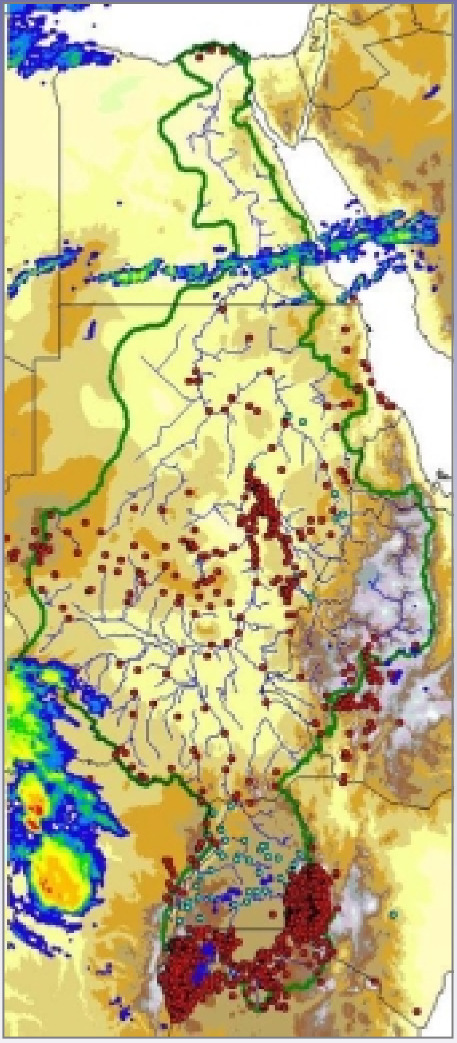
Nile Decision Support Tool Database Design Principles

Burundi
Congo
Egypt
Eritrea
Ethiopia
Kenya
Rwanda
Sudan
Tanzania
Uganda



Developed collaboratively by

The Nile Basin Nations,

The Georgia Water Resources Institute at the Georgia Institute of Technology,

and

The Food and Agriculture Organization of the United Nations

June 2003

Nile Decision Support Tool (Nile DST)
Database Design Principles

Report developed by

Stephen Bourne
Research Associate

Aris Georgakakos
Project Director

Georgia Water Resources Institute
School of Civil and Environmental Engineering
Georgia Institute of Technology

In collaboration with

The Nile Basin Nations

and

The Food and Agriculture Organization (FAO)
of the United Nations
Nile Basin Water Resources Project (GCP/INT/752/ITA)

June 2003

Acknowledgements

This report and associated software were developed by the Georgia Water Resources Institute (GWRI) at the Georgia Institute of Technology as part of the Nile Basin Water Resources Project (GCP/INT/752/ITA). This project was funded by the Government of Italy and was executed for the Nile Basin nations by the Food and Agriculture Organization (FAO) of the United Nations.

The GWRI Director and project staff are grateful to the Nile Basin nations (Burundi, Congo, Egypt, Eritrea, Ethiopia, Kenya, Rwanda, Sudan, Tanzania, and Uganda), their focal point institutions, their Project Steering Committee (PSC) members, and their National Modelers for entrusting us to work with them in this important basin-wide project. The development of databases, models, technical reports, software, and user manuals are key but not the only project accomplishments. Even more important are the evolving contributions relating to people and the difference the project is poised to make in data and information sharing, developing a common knowledge base for policy debates, and long term capacity building.

GWRI is also grateful to the Government of Italy and to FAO for sponsoring this project and for providing dependable logistical and technical support through the FAO offices in Rome and Entebbe.

It is our hope that the Nile DST effort will contribute in some positive way to the historic process of the Nile Basin nations to create a sustainable and peaceful future.

Aris Georgakakos
GWRI Director
Atlanta, June 2003

Disclaimer and Copyright Notice

Table of Contents

Nile DST: Database

1. Introduction

The Nile DST Database is a comprehensive data warehousing structure capable of storing and visualizing meteorgological, hydrological, climatological, agricultural, river basin management, demographic and spatial data. This report will describe the design, and development of the database and the DST interface structure into which it fits.

The report is comprised of two sequential sections:

- Establishing Design Goals
- Database Design and Structure

2. Establishing Design Goals

Designing database structures to handle the range of applications held in the Nile DST is a considerable task. Each application requires its own set of input data and generates its own set of output data. The data used is both spatially and temporally distributed with multiple resolutions. The data is regularly and irregularly spaced, and in many cases, temporally sparse. Consequently, the database design needs to account for many different possible data configurations. Furthermore, the Nile DST is intended to be an evolving tool, capable of housing, providing as input to models, and visualizing many new data types that have not been considered.

To design the Nile DST databasing system three questions were considered:

- What data will be used?
- How will the data be used?
- Based on answers to the first two questions, what goals can be defined that will meet the databasing needs and how attainable are these goals?

2.1. What data will be used?

To answer the question, “**What data will be used?**,” GWRI considered the applications that will be included in the Nile DST. As figure 1 shows, they include **Remote Sensing and Ground Monitoring, Hydrology, River Basin Management,** and **Agricultural and Urban Planning.** The **Database** and its associated analysis tools are also included.   Following is a specific, representative application from each of these larger areas and the data that the application would need as input and would generate as output.

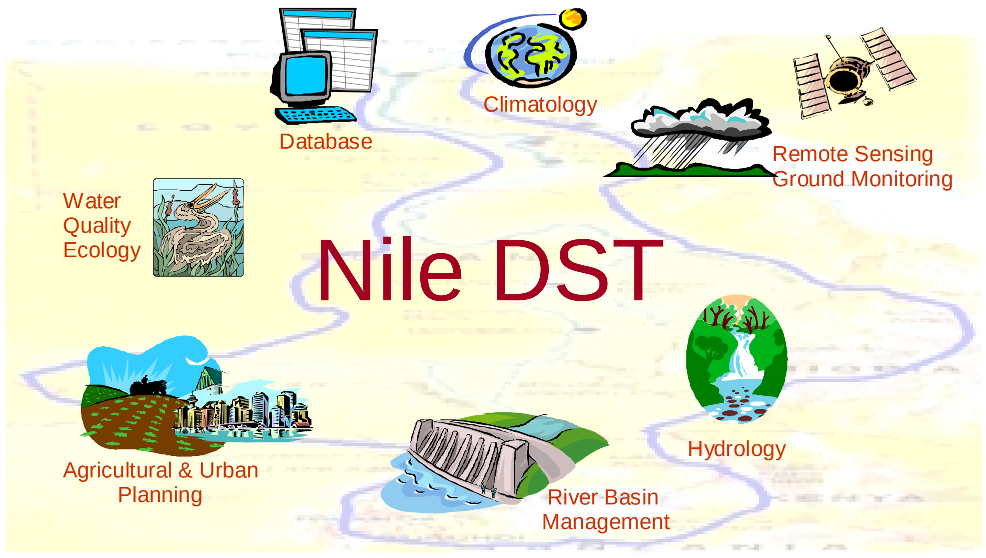


Figure 1, The Nile DST will house several applications.

River Basin Management – Hydropower Control Simulation and Optimization requires data on water input into reservoirs, schematic data describing the river network, and performance data for the hydropower facilities on the river. Optimization produces time series of optimal control policies and facilities operation. Figure 2 shows the data needs.

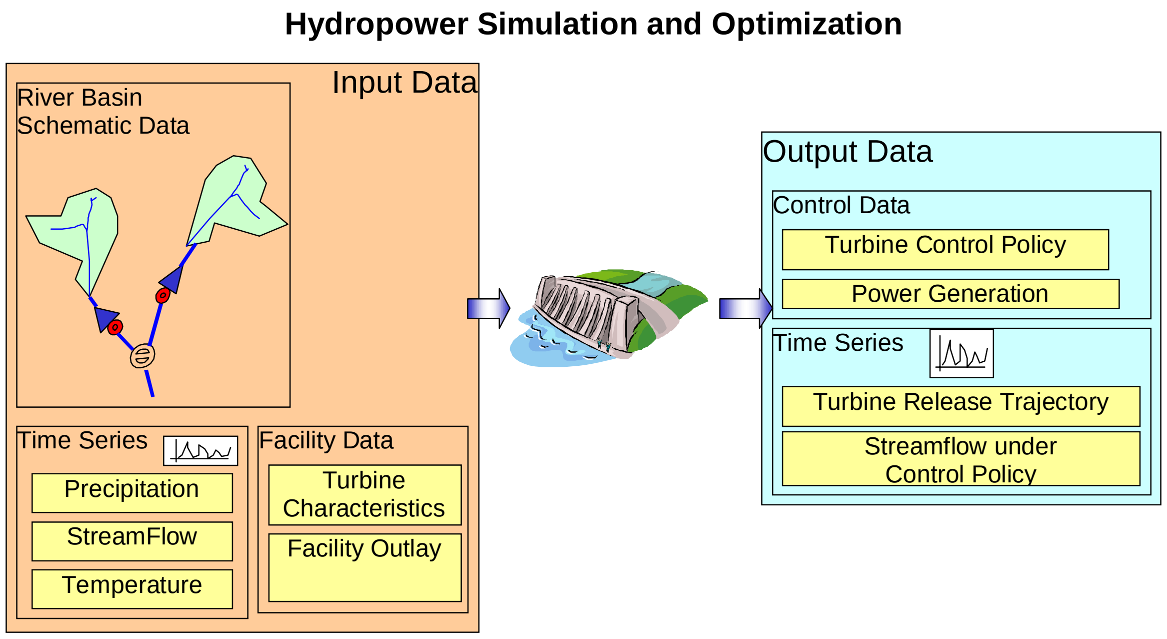


Figure 2, River Basin Management requires several forms of input data and generates several forms of output data.

Hydrology – Hydrologic Modeling requires a range of input data and generates many sets of output data. Input data includes time series of precipitation and temperature over the study watershed. Soil properties, elevations, and land cover for the watershed must also be known. Calibration of these models requires known streamflow as well. Outputs include modeled streamflow, runoff, soil moisture, and ground water flow. In the case of a distributed hydrologic model, these parameters must be known for all sub-basins in the model, and the spatial connectivity of the river network must also be known. Figure 3 shows a schematic of the data needs.

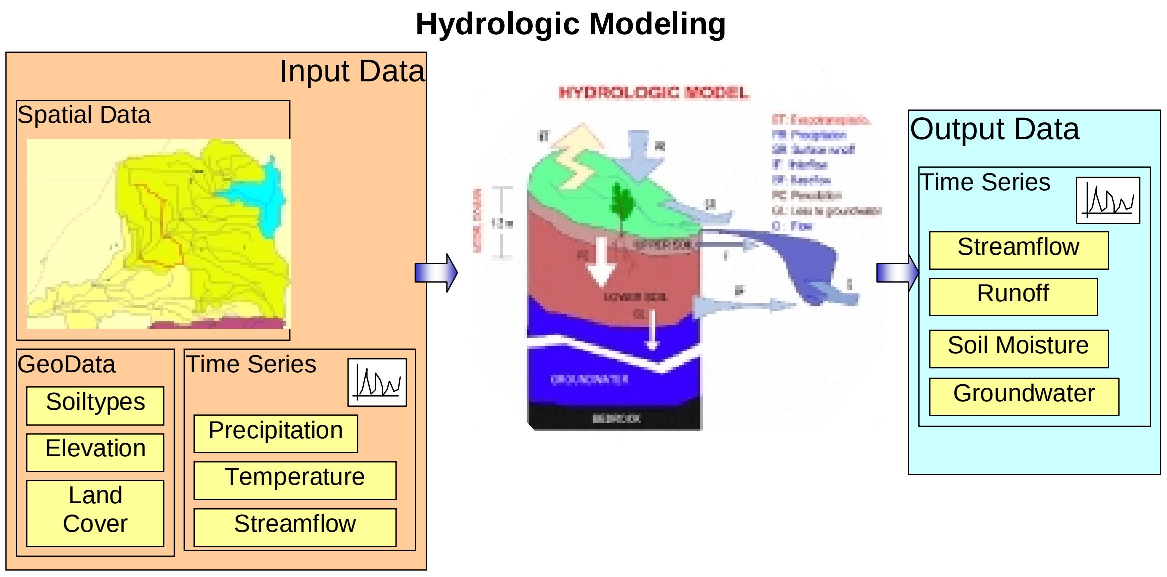


Figure 3, Hydrologic Modeling requires spatial and temporal input data and generates output of temporal data.

