer is made it is maintained in the application's memory until sonic event removes the object from memory such as the end of a process. However, the ODMS usually goes onc step further. When an object is transferred, the ODMS also transfers data from disk that is near the target object. This cuts down on network access.

Most ODMS provide several mechanisms for transfer units. Objects may be clustered according to *fetch quantum* [0111, 1993], a user defined number of pages. A page is the smallest physical unit of transfer. DTED data is a good example of the use of page clustering. DTED 1 .evel 1 data is split into 1 degree by 1 degree cells. The data can be stored in 300 pages (about 2 megabytes if a standard page is 8kb). Since, each 1 degree by 1 degree cell has a predetermined size a fetch quantum can be set to optimize performance. A fetch quantum of 4 provides about 400 statue miles of 1Y1'1;11 Level 1 coverage. The same fetch quantum (using DTED 1 .evel 2) produces about a third of that area. Terbase uses this type of clustering with raster data.

Segment clustering is most used by Terbase. Objects may be clustered in larger units, which have logical boundaries for grouping. This form of clustering provides greater flexibility. Vector data is usually stored in segments because a good estimated size of granularity can not be determined. For instance, ITD data is stored in segments based on 7.5 minute quadrangles.

in Terbase there arc many other levels of clustering involved. Each data source is maintained in its own database file. For instance, DTED and ITD data are independent of each other at the operating system level. Because each data source is in a separate data file, fetch quantum can be controlled independentl y. This allows each data set maximum performance. When the user uses Terbase all data is presented as one database. Therefore, there is no apparent loss in functionality.

The data structures (AOI, asset,...) defined by Terbase arc a small set oft ypcs that are used by the initial Autonomous Mission Planning System. These structures arc known to be suited for the implementation of the system. Terbase was, however, designed for usc by any system that need a Geographical information System (GI S).

Registration and Concept Interface

The Terbase system will define an interface object called *Concept.* With a *Conceptinterface* the application programmer can extend the storage capabilitie s of Terbase. First, the programmer must create a logger for the new data type. in the example system, the logger was a program that created data files from Defense Mapping Agency's ADRG product. The programmer will then have to place a standard wrapper, defined by Terbase, around the new module. This is called registering the module.

The wrapper provides attributes that the programmer uses to adapt Terbase for the new data type. One attribute is used to declare the model of data being added. The models that will be defined by Terbase arc Raster, Vector, and Attribute.

Raster Data. Raster Data is mostly images. The data usually represents a matrix. Its indexing scheme is most likely to be tiled into smaller structures of the same type. Clustering is based on tile boundaries. Some examples of Raster data are satellite images, photographs and screen dumps.

Vector Data. Vector Data will require less storage space than raster data but more time to process. An object whose origin is a vector source most likely can not be decomposed into smaller objects that use similar structures. By default, Terbase will set its clustering to be type based. Examples include land use, roads, and city locations.

Attribute Data. Attribute Data is usually text data. It is most times human readable and uses ASCII or some other character code standard. 1 lowever, attributes can be binary. The default indexing provided by Terbase for this type data is by reference.

There will be other such attributes. These may include clustering schemes such as Composite objects, References, or Object types and clustering granularity of Page and Segment [Cattell 1991]. The Index scheme may also be dynamic. Some attributes may explicitly designate implementation. If there are new data structures they will also have to be registered within the Terbase system.

7. SUMMARY

Modeling and Simulation applications require large amounts of the data in various formats. Object oriented databases can handle such data easily and efficiently and make the process of accessing and querying the data straight-forward. Terbase is an interface placed on top of a ODMS to provide a programmer with GIS-type functionality. The implementation of the system can be hidden and the application programmer can still extend the functionality of the ()]) MS.

The easy access to the data definition has proven a valuable feature. The ability to operate on data type definition al runtime provides the user with a customization feature. If a new data type is required, the programmer can create the logger and register it to Terbase. Registration automatically sets default clustering, backup, and indexing for the data.

I lath new data type is stored in a new database. Registration provi dcs all the interface, therefore, the programmer never has to realize the discontinuity in the system. The default clustering scheme depends on the data type. Raster data is page clustered while vector data and attribute data are clustered by thei r feat ure definition.

An indexing scheme allows the user to access all the data by traversing references. Indexing is affected by the creation and deletion of features but the programmer dots not need to be concerned.

'J'here arc parts of the underlying, ODMS with which the programmer must contend. The most noticeable is the query language, but, language translators can be added to the Terbase interface. This will make any query language avai lable the programmer.

Terbase is still in development. Terbase is not scheduled to be completed until near the end of this century. The system model is under development. The interface has not been completely defined and many performance concerns have not been addressed, however, much of its functional background has been defined. Many of the functions needed by the mission planner will initially be prc-existing software "registered" into Terbase.

8. REFERENCES

Cattell, R.G.G 1991, "Appendix. Products and Prototypes," *Object Data Management,* pp **239-267,** Addison-Wesley, Reading, MA

ODI 1993. *ObjectStore™ User's Manual*, Object Design Inc., Bulington, MA.

Sapounas, D. and Kreitzberg, '1'. 1994. "The Tactical M ovement Anal yzer." Technical Report NSWCDD/TR-94/99. Naval Surface Warfare Center, Dahlgren Division, Dahlgren, VA (Sep.).

9. BIOGRAPHY

Mose LaMark *Johnson.* received his BS in Computer Science from Jackson State University in 1994. 1 Ie currently works at Jet Propulsion Laboratory in the Cartographic Applications Group as a rapid protot ype developer and database, engineer. His interests are in man-machi ne interfaces, artificial knowledge extraction, and database. design.

E-Mail: mlj@tazboy.jpl.nasa.gov

FAX: (818) 354-8982.

Dr. Nevin A. Bryant. has been the Technical Group Supervisor of Cartographic Applications at Caltech's Jet Propulsion Laboratory since 1983. Dr. Bryant was the cognizant design engineer for the Image Intelligence and Message Parsing and Generation Subsystems in the All Source Analysis S ystem (AS AS) project sponsored by the Army from 1983- 1987. Since 1988, Dr. Bryant has been the Principal Scientist for the ASAS TechBase Program, the. research and development activity of the ASAS Program office. in that capacity he has been responsible for the guidance of both JPL and universit y research on i mproved techniques for data fusion, spatial reasoning, and map knowledge base functions.

Ii-Mail: Nevin.Bryant@jpl.nasa.gov

FAX: (818) 354-8982

Demetrios Sapounas. received a Ph.D. from Cranfield University, an M.%. from Wichita State University, and a B. Sc. from Tulane University, all in Computer Science. For the past six years he is a Computer Scientist with the Naval Surface Warfare Center, Dahlgren Division, where he conduct s research and develo pment . His interests are in visual simulations, nlan-machine interfaces, multimedia, and animation.

E-Mail: ds@aris.nswc.navy.mil