Methodology for Flow and Salinity Estimates in the Sacramento-San Joaquin Delta and Suisun Marsh

23rd Annual Progress Report
June 2002

Chapter 13:
DSM2 Input Database and Data Management System

Author: Eli Ateljevich and Tawnly Pranger

13 DSM2 Input Database and Data Management System

13.1 Introduction

A feature of DSM2 since its inception has been its flexible text input system, which improved upon the default "fixed-width FORTRAN" restrictions of previous models. There are a few shortcomings that have not been addressed, however. One is a lack of transparency. The current system allows for an excessive number of ways to accomplish some tasks. The inner workings of the model reflect this complexity as well – a significant amount of the source code is dedicated to handling switches, options, priorities, and text substitution features. A second difficulty with the text system is the problem of data management. DSM2 text input is not fully compatible with database storage nor is it easily converted to one of the newly established text standards for data such as XML. This makes version control and data standardization efforts difficult to automate using modern tools. For some kinds of studies, the standard for data acquisition is still to copy files from a colleague or previous project.

The DSM2 database project marks a transition towards more rigorous data management. This is a multifaceted effort:

- DSM2 has been reprogrammed to accept connections directly to a database,
- A normalized database structure has been devised for the input,
- An application (the "user interface") has been created for viewing and editing the data, and
- A replication scheme has been planned that will ensure data consistency across a network of users.

At the same time, every effort has been made to retain flexibility features from the original text input system that DSM2 modelers are known to use often and correctly.

In terms of data management, the goals for the DSM2 database project are as follows:

- Data standardization: users should be able to synchronize to a single, "best" version of standard input packages across a network without any bookkeeping or responsibilities on their part. The private passing of data from user to user should be discouraged (mechanically).
- Reusability: users should be able to work with bundles of standardized, frequently used components (e.g., the "Standard Delta Grid" or "South Delta Permanent Barriers").

- Security: the most important standard input should be administered and version controlled.

- Reproducibility: the user should be able to access data from the past (e.g. “as it was on March 15, 2002”).

- Externalization of quality control: data validation rules should be incorporated into the database and by the user interface. This is more efficient, reliable, and visible than the black box quality checking that goes on every time DSM2 is run.

- Modernization: the data management system should make use of high quality technology, including proprietary database applications and libraries. It is this commitment that makes such a large laundry list of goals possible under the umbrella of a single project.

The remainder of this chapter describes tools and methods that help meet these goals. Most of them are currently functional and are in beta testing at the time of writing. The chapter is not intended as a user’s manual for individual components, but rather to describe their contribution to the data management strategy.

13.2 Database and Data Management

The heart of the project is the DSM2 input database. Water resource engineers in California are accustomed to applying the term “database” to anything from a few tables in an Excel spreadsheet or Access desktop database file to large institutional databases such as the IEP data vaults or CDEC. In the case of the spreadsheet or desktop database, the data is actually on file in the user’s computer, and if the user manipulates the information and passes it to another user no data management has occurred. At the other extreme, users querying IEP or CDEC databases have no administrative control of the data. Access to the database is offered by means of custom web pages or software agents. In fact, the implementation details of the database are not known or important to the user.

The DSM2 database lies between these extremes. From the user’s point of view, the database appears like a Microsoft Access desktop database. The user will also have local copies of the DSM2 user interface and DSM2 numerical model software, which interact with the desktop database independently, as indicated in Figure 13.1. The user interface provides a graphical environment for editing data and preparing simulations. DSM2 queries this data at runtime, communicating with the database on a “read-only” basis.

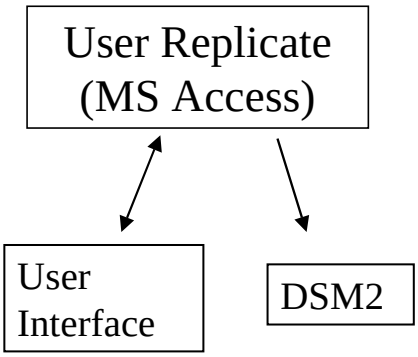

Figure 13.1: Single User Interface for Editing Data and Creating Simulations.