Agricultural Planning – Crop Simulation requires a range of input data and generates many sets of output data. The input data required describes the environment in which the crop grows and the type of crop that is being planted. Environmental parameters are the soil type, elevation, latitude, and meteorological data such as time series of precipitation, temperature, wind speed, and sunshine hours. Crop type data includes the proposed crop, cultivar type, and planting technique. The output data generated by the model includes the yield (both rainfed and irrigated), the recommended irrigation schedule according to the irrigation method used, and time series of biomass, drought stress, soil moisture, and Leaf Area Index (LAI). Figure 4 summarizes the data needs.

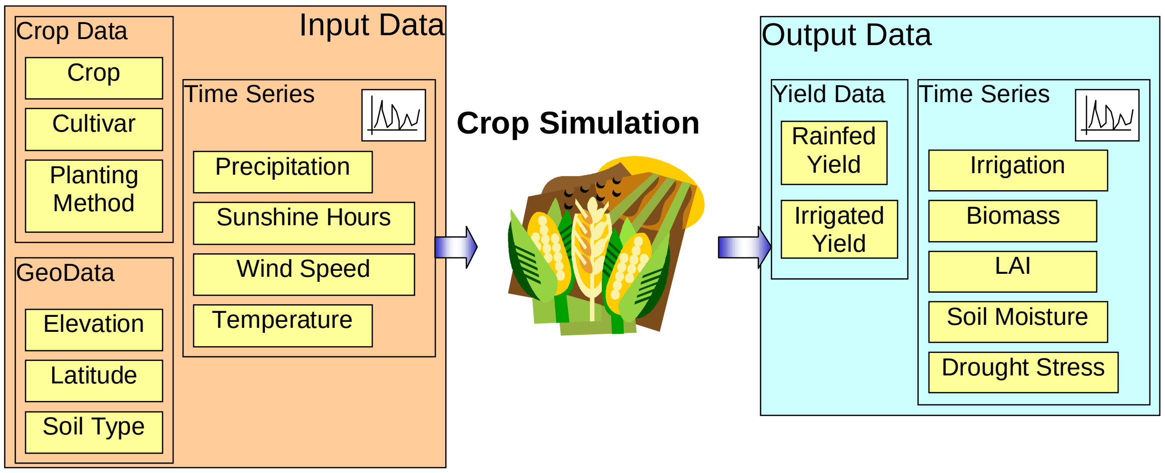


Figure 4, Agricultural crop modeling requires crop data, geo data, and time series as input and generates yield and associated time series as output.

Remote Sensing – Rainfall Area and Volume Estimation uses remotely-sensed electromagnetic radiation (EM) data to estimate rainfall areas and volumes. Large amounts of data are required for input for this modeling, as a relatively fine temporal resolution is used (approximately every 15 – 30 mins). Two-dimensional EM measurements of the entire spatial domain are required and must be stored and processed very quickly to produce model results. Output for the models includes two-dimensional maps of rainfall area and rainfall volume for the study period.   Figure 5 describes the process.

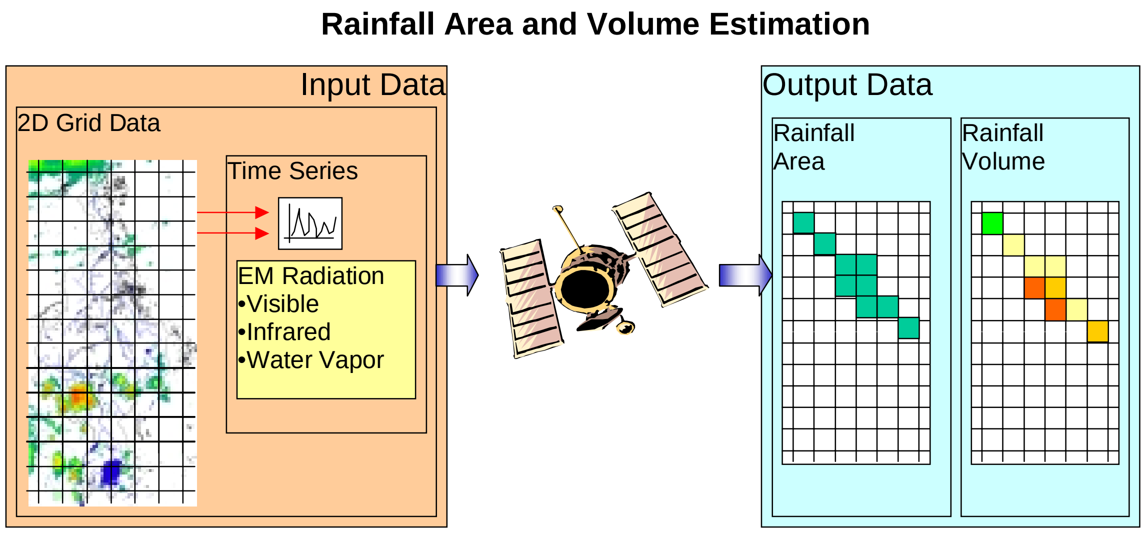


Figure 5, To estimate rainfall area and volume, a large data set of EM radiation images is required. Equally large sets of images are created to describe the output.

Data Analysis – Dataset Generation is frequently done when model input is required. An example is generating hydrologically and statistically acceptable time series of meteorological parameters like precipitation and evaporation for large areas like watersheds. This type of generation typically requires data from gauges. It also requires grids of reasonable resolution for spatial downscaling. Figure 6 outlines the data needs.

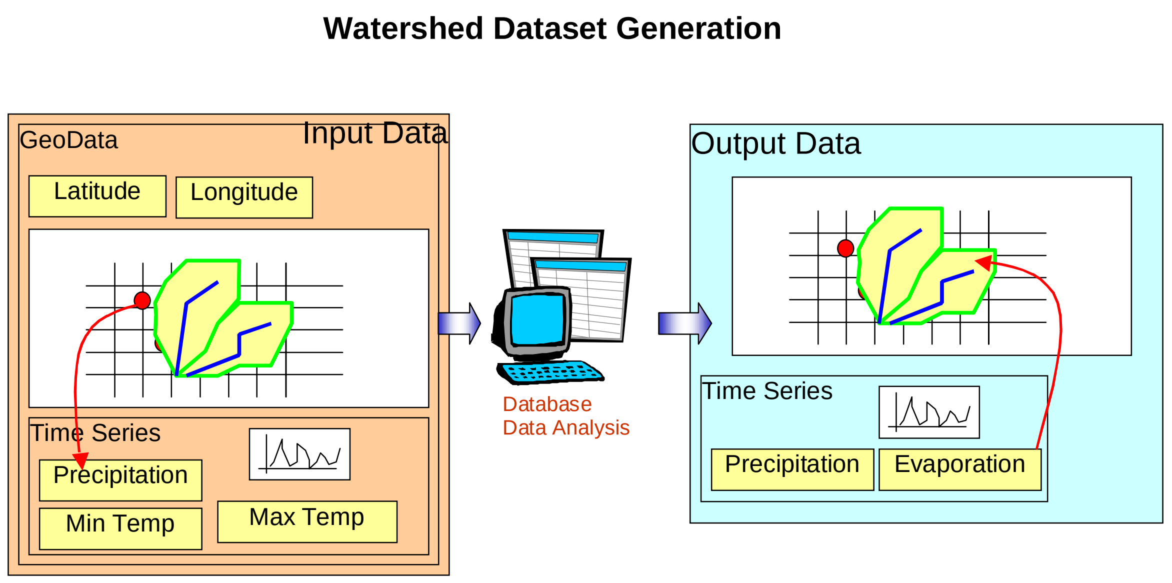


Figure 6, Generation of datasets of watershed-specific parameters such as precipitation and evaporation requires data measured at stations and intermediate grids for spatial upscaling.

The previous examples indicate there are some common data needs between applications. For example, almost all applications in the Nile DST will need station data. In some cases, only a subset of the applications requires a certain type of data. For example, river basin management and hydrology require the spatial layout of the river networks. The other modules do not. Finally, there are applications, such as remote sensing, that do not require any of the data the other applications use. Figure 7 below shows the interdependency of the applications in terms of input data types.

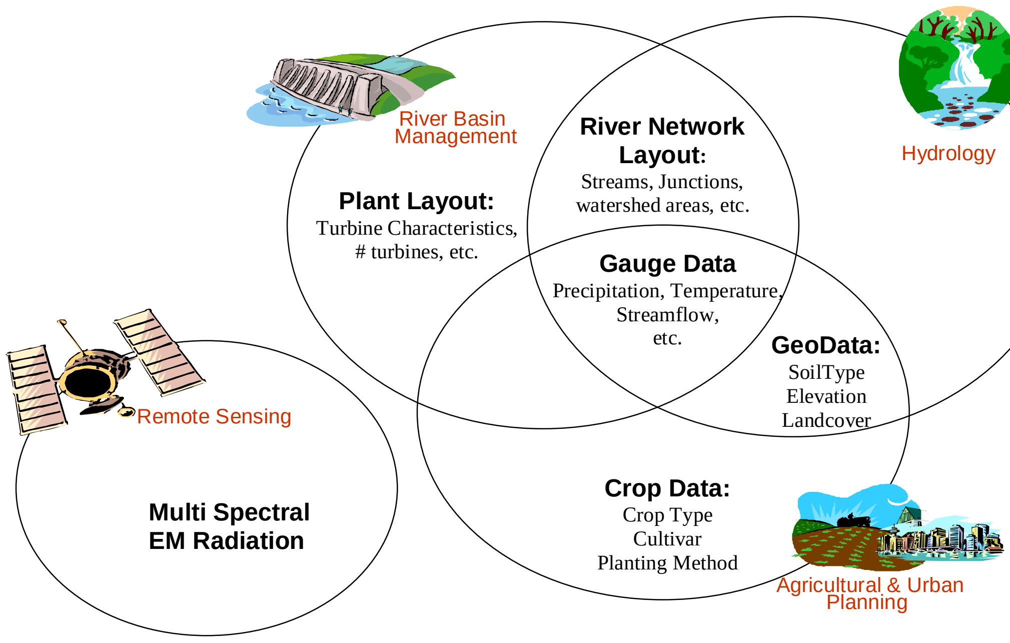

River Basin Management
Hydrology
River Network Layout:
Streams, Junctions, watershed areas, etc.
Plant Layout:
Turbine Characteristics, # turbines, etc.
Gauge Data
Precipitation, Temperature, Streamflow, etc.
GeoData:
SoilType
Elevation
Landcover
Remote Sensing
Multi Spectral EM Radiation
Crop Data:
Crop Type
Cultivar
Planting Method
Agricultural & Urban Planning

Figure 7, There is a great deal of overlap in the types of data required by the different applications..

2.2. How will the data be used?

There are three main ways the data will be used in the Nile DST: visualization, analysis, and application input/output.

Data Visualization is a major function of the database. Simply looking at the data can be extremely informative, especially when looking at maps that show the spatial distributions and variations of hydro-meteorological data. There are four key needs in terms of visualization:

- **Cataloging**: The Nile DST database is enormous, and it promises to get larger as new data is added in the future. It is essential that we concisely catalog the entire contents of the database. This will promote total awareness of the possibilities the database affords. Additionally, cataloging will reduce errors in database usage and augmentation as it promotes transparency in these processes.

- **Charting**: Visualizing time series of gauge station data, data analysis products, frequency curves, individual pixel time series for remotely sensed data, and pixel transects of remotely sensed data maps are just a few uses of data charting. The database design should provide tools for all these charting needs.

- **Mapping**: Because the majority of the Nile DST data is spatially georeferenced, mapping tools are required. At the simplest level, the tools should facilitate observation of geo-databases containing information such as land elevation, locations of streamflow and hydrometeorological gauges. In more advanced applications, the tools should produce maps like the one in figure 8. This is a map of maximum temperature for a given day. The stations have been colored according to how frequently their temperature is exceeded.

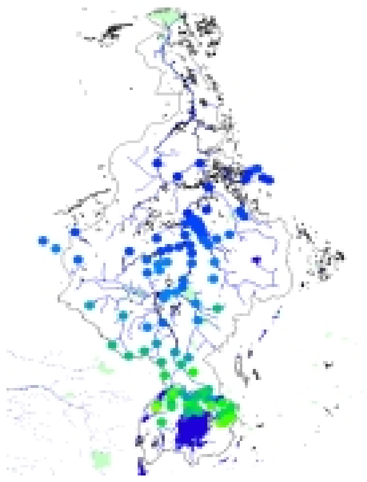


Figure 8, Mapping tools within the database should be able to visualize data products like this one. This is a map of maximum temperature, where the stations have been colored according to how frequently their temperature is exceeded.

- **Animating**: Charting visualizes a range of temporal data at a single point. Mapping visualizes a range of spatial data at a single instant. To observe natural spatio-temporal patterns another type of visualization is needed. Animation allows for visualizing the full development of spatially varying data over time. This is useful when searching for migratory patterns in parameter variance. For example, one can observe the movement and life cycle of rainfall generating areas in remotely sensed data. Another example is the migratory patterns of drought over large areas.

Data Analysis refers to generating meaningful products of data based on the original data in the database. This function is essential in the database as it allows the user to find clear patterns in the data and relationships between data types. There are many examples of data analysis:

- **Basic Statistics –** at the base level, the user would like to know simple statistics about the data he is using. For example, if the user suspects that some data at a gauge in the database may have been recorded erroneously, he can look for outliers by viewing maximum and minimum data points for that gauge. If the user would like to observe the difference in rainfall in two geographically distinct areas, he may look at the means of the data at gauges in those regions.   Possible statistics are the mean, median, variance, standard deviation, maximum, minimum and count of the data.

- **Arithmetic Products –** the user may also want to generate products based on the original data. For instance, to emphasize the extremes in a dataset it is often useful to look at the data to the second or fourth power. Another example of product generation is the simple addition of two time series datasets to make one. A full suite of tools should be available to allow the user to do these types of analyses.

- **Aggregation –** Aggregation refers to the grouping of elements into one representative element. For example, in a time series, one may want to aggregate data from daily values to monthly values. This is often useful as the important periodicities in time series that could not have been seen at high resolution (daily) can be seen at lower resolution (monthly).

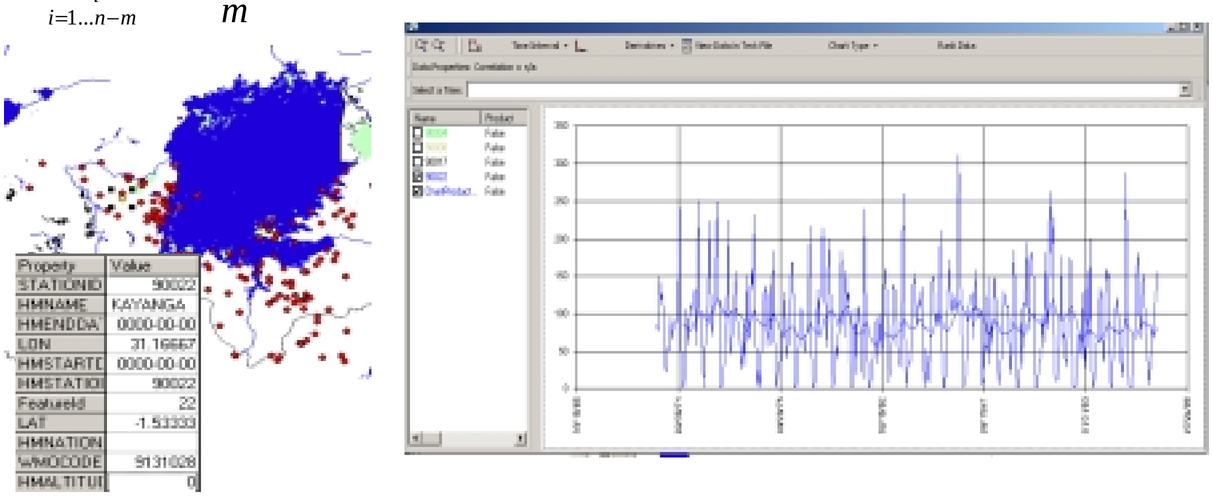
  Aggregation can be both spatial and temporal. An example is the conversion of a set of neighboring individual gauge data time series to one series for the entire area they cover. There are many methods of spatial aggregation. They range from simple arithmetic means (simply averaging the values of all stations for each time element), to inverse distance weighting (weighting each value in the spatial average according to how far away from the point of interest it is), to kriging (generating geospatial data that is statistically equivalent to the original data).

The sample data analysis that we saw previously is an example of spatial aggregation. To review, the recommended method is to aggregate the data from each station to an intermediate grid, the resolution of which is suitably fine relative to the target watershed. The watershed MAP is then calculated by using contributions from each grid cell over which the watershed lies. The contributions are weighted by the percentage of grid cell area covered by the watershed.

- **Filtering –** Filtering refers to the conversion of data series (especially time series) from their original state to another. Filter operations normally work on each element in the series (or a group of neighboring elements. Some examples of filtering are:

  Moving Averaging – moving averaging (or low pass) filtering creates a new series where each element is the average of the previous ***m*** elements in the original series, where ***m*** is the moving average period. Moving averaging is useful when we want to find low frequency periodicities. Often, natural systems have these periodicities. Figure 9 shows the moving average filter.

$$\underset{i=1\ldots n-m}{x_i'} = \frac{\sum_{j=i\ldots i+m-1} x_j}{m}$$ x' = new element, x = old element, n = number of elements, m = period



Property	Value
STATIONID	90022
HMNAME	KAYANGA
HMENDDA	0000-00-00
LON	31.16667
HMSTARTD	0000-00-00
HMSTATIO	90022
FeatureId	22
LAT	-1.53333
HMNATION	
WMOCODE	9131028
HMALTITU	0

Figure 9, Moving average filtering.

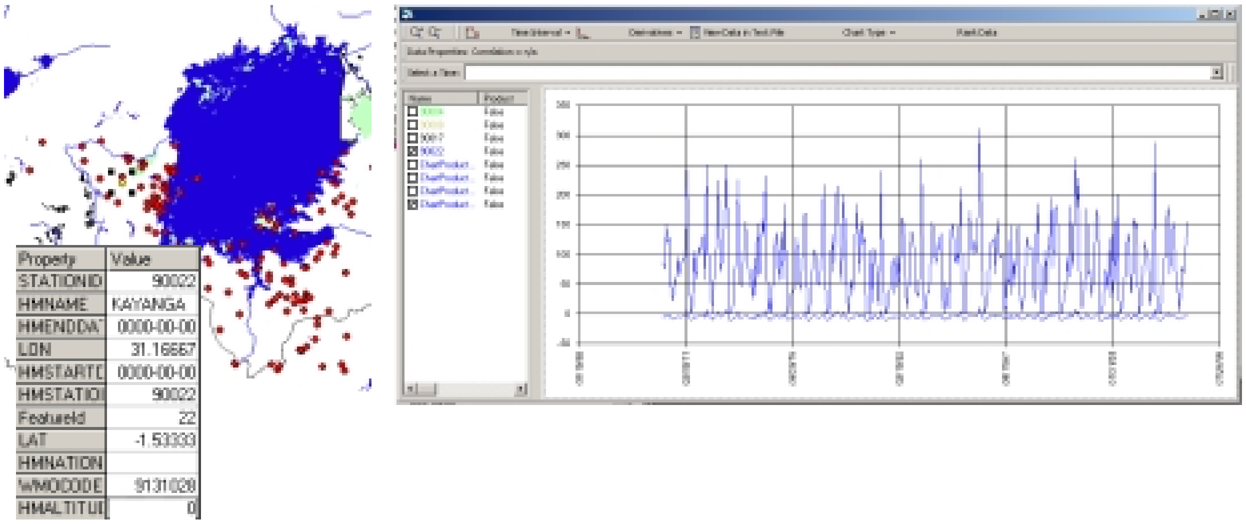
Cumulative Averaging – cumulative averaging is similar to moving averaging in that each element in the product series is the average of previous elements. In cumulative averaging, however, each product element is the average of all elements in the original series up to the current element.

Normalizing – Normalizing has several definitions. The definition presented here refers to converting the range of data in the original series to a new range. This is helpful in comparing two datasets which have drastically different value ranges (eg. tens vs. thousands). Figure 10 shows the normalizing process.

Anomalizing – Anomalizing data means subtracting each value in the data series from some mean value of the series. Because of seasonality in hydrometeorological data, it is often necessary to calculate means for each period in the year, where the period may be a month for example. Each element in the product series then becomes the value of the original element subtracted from the mean for corresponding period in the year. Figure 11 shows the animalizing process.

$$\underset{i=1\ldots n}{x_i'} = \left(\frac{x_i - x_{\min}}{x_{\max} - x_{\min}}\right)(x'_{\max} - x'_{\min}) + x'_{\min}$$

x' = new element, *x* = old element
n = number of elements



Property	Value
STATIONID	90022
HMNAME	KAYANGA
HMENDDAT	0000-00-00
LON	31.16667
HMSTARTD	0000-00-00
HMSTATIO	90022
FeatureId	22
LAT	-1.53333
HMNATION	
WMOCODE	9131028
HMALTITUI	0

Figure 10, Normalizing

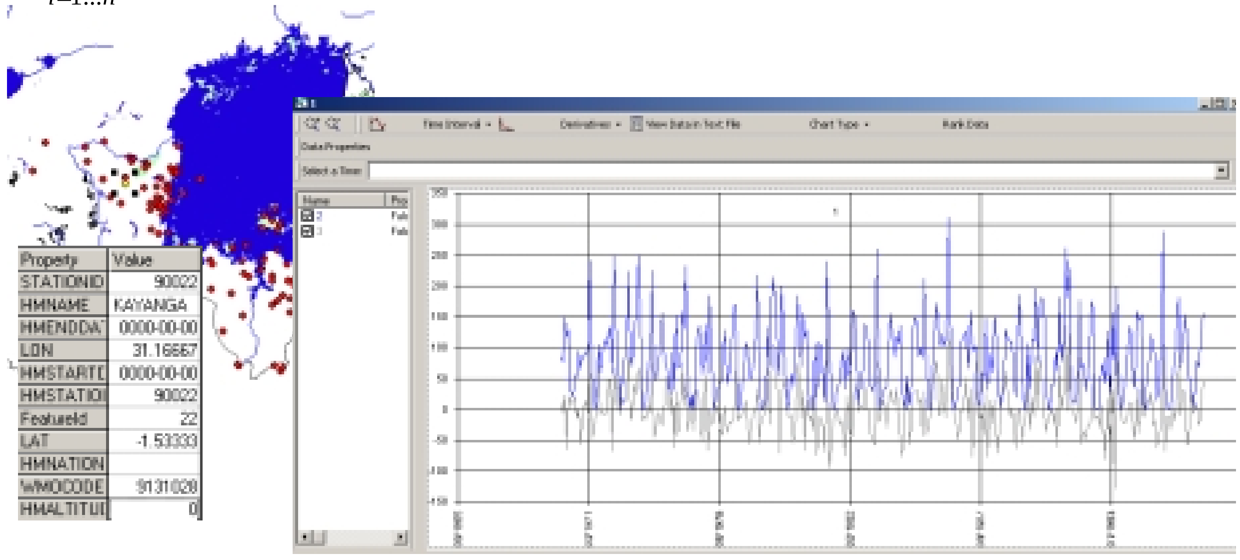
Thresholding – Thresholding data means isolating from the original series, that set of data that is more than or less than a certain threshold. The figure below, for instance, shows the data that has values equal to or less than the 30% of the original range of data. Thresholding is useful in studying extremes in data. For example, drought and flood periods are often reflected as extreme and sustained lows and highs in precipitation records. Isolating the extremes in the record can thus help when trying to understand these type of phenomena. Figure 12 shows thresholding.