As implied by Figure 13.2, the desktop database is also a *replicate* in a larger, enterprise-wide data synchronization scheme. In a database context, the term *replication* refers to the sharing and synchronization of data within an organization or wider community. There are as many replication models as there are *business rules* for sharing data. The replication model for DSM2 is a consolidation model. The database *master* is a repository of the most important simulation data to which the replicates have full read access and limited (protected) write access.

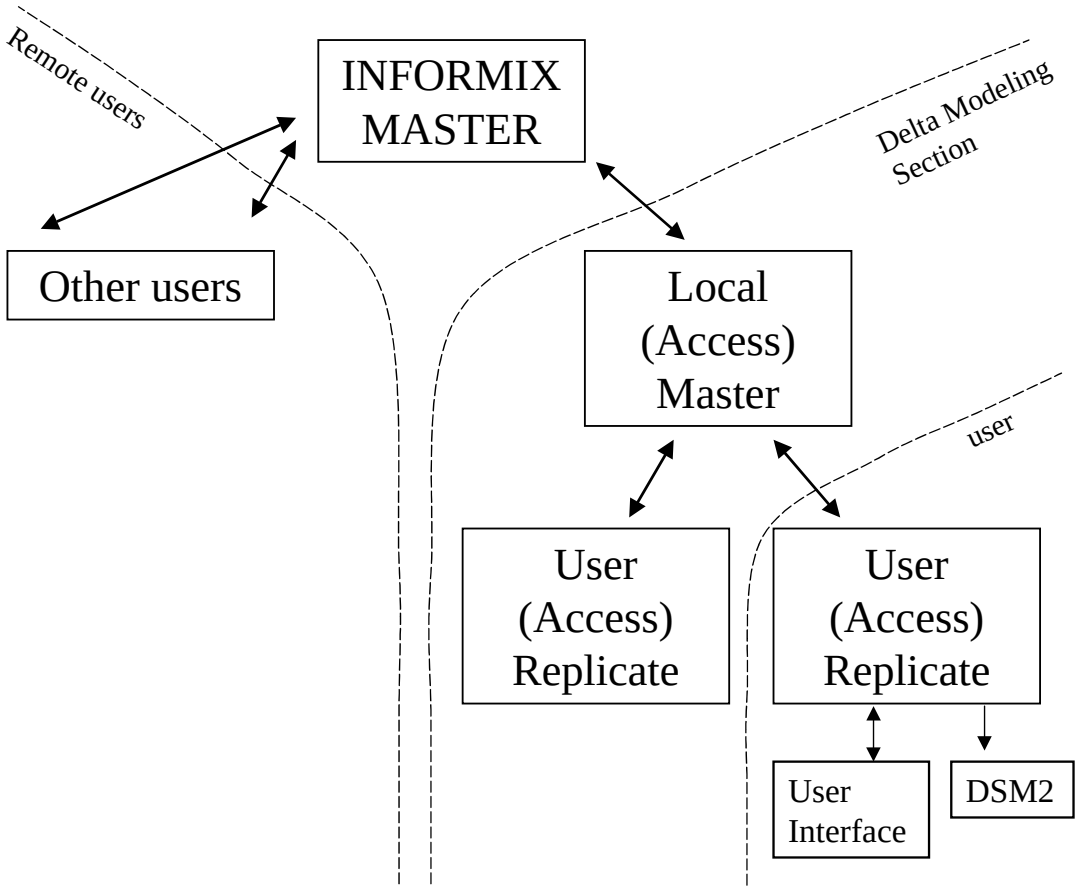

Figure 13.2: Single User in Context of a Local (MS Access) Replicate System and a Remote (Informix) Replication System.

Within the Delta Modeling Section, replication can be achieved almost entirely with Microsoft Access features. The user receives a replicate of the master database, which is a nearly full-function desktop database (the limitations can be found in the MS Access documentation; replicate status cannot be altered). The use of the Informix component has introduced two additional needs. First, it is desirable to have an Internet-capable server in order to share DSM2 data with the wider Delta modeling community. Second, Informix is an industrial strength "transactional" database that is able to log and timestamp every change made to it. Later in this chapter it will be shown that these timestamps are the key to an entirely different notion of "replicability" – the ability to reproduce earlier simulations.