- **Frequency of Exceedance Analysis –** frequency of exceedance analysis looks at data in its own domain. The set of data points in a time series represents the domain of values that elements from the series can take on. Therefore, one can compare a single value in the time series to the remaining points in the series and see how frequently that value occurs or is exceeded. To conduct a frequency analysis, first rank all of the data in the time series. Then assign plotting positions to each datapoint according to its rank with the cunane plotting position function (Figure 13).

  The plotting positions range from 0 to 1 (0% to 100%) and give the frequency with which an element's value is exceeded. In figure 13 is precipitation data for Kyanga, Tanzania. Both the data and the corresponding frequency curve are shown. Each value in the original data takes some position on the frequency curve. As is expected for precipitation, high values are concentrated within the lowest 10 percentage points in the frequency curve and zero precipitation (which is common) occurs most frequently (from ~40% to 100%).

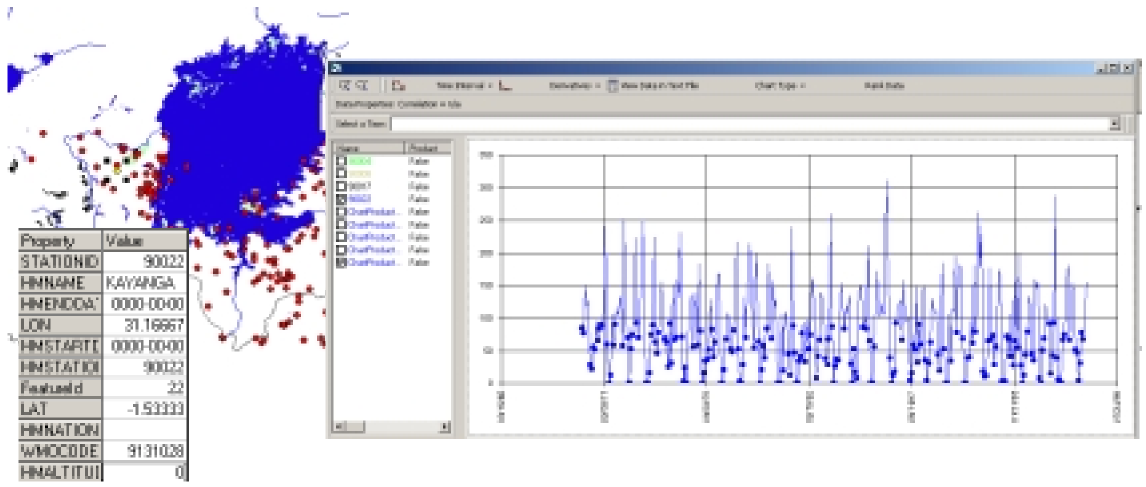
$$x_i' = x_i - \bar{x}_j,\ j = 1 \ldots 12$$
$i = 1 \ldots n$

x' = new element, x = old element, n = number of elements,



Property	Value
STATIONID	90022
HMNAME	KAYANGA
HMENDDA'	0000-00-00
LON	31.16667
HMSTARTE	0000-00-00
HMSTATIO	90022
FeatureId	22
LAT	-1.53333
HMNATION	
WMOCODE	9131028
HMALTITUI	0

Figure 11, Anomalizing



Property	Value
STATIONID	90022
HMNAME	KAYANGA
HMENDDA'	0000-00-00
LON	31.16667
HMSTARTE	0000-00-00
HMSTATIO	90022
FeatureId	22
LAT	-1.53333
HMNATION	
WMOCODE	9131028
HMALTITUI	0

Figure 12, Thresholding

$$\underset{i=1\ldots n}{x_i} = \frac{m - 0.4}{n - 0.2}$$

x = plotting position, m = rank, n = number of elements

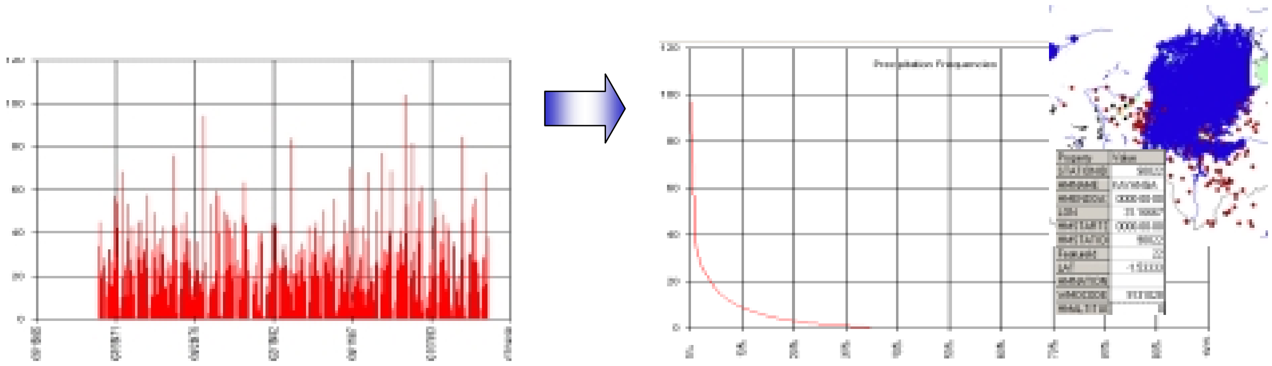


Figure 13, Frequency of Exceedance Analysis

- **Correlation Analysis involves looking for relationships between data of different types.** Often, natural systems exhibit similar behavior. This behavior is evidenced by correlated data series and can be quantified with the correlation coefficient between the series. The correlation coefficient, ρ, for series *x* and *y* ranges between 1 and –1 and is shown in figure 14. Figure 14 shows time series of maximum temperature for the Kyanga Station in Tanzania and for the Kosti Station in Sudan. The temperatures have been aggregated to yearly averages. Note first that the temperatures are different in terms of their average values. The Sudan temperatures range between 34 and 26 C, while the Tanzanian temperature range between 22 and 26 C (more than 10 deg. less). The correlation coefficient in this case is 0.263. This indicates a slight positive correlation. We might expect that the correlation would be higher, but due to the differing weather systems in these two areas, the correlation is low.

$$\underset{i=1\ldots n}{\rho} = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

ρ = correlation coefficient, n = number of elements

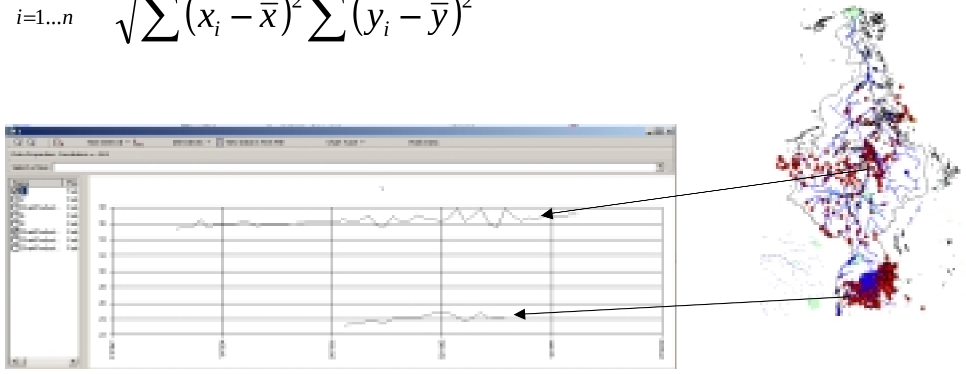


Figure 14, Correlation Analysis. Correlation between time series of maximum temperature for the Kyanga Station in Tanzania and for the Kosti Station in Sudan

Application Input/Output is the third major way in which data will be used. The applications in the Nile DST contain state of the art models that require large amounts of input data, formatted in specific ways. The applications output equally large amounts of data that must be visualized and analyzed. Figure 15 shows the similarities and differences in the applications' data needs.

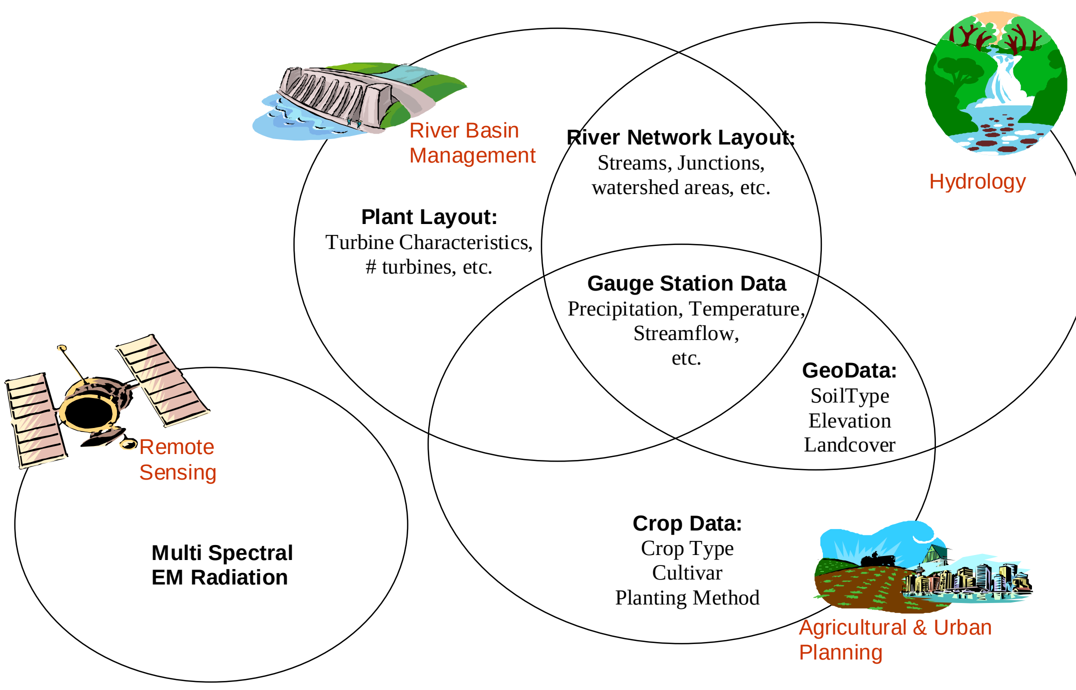

River Basin Management
Hydrology
River Network Layout:
Streams, Junctions, watershed areas, etc.
Plant Layout:
Turbine Characteristics, # turbines, etc.
Gauge Station Data
Precipitation, Temperature, Streamflow, etc.
GeoData:
SoilType
Elevation
Landcover
Remote Sensing
Multi Spectral EM Radiation
Crop Data:
Crop Type
Cultivar
Planting Method
Agricultural & Urban Planning

Figure 15, Applications' similarities and differences in their input and output.

In addition to simply having the data available, a good database design will have to be able to **preprocess** the data for the application. There are several examples of the need for preprocessing:

River Basin Management models often require data be at 10 day temporal resolution. Most of the gauge data in the database is recorded once per day. Therefore, analysis tools like the **aggregation** tool described previously are required to prepare a suitable input series for the model.

Hydrologic Modeling requires data to be associated with a watershed. Therefore, MAP datasets must be generated to use as input to the hydrology application. For this reason, the methods described previously for **spatial aggregation** are necessary.

Post processing is also required with application output. For example:

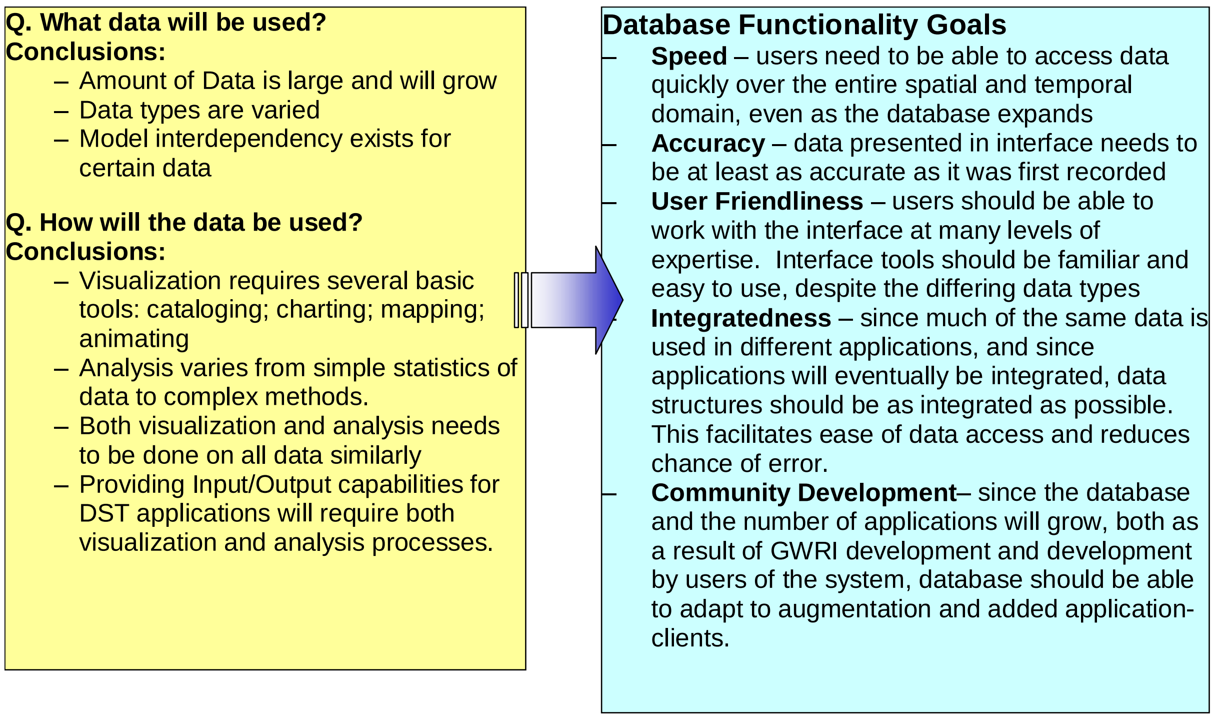
Agricultural Planning allows the user to create composite planting schemes. This means designing agricultural fields with more than one crop in them, or **intercropping**. Composite scheming uses the yield and irrigation data for each individual crop to produce the composite. This data is output from the crop simulation models and must be post processed to find the composite result.

The answer to, “How will the data be used?” is a complex one**.** It requires tools for visualizing data, analyzing data, and providing data to the Nile DST applications. The tools and applications shown in the previous pages illustrate a few examples of how the data will be used.

The more important point is that the previous pages only give a few of the possible examples. Since the Nile DST will be an evolving tool, a database design should not (and cannot) anticipate exactly the tools that will be required. The design should, however, provide a framework that can grow with the development of the Nile DST.

2.3. Design and Structure Goals

Addressing the two previous questions of what data will be used and how the data will be used leads to several clear goals that need to be met in creating a good database design. Figure 16 shows the development of the goals. Each goal will be described in turn below.



Q. What data will be used?
Conclusions:

– Amount of Data is large and will grow
– Data types are varied
– Model interdependency exists for certain data

Q. How will the data be used?
Conclusions:

– Visualization requires several basic tools: cataloging; charting; mapping; animating
– Analysis varies from simple statistics of data to complex methods.
– Both visualization and analysis needs to be done on all data similarly
– Providing Input/Output capabilities for DST applications will require both visualization and analysis processes.

Database Functionality Goals

– **Speed** – users need to be able to access data quickly over the entire spatial and temporal domain, even as the database expands
– **Accuracy** – data presented in interface needs to be at least as accurate as it was first recorded
– **User Friendliness** – users should be able to work with the interface at many levels of expertise. Interface tools should be familiar and easy to use, despite the differing data types
– **Integratedness** – since much of the same data is used in different applications, and since applications will eventually be integrated, data structures should be as integrated as possible. This facilitates ease of data access and reduces chance of error.
– **Community Development**– since the database and the number of applications will grow, both as a result of GWRI development and development by users of the system, database should be able to adapt to augmentation and added application-clients.

Figure 16, Several clear goals stem from the answers to the questions: What data will be used?, How will the data be used?

Speed is essential in a good geospatial and temporal database design. It is clear that the Nile DST database will be large. As more data is added and as more applications are developed, the database will grow exponentially. It is important that this growth does not impact on access speed for the database user. In an operating interface there are many processes that will be resource-intensive. These include, **Data Access**, **Visualization**, **Data Analysis**, and **Model Running**. The database design will affect the first three of these elements. Since these elements are the most used during interface use, it is important to make their running times as short as possible. Ideally, their running times should be imperceptible to the user. When they are not imperceptible, the interface should inform the user that a long process will take place. Figure 17 shows that the majority of the resource intensive processes are affected by database design.

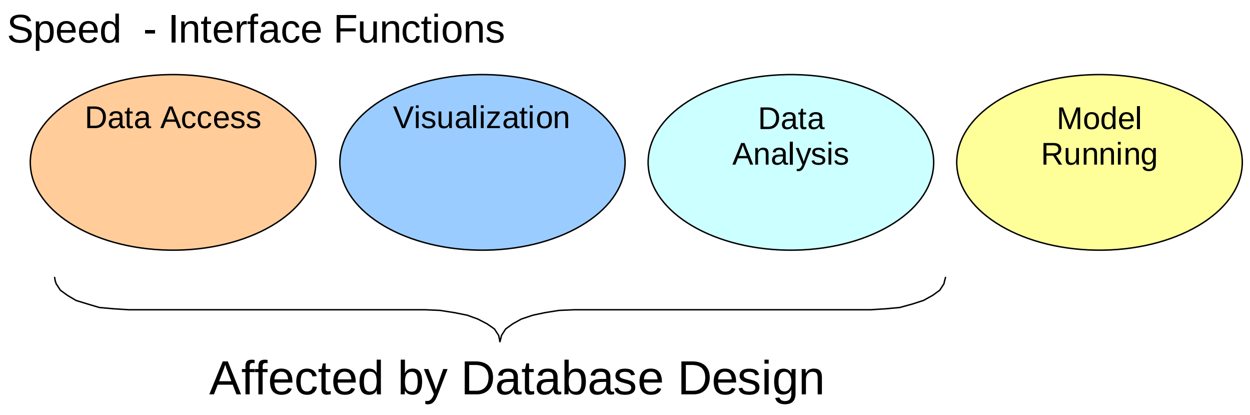


Figure 17, Speed of the interface is affected by good database design.

Accuracy – With the size of the database, it is tempting to use aggregated, averaged, filtered, or in some other way preprocessed data. This reduces hard drive space requirements and speeds up interface operation. This type of approach, however, compromises accuracy in the database and can lead to inaccurate results when the data is used in models for analysis. Additionally, since it is intended that the Nile DST database will be augmented frequently, it is most convenient that the data be kept in the form in which it was originally recorded. For these reasons, the database should house the data in the original form, at least as accurately as it was first recorded. Any analysis of the data for aggregation or averaging purposes should happen within the interface.

The term "at least" as accurately is used here because some data is recorded erroneously. In these cases, quality control measures should be introduced to remove or modify erroneous data. It is important to remember that many of the models in the Nile DST application will not work with erroneous data and so it is essential to maintain an accurate and acceptable database.

User Friendliness – It is clear that there will be some complexity to the database and interface that displays it. The differing data types will mean differing methods for visualizing and processing data. The data analysis methods range in complexity as do the models and their results.

In addition to this, there will be a range of user profiles. Users may include **Engineers, Scientists**, and **Water Professionals**, each of whom will communicate their DST analysis results to Basin Planners and Water Managers. It is important that the interface display the full range of results and make accessible the full range of applications to each user profile in an easy to understand manner.

Integratedness – Answering the question about what data will be used showed that many of the applications use the same type of data. In addition to this, many applications will be integrated to more accurately model the natural systems and facilitate better planning. Because the same data is often used, it would be redundant to provide data to different applications with different methods. For this reason a database design should be as integrated as possible so that the same methods that provide data for one application can provide data for another application.

Additionally, if all data is handled in this way, then errors in data input and generation are encountered only once and can be fixed at that point. This leads to vast reductions in quality controlling requirements and more accurate model results. Figure 18 shows this process.

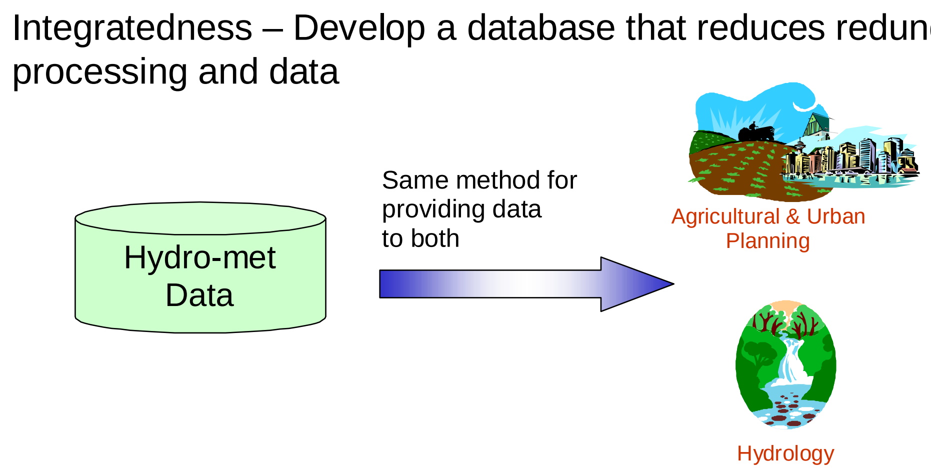

Integratedness – Develop a database that reduces redundant processing and data
Hydro-met Data
Same method for providing data to both
Agricultural & Urban Planning
Hydrology

Figure 18, Integrated database design reduces redundancy and speeds quality control.