The link between Informix and the Microsoft Access local master is accomplished with a Visual Basic module. Both the administration of the Informix database and the link to the local master are being carried out with the assistance of IEP database specialists.

13.3 Database Structure

The tables in the database are based on the standard relational model. The attributes of model elements such as channels and gates are stored in separate tables, using one column (also known as a *field*) per attribute. When data from different tables need to be brought together or cross-referenced, the tables can be joined during retrieval using a common field. For example, assume the database contains two tables, Channel and GridDescription, with the following fields:

Channel	**GridDescription**
ChanID	GridID
GridID	GridDescription
Length	Creator
Manning	
Dispersion	

Note: these listings are a simplification of the tables in the database.

The two tables can be joined on the GridID field to produce a result set containing information from both tables, for example ChanID, Length, and GridDescription.

In order to facilitate joins and queries, tables in the DSM2 Input Database conform to *third normal form*. Table normalization is the adherence to design rules that minimize data redundancy and prevent "anomalies" during queries (orphaning of records, contradictory entries, etc.). There are five levels of normalization described in standard texts, each indicating a stricter set of rules; third normal form is usually considered adequate by industry standards. One of the most visible consequences of data normalization is that where "one-to-many" relationships are present, a hierarchy of tables is required. A good example is the channel hierarchy. One Channel may have many Cross-sections. And each Cross-section may represent many Layers of geometry data. The hierarchy for Channels is shown in Figure 13.3.

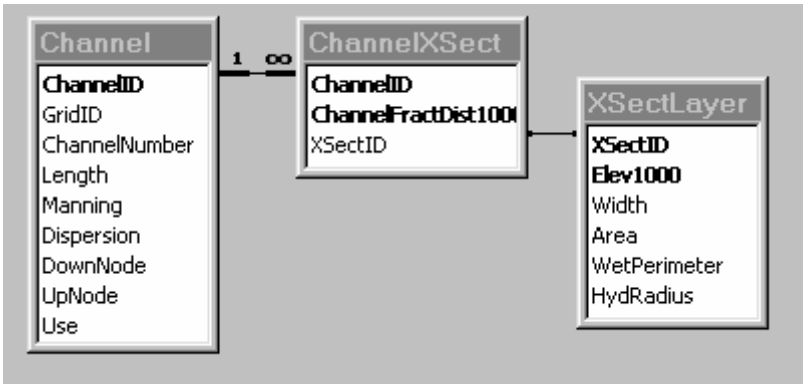

Figure 13.3: Hierarchy of the Channel Table.

Normalized tables are not intuitive to read, but they allow data from different tables to be *joined* or *linked* flexibly to form a variety of useful views. This preference of many potential views over one good immediate view underscores an important aspect of database design: **separation of the table design and the "view" of the data**.

Joining tables to form new views was described in the preceding paragraphs; *linking* tables is illustrated in Figure 13.4, a sample from the DSM2 User Interface. The tables in the top table are linked to those in the bottom panel. If the user selects a channel in the top panel, related cross-section data automatically appear.