Community Development – we expect the interface and database to grow, both through GWRI development, and through the development by its users. The database should be able to handle this growth seamlessly. A databasing structure should be equipped to add new data, both in terms of augmenting pre-exisiting databases and adding entirely new types of data. In addition, the database structure should be able to handle any number of application-clients. That is, as new applications are added to the Nile DST from the user community, the database should be able to provide those applications whatever data they need.

3. Database Design and Structure

Motivated by the goals established previously for the database's functionality, the design process became more clearly defined. Several key decisions and designs were considered with the goals in mind.

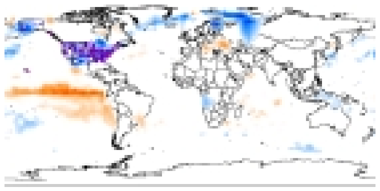
3.1. Application-specific vs. generic database

Perhaps the most important decision was the decision to move away from application specific databases (seen in the LVDSS database) and more toward a generic database, which is application independent. Though requiring more careful planning, this design is more dynamic and can accept new data types more easily. Furthermore, this design allows for centralized data quality control and reduces redundancy in the data.

3.2. Development of Custom Objects

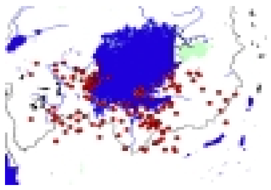
Given that the database structure will be generic, a generic method of representing data was required. To develop this method, a closer look at the character of the data to be held in the database was required.

Firstly the data has **multiple spatial domains and resolutions**. The gauge data in the database is country-specific. Since each gauge can be at any latitude or longitude, the scale is infinitesimal. The remotely sensed data will cover the entire Nile Basin at a resolution of 0.05 degrees. In the future more data will be added. This data will have ranging domains based on the area of interest. At the extreme, we may use datasets of meteorological data produced by global climatological models (GCM), for use in forecasting studies. This data has a global domain, with a very coarse 5 degree resolution. Figure 19 shows the varying spatial resolutions and domains.



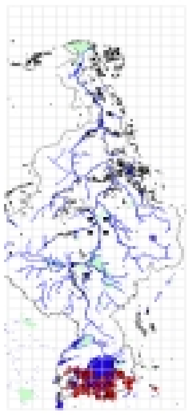
GCM Dataset of global Temperature

Spatial Domain - Global
ΔX = 1 deg. Lat
ΔY = 1 deg. Lon



Tanzania Hydromet Stations.

Spatial Domain - National
ΔX = Infinitesimal
ΔY = Infinitesimal



EM Radiation
Meteosat Broadcast Data

Spatial Domain – Nile Basin
ΔX = 0.05 deg. Lat
ΔY = 0.05 deg. Lon

Figure 19, Varying Spatial Domains and Resolutions.

The data has multiple temporal domains and resolutions. Figure 20 shows three possible temporal resolutions and domains Nile DST applications may work with. In the case of remotely sensed data, the remote sensing application will work with data form one day and will have a nominal resolution of 30 minutes. In the case of a derived data product like Lake Victoria Net Basin Supply, the data analysis application may require 100 years of data with a resolution of 1 year. In the case of gauge data, precipitation may be recorded at a gauge every day and the agricultural planning application will look at a one year time horizon.

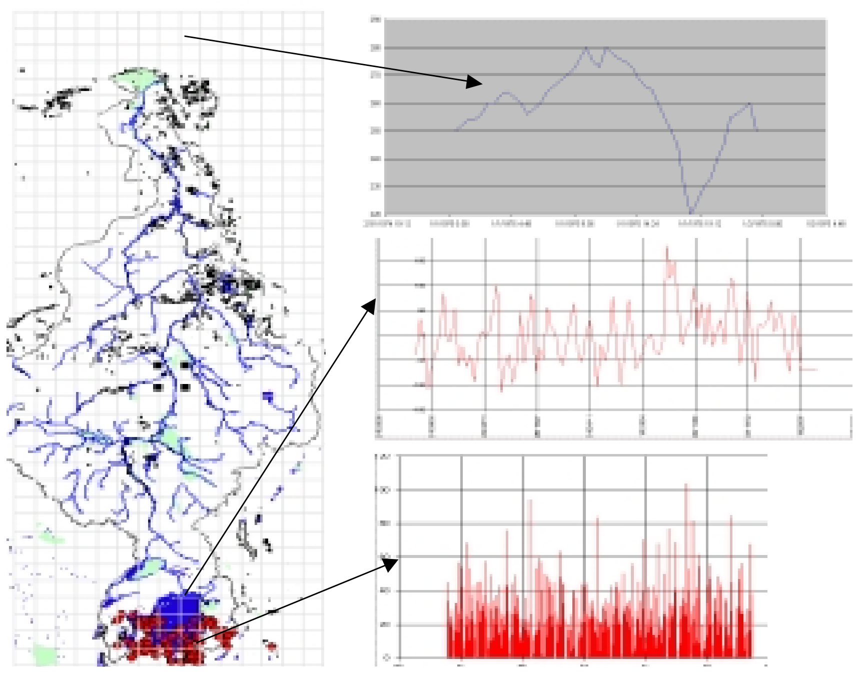


Figure 20, Varying Temporal Domains and Resolutions.

An additional issue with the data is sparcity. This refers to missing data in the data set. In many cases, the Nile DST gauge data was recorded for several years and then stopped for an extended period and then recorded again. In other cases, data is missing only for one or two days sporadically within the gauge dataset.

The character of the data indicates that the data structure will need to handle the following:

- Spatially georeferenced data
- Data with even and uneven spatial distributions,
- Data that covers spatial domains from single countries to the entire globe
- Data that has spatial resolutions from the continuum level (gauge data) to large areas such as five deg.
- Data with temporal domains ranging from one day to multiple centuries.
- Data with temporal resolutions from seconds to decades.
- Data that is sparse in its record.

To handle the issue of spatial georeferencing in the data, the interface will use **geographic information systems** (GIS) in the interface. GIS systems are capable of visualizing spatial data at continuum level. Since gauges are unevenly distributed, and in some cases very close together, this is this only way to fully represent the data.

Additionally, since datasets such as the remotely sensed data and GCM data are evenly distributed and represent the value for a spatial area (grid cell), the interface needs to house a visualization system that can represent these areas.   With its polygon representing capabilities, GIS can visualize such grids.

Over the past few years GISs have proven to be a very informative and user-friendly method of conveying spatial data. They are naturally integrated since they can show many sets of spatially distributed data at once.

To handle the remaining issues (spatial and temporal resolution and domain), an all-encompassing method of storing and processing data within the interface was developed. This method will ensure easy community development since it will provide a single programming and databasing structure that potential developers can learn.

A technique called object-oriented programming was used to develop data structures to store and process the data within the interface. The character of the data shows that the data can be split into two categories (or possible objects):

- **Gauge Stations (Ground Monitored (GM) data) –** these stations are always unevenly spatially distributed. They typically have national spatial domains and their data is normally daily data and is sometimes sparse. Because gauges have sparcity in their temporal records and are not spatially even, they have relatively few data.

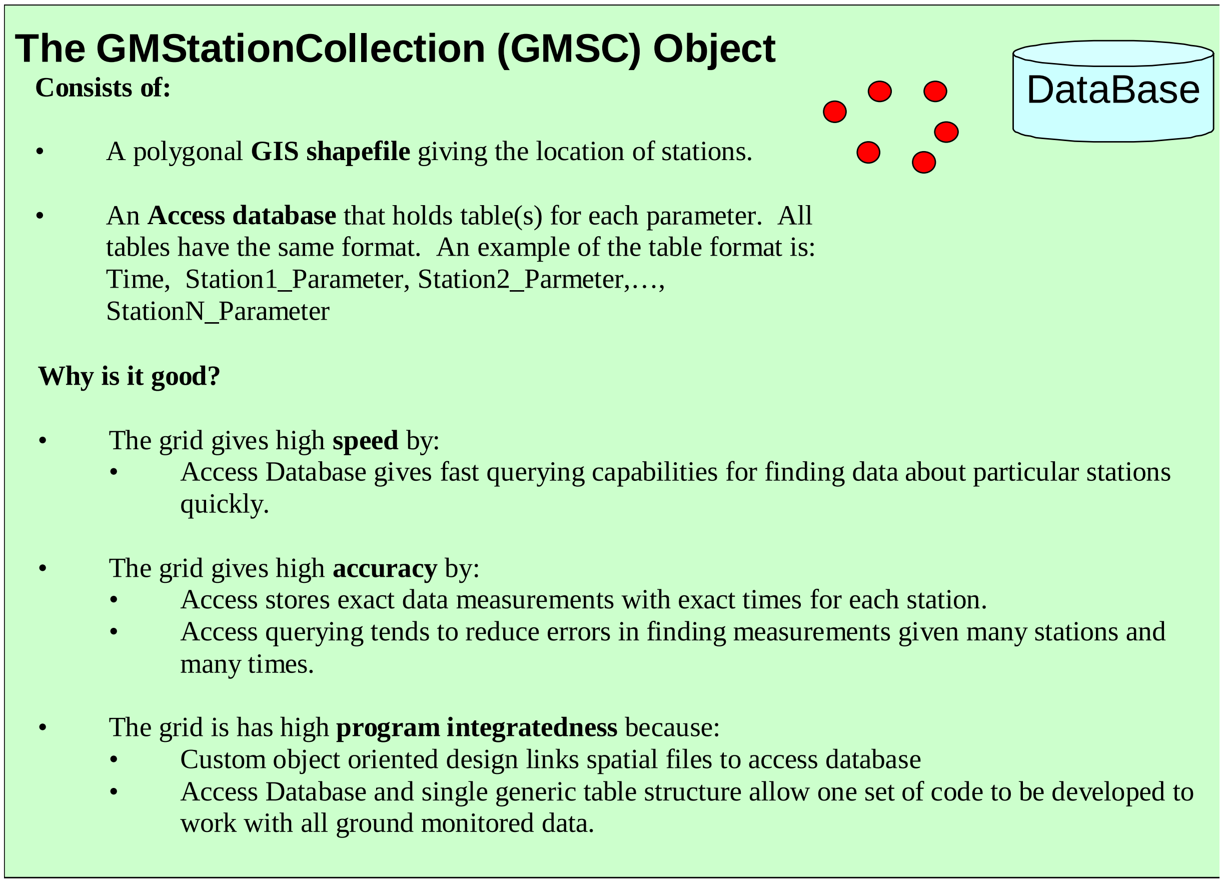
- **Gridded Data –** gridded data, like the remotely sensed data, is always evenly spatially distributed. Its domain can vary but normally ranges from regional (Nile Basin) to Global levels (GCM). Gridded Data is normally not sparse and its temporal resolution does not change. Because grids have evenly spaced data and complete temporal data, they normally have relatively many data.

It is therefore possible to generate two data objects, one for handling gauge data and one for handling gridded data.

- **Ground Monitored Station Collection (GMSC) Object. –** The GMSC object is shown in figure 21. The GMSC object handles gauge data. It is comprised of a collection of stations that represent each real ground gauge. It has a map property that refers to the map of stations that the user sees in the interface. It also is linked to a database system that holds the data for each station.   It helps to achieve the goals of accuracy, speed, and program integratedness.

- **The Grid Object –** The Grid object is shown in figure 22. The Grid object handles gridded data. It is comprised of a collection of grid cells that cover the grid's extent. It has a map property that refers to the map of grid cells the user sees in the interface. It is also linked to a database system that holds multiple binary files, which hold time series for each of the grid cells.

Figure 23 shows how the objects can be employed for each of the data sets the applications will need.



The GMStationCollection (GMSC) Object

Consists of:

- A polygonal **GIS shapefile** giving the location of stations.
- An **Access database** that holds table(s) for each parameter. All tables have the same format. An example of the table format is: Time, Station1_Parameter, Station2_Parmeter,…, StationN_Parameter

Why is it good?

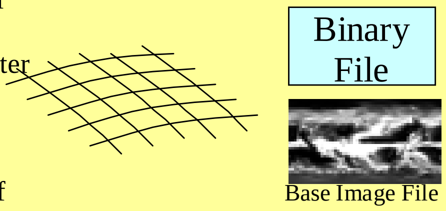
- The grid gives high **speed** by:
  - Access Database gives fast querying capabilities for finding data about particular stations quickly.
- The grid gives high **accuracy** by:
  - Access stores exact data measurements with exact times for each station.
  - Access querying tends to reduce errors in finding measurements given many stations and many times.
- The grid is has high **program integratedness** because:
  - Custom object oriented design links spatial files to access database
  - Access Database and single generic table structure allow one set of code to be developed to work with all ground monitored data.

Figure 22, Ground Monitored Station Collection Object.

The Grid Object

Consists of:

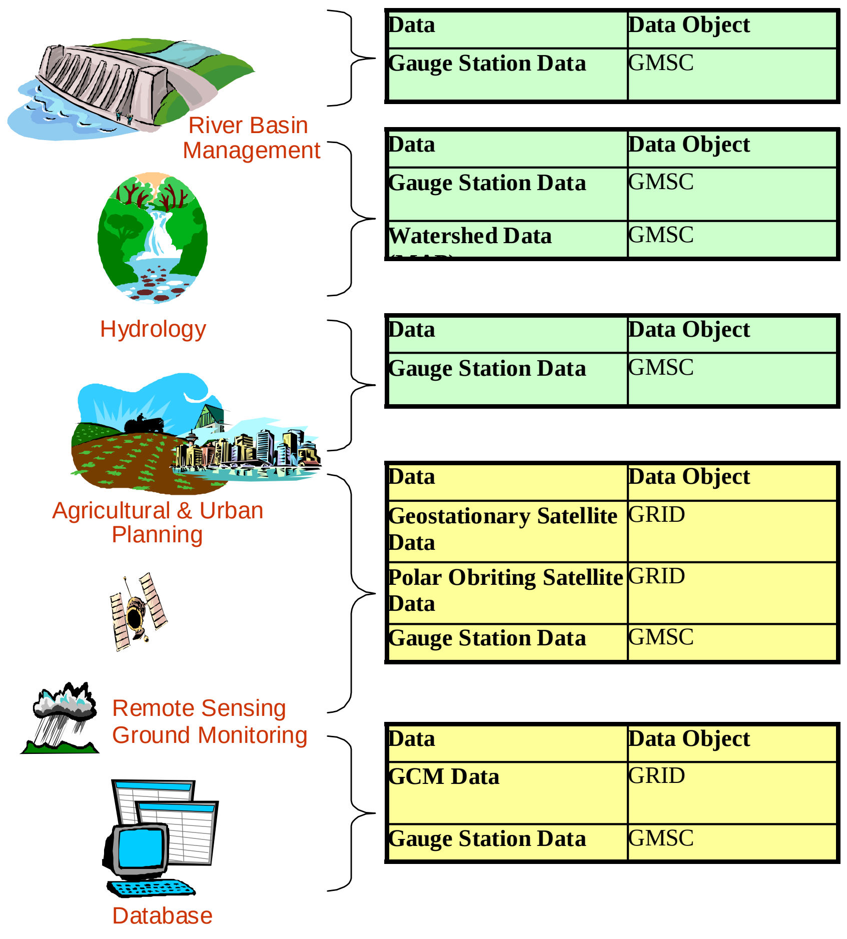
- A polygonal **GIS shapefile** giving the location and extent of all grids cells in the grid.
- A collection of **Binary Files** that hold the data for each parameter the grid has. These files are in a format specified in the remote sensing manual.
- A collection of **Base Image** bmp files, one for each time step, colored with 256 gray scales according to the expected range of values of the parameter of interest. And sized according to the number of grid cells in the grid.



Why is it good?

- The grid gives very high **speed** by:
  - Generating images with Base Image File. Image processing time is a function only of the number of gridcells in the grid.
  - Pulling time series for a given point quickly through SRTC format.
- The grid gives very high **accuracy** by:
  - Pixel value mapping
  - Text file storage of original data
- The grid is has high **program integratedness** because:
  - Custom object oriented design links text files to image files to grid's spatial properties.

Figure 21, Ground Monitored Station Collection Object.


River Basin Management
Hydrology
Agricultural & Urban Planning
Remote Sensing Ground Monitoring
Database

Data	Data Object
Gauge Station Data	GMSC

Data	Data Object
Gauge Station Data	GMSC
Watershed Data	GMSC

Data	Data Object
Gauge Station Data	GMSC

Data	Data Object
Geostationary Satellite Data	GRID
Polar Obriting Satellite Data	GRID
Gauge Station Data	GMSC

Data	Data Object
GCM Data	GRID
Gauge Station Data	GMSC

Figure 23, Employing the Objects.

Perhaps the most important feature of these two data objects is that they are generic. This means that they can hold data of almost any type. Therefore, as the database grows and new data types are added, there is no need to redevelop or even addend the database structure. For example, another module that could possibly be added to the database is the economic module. This module would use data such as time series of national market prices for grain. This data would be stored as a GMSC with each country represented as a gauge (GM Station). Another example might be using a climatological model where a gridded dataset of Sea Surface Temperatures is used. This could be represented with a grid object.

3.3. Developing an Interface that Uses the Objects

Generic objects have been created for reflecting the data's character. These objects can accomplish the goals of speed, accuracy, and integratedness. **The remaining task is to develop an interface that uses these objects?** We addressed this question by considering how the entire database will be structured and how the objects will fit into the structure.

Because there is a wide range of applications that may grow, the idea of a modular interface used in the LVDSS was used. Modular design means that custom interfaces can be built that hold only the modules that the user is interested in. For example, a user may want an interface with just River Basin Management. The interface can be created simply by including just the River Basin Management module in a new interface shell.

The **database is centralized**. This helps with integratedness and redundancy. As has been shown, a generic database structure is preferable given the likelihood of DST expansion in the future. Additionally, a central database means that new data is augmented only once and is immediately accessible by all modules in the interface. Centralization also means that quality controlling the database is centralized and, thus, less labor-intensive. Figure 24 shows a schematic of the Nile DST interface. Note that instead of many application-specific databases, one central database is used by all applications.

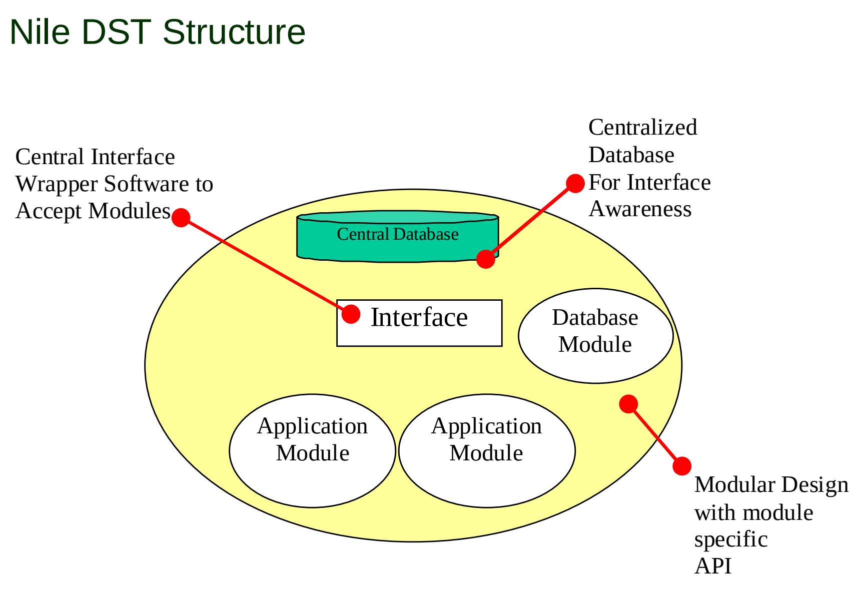

Nile DST Structure
Central Interface Wrapper Software to Accept Modules
Centralized Database For Interface Awareness
Central Database
Interface
Database Module
Application Module
Application Module
Modular Design with module specific API

Figure 24, The Nile DST Interface. Note there is one central database.

The **GRID objects** are held within the centralized database as shown in figure 25. Metadata about the grid is stored in a table in the metadata database (metabase). This metadata provides hard drive path information that leads the interface to the actual data files that hold the data. As the user requests images for the available times, the interface creates these images and adds them to the image directory structure.

Grid data is organized by time horizon and by parameter. Each binary file has a start time and end time and contains values of one parameter. The time horizon is the length of time between the start time and end time of the binary file. Thus, the metabase stores one row for each binary file that specifies the start time, end time and a parameter of the file. When the interface needs information, it just needs to know the parameter and time that has been requested. With the metabase, the interface can then find the right file to look in for the data and generate the image from this data.