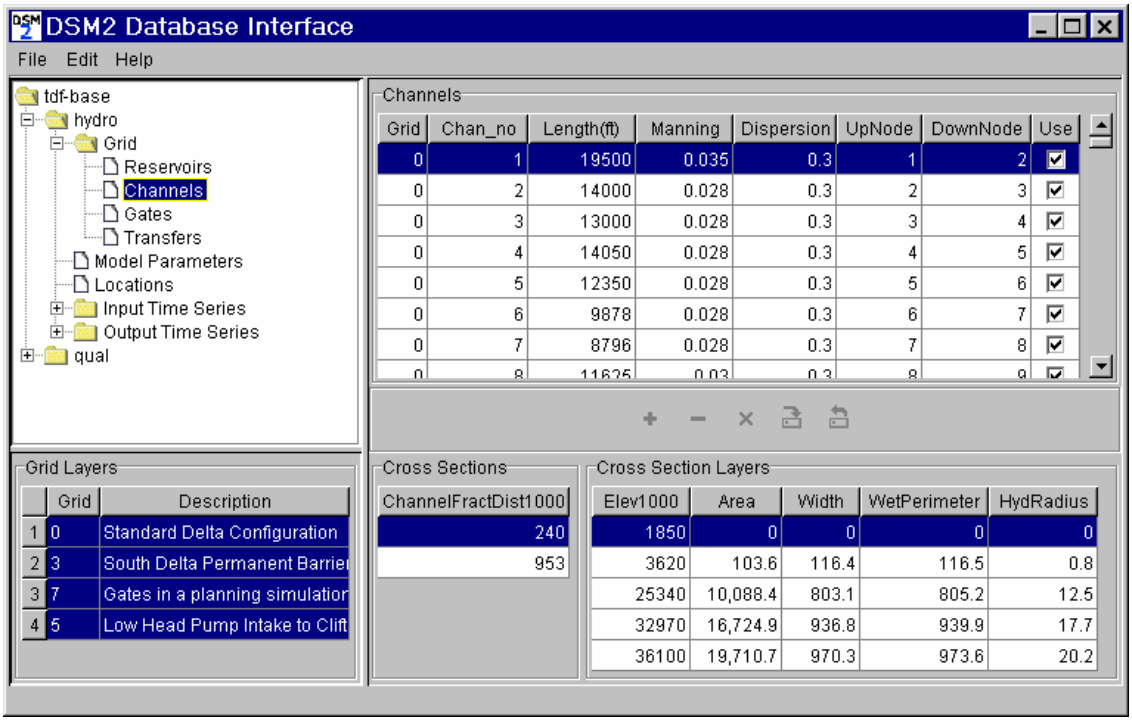

Figure 13.4: DSM2 Database Interface (Channels).

Software clients such as the new DSM2 User Interface and the DSM2 numerical model retrieve information from the database using queries written in the Structured Query Language (SQL) and implemented using software libraries designed for SQL. These libraries in turn rely on

database connectivity protocols (ODBC), which allow disparate data sources and clients to talk to one another. DSM2 connects to the input database using the CainamaSoft f90SQL library and the ODBC protocol. The user interface uses Java libraries by Borland and the JDBC:ODBC protocol. Further discussion of data normalization, SQL and ODBC is beyond the scope of this chapter; these subjects are described in most database software manuals and books.

13.4 Component Sets and Layers

There is one aspect of modeling practice that was particularly important to capture in the database design, and that is the frequent reuse of groups of components. A component is an object used by DSM2 such as a channel, a parameter, or information for inputs and outputs. To understand the importance of component reuse, one only need reflect on the type of work done by DWR (and most other institutional modelers). The bulk of the work for institutional modelers is not to pioneer new domains, but to investigate changes in a small handful of established ones. DSM2 modelers in DWR currently focus on the main Delta and an extension of the San Joaquin River.

If the number of model domains is small, the number of individual management decisions is at most medium sized – a few dozen are under active consideration at any one time. However, these components act like letters in an alphabet or notes in a scale – their permutations generate a large number of scenarios for investigation. For instance, a recent request for a study specified: standard Delta grid, CALSIM boundary input, South Delta Permanent Barriers, Through-Delta Facility, and low-head pump alternate intake to Clifton Court intake. All of the components on this list are potentially useful in other contexts.

In the new database, logically connected component groups like the standard Delta grid, the CALSIM Input set, or South Delta Permanent Barriers can be tagged together in one set. These sets may then be freely combined to construct a specific run. There are four kinds of sets:

- Grid components: Groups of channels, gates, reservoirs, and object-to-object transfers.
- Parameter sets: Groups of model parameters (grid resolution, closure parameters) that are commonly applied to numerical models.
- Input sets: Groups of input time series for boundary data (flow, stage, and water quality boundary conditions).
- Output sets: Groups of output time series commonly requested together.

All that remains is to define carefully the behavior when several component sets of a particular type (for example two sets of grid components) are used together. In the preceding example, the standard (historical) Delta grid defines temporary barriers in the south Delta, but the proposed study must overwrite these temporary barriers with the above-mentioned South Delta Permanent Barriers.