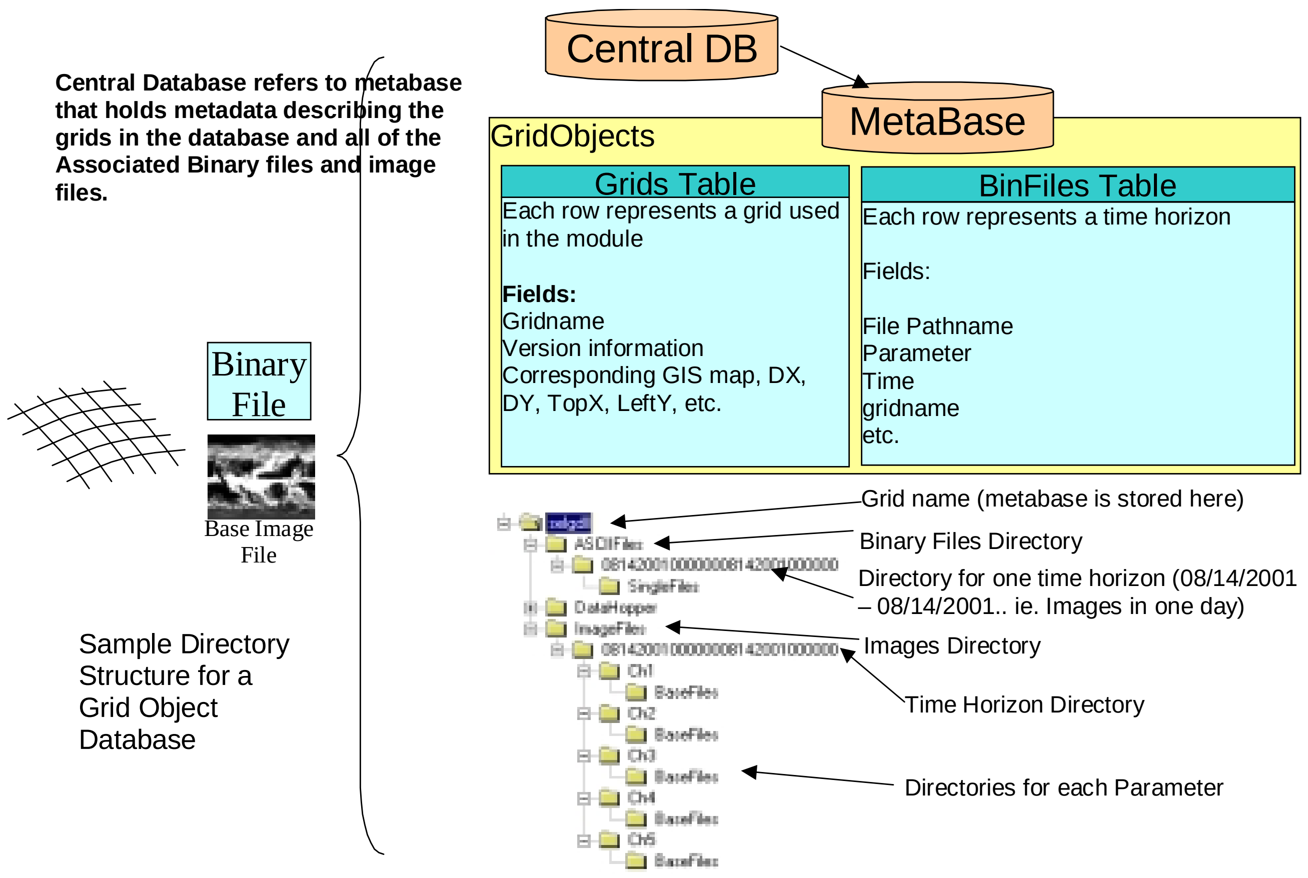

Central Database refers to metabase that holds metadata describing the grids in the database and all of the Associated Binary files and image files.
Central DB
MetaBase
GridObjects
Grids Table
Each row represents a grid used in the module
Fields:
Gridname
Version information
Corresponding GIS map, DX, DY, TopX, LeftY, etc.
BinFiles Table
Each row represents a time horizon
Fields:
File Pathname
Parameter
Time
gridname
etc.
Binary File
Base Image File
Sample Directory Structure for a Grid Object Database
ASCIIFiles
081420010000000081420010000000
SingleFiles
DataHopper
ImageFiles
081420010000000081420010000000
Ch1
BaseFiles
Ch2
BaseFiles
Ch3
BaseFiles
Ch4
BaseFiles
Ch5
BaseFiles
Grid name (metabase is stored here)
Binary Files Directory
Directory for one time horizon (08/14/2001 – 08/14/2001.. ie. Images in one day)
Images Directory
Time Horizon Directory
Directories for each Parameter

Figure 25, A grid database. The database is comprised of a metabase and directory structure. The metabase holds the pathnames of files for each parameter. These files are stored in the directory structure. The images are created as the user requests them from the database.

The **GMSC objects** are held within the centralized database as shown in figure 26. For each GMSC, there are two records in the central database's Database Preferences table. One record gives the associated GMSC map. The other record gives the path of a meta data database (metabase). This metabase in turn contains a record for each station and each parameter that was recorded at the station. Useful information such as the number of records, temporal resolution, start date and end date are stored in the metabase. Also stored is the path of the database that holds the data for the station and the name of the table that holds the data.

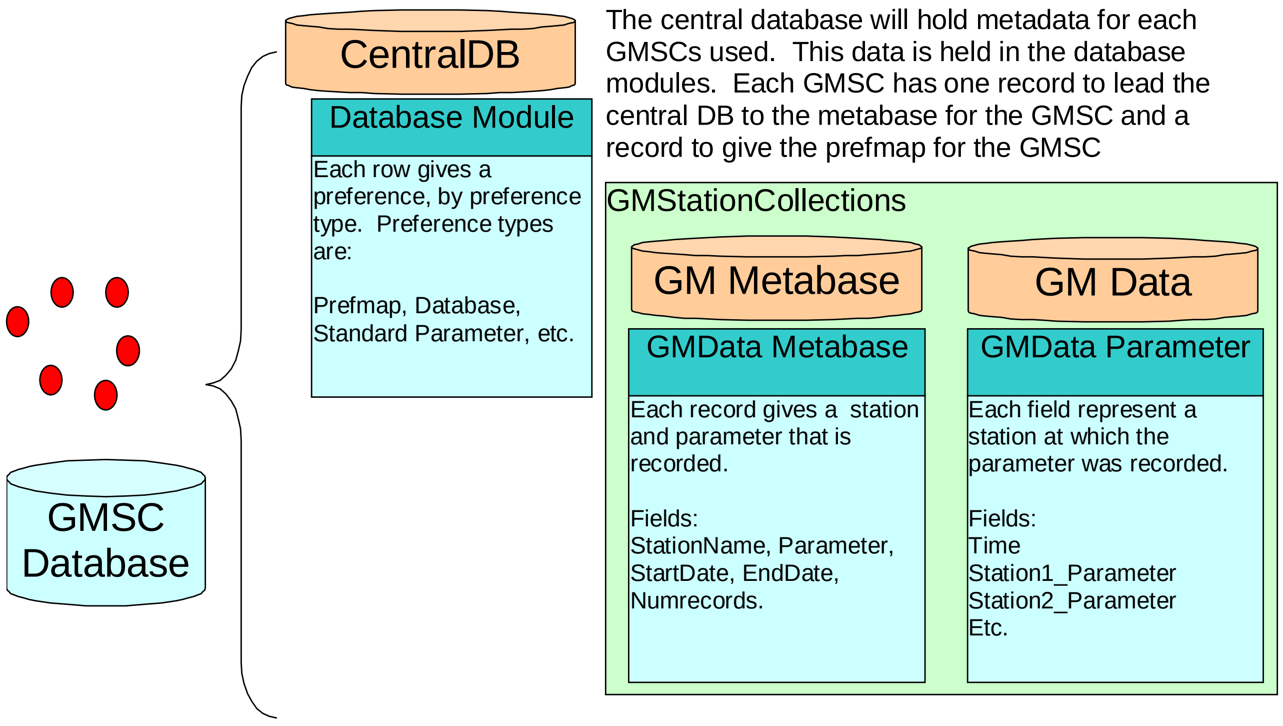

CentralDB
Database Module
Each row gives a preference, by preference type. Preference types are:
Prefmap, Database, Standard Parameter, etc.
The central database will hold metadata for each GMSCs used. This data is held in the database modules. Each GMSC has one record to lead the central DB to the metabase for the GMSC and a record to give the prefmap for the GMSC
GMStationCollections
GM Metabase
GMData Metabase
Each record gives a station and parameter that is recorded.
Fields:
StationName, Parameter, StartDate, EndDate, Numrecords.
GM Data
GMData Parameter
Each field represent a station at which the parameter was recorded.
Fields:
Time
Station1_Parameter
Station2_Parameter
Etc.
GMSC Database

Figure 26, A GMSC database. The database is comprised of a metabase and database. The metabase holds a record for each stations/parameter pair and tells the interface where in the database to go to find the information for the station and parameters it is looking for. The database holds the data in parameter-specific tables.

The database design gives a **data flow** through the interface. This data flow is shown in figure 27. Each time data is requested, the queries must find the correct path to go to through the metabase and proceed on to the target database. A result of this is that the central database only stores a minimum amount of metadata about any database that is added to the interface. Since this data is at a minimum, many databases can be plugged into the interface without affecting performance time in terms of data access. The interface can thus have a large number of databases and access any data in these databases very quickly.

Data Flow in the Nile DST

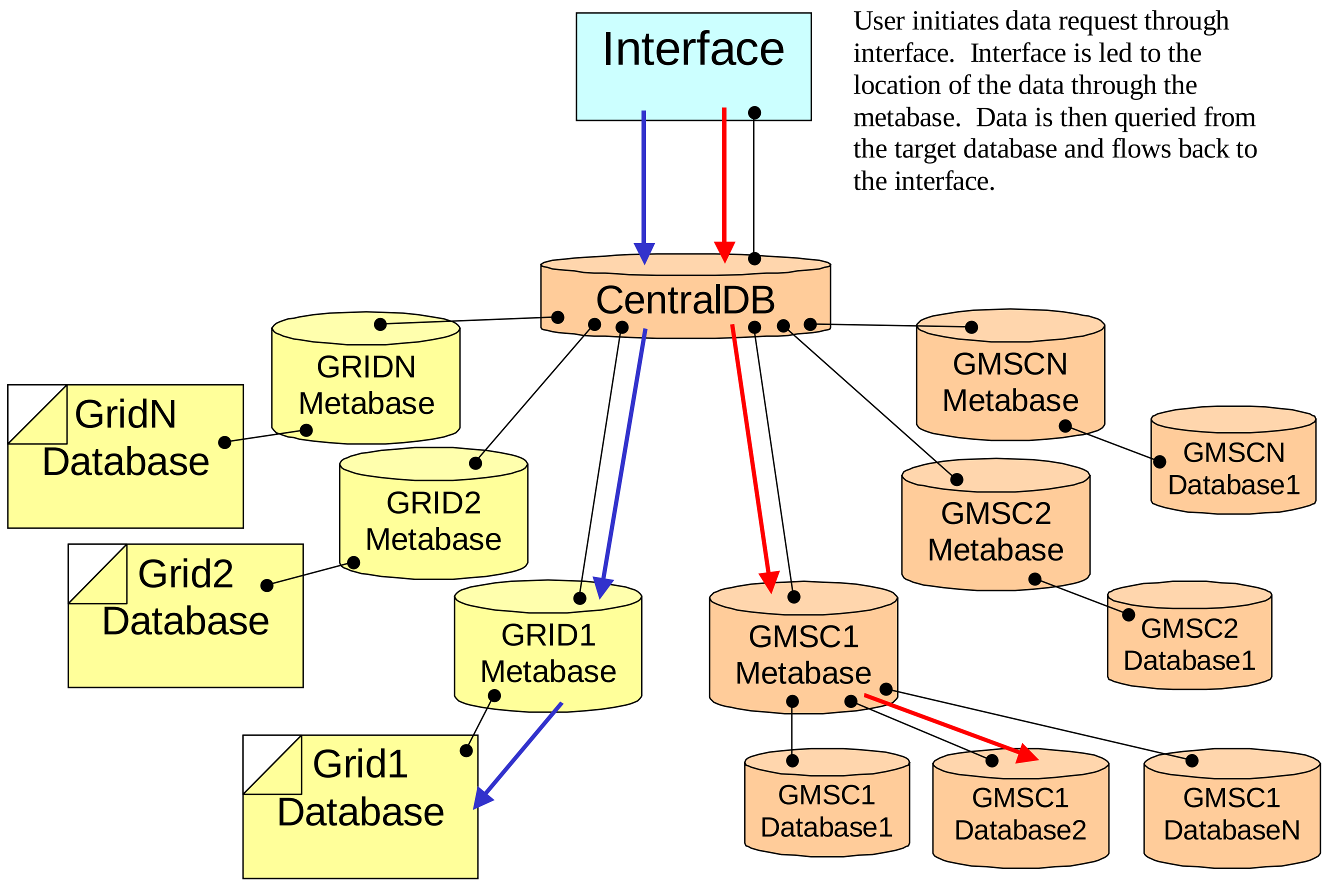

User initiates data request through interface. Interface is led to the location of the data through the metabase. Data is then queried from the target database and flows back to the interface.

Figure 27, Data Flow in the Nile DST Database. The hierarchical database structure means there is a data flow path each time the interface requests data from the interface to the central database to the metabase to the target database and back. This structure means a large number of databases can be added to the system without adversely affecting data access speeds.

4. Conclusion

Through a comprehensive analysis of the database and interface requirements, a generic design for the interface and database were created. This design took into account goals of speed, accuracy, integratedness, user-friendliness, and community development. The design is generic enough to accommodate new data in the future and it provides a good foundation for expansion of the DST.