The prioritization is accomplished through a “layer” scheme similar to that used in image software such as Adobe PhotoShop™. Figure 13.5 illustrates this principle. The base layer 1 shows a model with 6 channels. The modeler wishes to alter channel 2 and extend the grid upstream of channel 3. This is accomplished by adding a new layer with only two elements: one for channel 2 and one for channel 7.

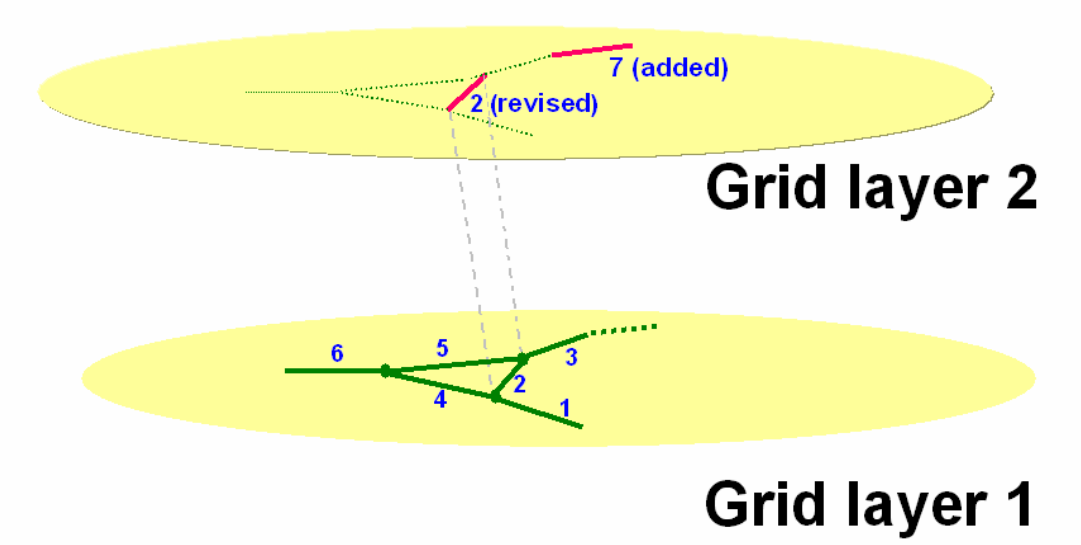

Figure 13.5: Input Layers.

In a sense, this layering scheme merely enforces good modeling practice – after all, the DSM2 text input system has always allowed overwriting. The layering system is only novel in two senses. First, it is enforceable. Users who want to dredge a channel as part of a what-if analysis cannot do so by making arbitrary changes on the standard Delta grid files. Second, overwriting is explicit. Every component carries the identification of the bundle it is defined on (e.g., the GridID field for channels shown earlier in Figure 13.4). The DSM2 User Interface is able to graphically represent:

- The input as DSM2 will see it.
- Components that have been overwritten by higher level layers.

Figure 13.6 shows how in the DSM2 User Interface entries are shaded when they are overridden by a higher-level layer, alerting the user that an entry has been overridden and allowing the user to see what the previous value was.

Some security is enforced at the level of the interface. Changes to the grid (or boundary inputs, parameters, etc.) are only allowed on one layer at a time. The interface does not allow changes until the user has selected an “editing layer”.

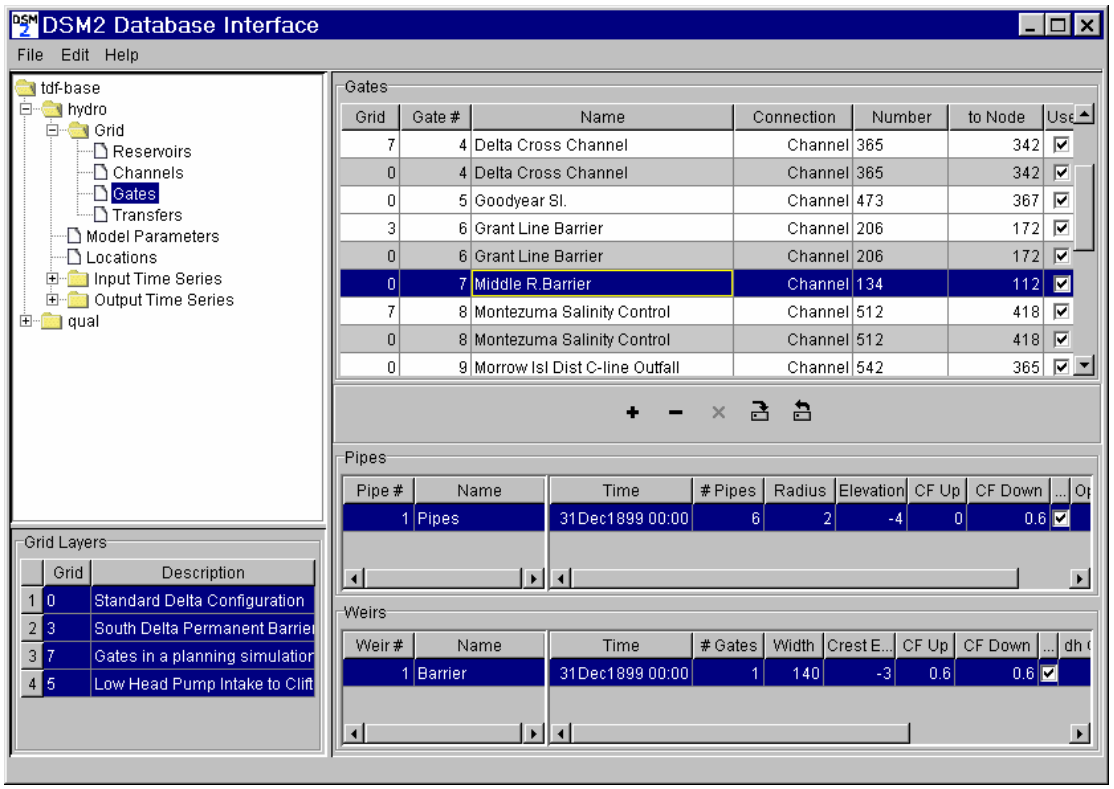

Figure 13.6: DSM2 Database Interface (Gates).

13.5 Version Control

The layering system described in the previous section is designed to facilitate experimentation, not version control. Version control is the management of refinements to the model over time. For instance, new bathymetry data may become available for channel 2, making it clear that the representation of this channel in the "standard Delta group" should be deeper. In version control, a "better" and a "worse" is usually involved. Contrast this example with an experiment: we want to analyze the outcome of dredging Channel 2. In this case, two simulations will be conducted with two different geometries for the channel, each "correct" for the scenario it represents.

Once an update has been finalized, modelers will want to make use of this correction in nearly all subsequent model runs. For the most part, distribution of the new version is effortless due to the enterprise replication scheme. The custodian of the standard Delta grid will make the necessary change and save it to the master database. The new version is then propagated to replicates upon synchronization.

In rare instances, simulations have to be performed again and replicability of a previous simulation is important (note: this discussion covers replication of an experiment, not database replication). Fortunately, the ability to roll back and view a previous version is a capability of transaction-based industrial databases, such as the Informix server. In other words, we can view "the input database as it was on March 15, 2002". As of June 2002, the Informix-based version control has not yet been implemented or tested.

13.6 Data Validation and the Externalization of Quality Control

When text is used for model input (with the exception of text that follows the XML standard), there is no choice but to validate the data inside the numerical model. This checking adds considerably to the complexity of the code, increasing the chances of software bugs. It is also a black box – without scrutiny of the source code, the user can only know from experience which items have been checked.

Externalization of quality control refers to the shifting of quality checks from DSM2 to the database and the user interface. Data validation is a standard feature of database software. The database is able to enforce field-level validation rules, such as: "channel length must be positive" or "Mannings coefficient must be between 0.0 and 1.0". It can also enforce some record-level validation rules involving more than one field, such as: "if two reservoirs have the same name, they cannot have a different reservoir number". The database passes these rules on to the user interface, so that they are enforced during data entry. The code to perform the checks is part of the underlying Microsoft, Informix, and Borland software, freeing DWR engineers to concentrate on aspects of the code in which they have better expertise and sufficient resources.

Besides being more efficient, external data validation is more transparent to the user. When data validation is internal to the numerical model, users make the error first and can only assume the internal checking will be exhaustive enough to catch them at run time. With the database and user interface, validation is pre-emptive and even interactive: graphical *picklists* are used for some input to restrict the range of entries to a prearranged data dictionary. Figure 13.7 shows the use of a picklist while selecting a *Location* for model output. This picklist displays all previously defined model locations along with descriptions, at the same time preventing the user from entering a *Location* that has not been defined yet (in another part of the interface).

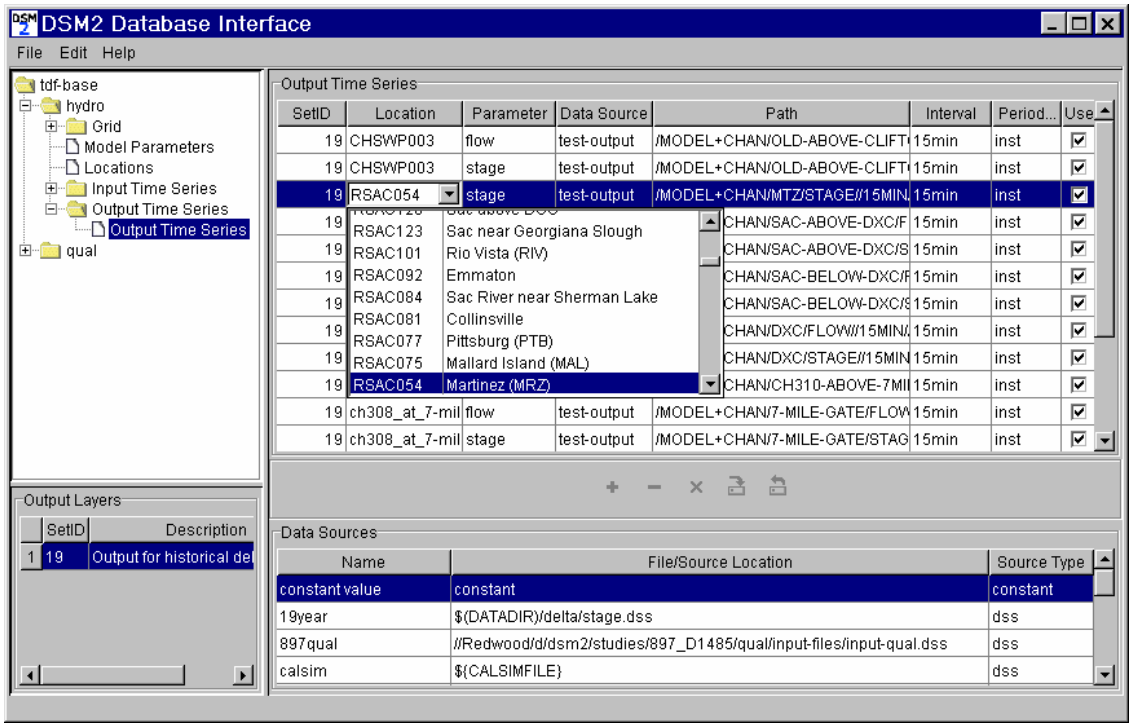

Figure 13.7: DSM2 Database Interface (Output Time Series).

13.7 Conclusions

The DSM2 Input Data project represents an effort not only to bring modern data management to DWR's Delta Modeling Section, but also to conform to the way Delta modelers do their work. The layering scheme described in this chapter is an innovation aimed at the typical institutional setting: few model domains, lots of management options. The other facets of the data management strategy – data validation, data normalization, version control and replication – simply make use of established database software features.

The database project described in this chapter is in a beta-testing state. A few DSM2 model runs have been performed using database input and checked successfully against text input equivalents. The user interface is complete, but still awaits documentation. The connection to the Informix server, which is needed for both external connectivity and version control, is still incomplete.

Finally, the database project heralds one more change in software development strategy. For the first time, a proprietary library (f90SQL by CainamaSoft) has been linked directly to DSM2. Moreover, since this library is only available for the PC, the UNIX environment must be abandoned – at least for the time being. These changes, while lamentable according to open-source philosophy, represent an excellent shift in resource allocation for DWR. The computational portion of the code is still completely open, while some of the data management chores have been shifted to specialty software that is well supported.