<寄稿>

Using EXTRAN for Real Time Control of a Large Urban CSO Network

E. Strand[1], D. Mears[4], Z. Vitasovic[1*], E. Burgess[2], B. Marengo[3], E. Speer[4]

[1]Camp Dresser & McKee, 11811 N.E. First Street, Suite 201, Bellevue, WA 98005
[2]Camp Dresser & McKee, 8805 Governor's Hill Drive, Suite 260, Cincinnati, OH 45249
[3]Philadelphia Water Department, Office of Watersheds, 1101 Market St., Philadelphia, PA 19107
[4]Reid Crowther Consulting, 155 NE 100th, Suite 301, Seattle WA 98125

Abstract

The Philadelphia Water Department is investigating the application of real time control (RTC) to maximize the utilization of its existing combined sewer network facilities in the Southwest Drainage District. In order to leverage the significant investment in existing SWMM-based modeling, a version of the EXTRAN routing model has been compiled as a Microsoft Windows® Dynamic Link Library (DLL) and included as part of the SewerCAT modeling environment.

Key Words : EXTRAN、RTC、CSO、SewerCAT、SWMM、DLL

Introduction

The problems of sewage overflows and local street flooding have traditionally been addressed by large scale capital improvement programs that focus on construction alternatives such as sewer separation or construction of storage facilities. The cost of such projects is often high, especially in older communities where the population density and the value of land is high. In the last few years, real time control (RTC) of conveyance systems has been emerging as an attractive alternative. In tandem, dynamic models used to simulate sewer systems have begun to incorporate real-time control algorithms associated with gate structures or pumps to provide engineers with the necessary tools to design and evaluate the effectiveness of RTC.

The most frequently used model for simulation of sewer networks in the US is Storm Water Management Model, or SWMM, a public-domain model endorsed by EPA. The traditional structure of SWMM model is characterized as a batch process. No interaction from the user can be intercepted by the modeling software, and therefore the determination of gate positions and pumping rates must be made before the model run is initiated.

SewerCAT is a public domain model developed by Reid Crowther. The SewerCAT model framework was developed to allow a user to interact with a model as it executes through simulation time. This is accomplished by utilizing hydraulic solution "engines" that solve for the equations of flow one step at time. If the engine is sufficiently encapsulated, the modeling framework sends a message containing the current state of the network (flows, heads, gate positions, pumping rates, etc.) and the current time-step's boundary conditions (inflows, tide levels, etc.) and asks the solution engine to solve for a single time-step. The results of the flow calculations are then returned to the framework (user interface). With these results, the interface can update graphic profiles and time series, calculate desired control settings based on the current conditions (including user interaction events) before returning this information back to the engine to

*Dr. Zdenko Vitasovic は, 現在、下水道網および最終沈殿施設のモデル化・自動制御などのスペシャリストとして、米国の Camp Dresser & McKee 社で活躍されています。また、今年スウェーデンで開催される IAW/ICA Conference (ICA2001) の国際プログラム委員も勤められています。

repeat the process until the entire simulation interval has been completed.

Because this object-oriented architecture encapsulates the hydraulic model in this manner, the user is not required to adhere to a single model. Furthermore, other aspects of the simulation environment can be manipulated and managed as objects. For example, a project where multiple versions of a sewer network are modeled to represent multiple capital strategic plans. Event simulations often require management of multiple sets of inflow data files, and model output often requires an intense level of data management for large projects. Figure 1 illustrates how these aspects of simulation can be made flexible in this environment.

The various components (models and data) must be able to communicate with the framework. However, this interface definition (as shown pictorially by the various interface geometries) does not preclude multiple objects from "fitting" into the framework.

Prior to the Philadelphia Water Department (PWD) participation, SewerCAT had been used successfully in this manner with two implicit models: RUNSTDY and Superlink. PWD began investigating the feasibility of real-time controls in 1999. This study would rely heavily on hydraulic modeling to quantify the results of various real-time control measures. Because PWD had previously completed extensive modeling of the Southwest Drainage District with EXTRAN, a significant advantage could be gained if this investment could be leveraged as part of the real-time control project. The "best of both worlds" solution for this project was to encapsulate EXTRAN into a Dynamic Link Libarary, compatible with the SewerCAT framework.

Methodology

In order to make it possible for SewerCAT users to do hydraulic modeling using the EXTRAN model, a SewerCAT-compatible software component that reproduced the computational characteristics of EXTRAN was required. The design for this component had to take into consideration several somewhat conflicting goals. A primary goal was the preservation of the computational integrity and stability of the EXTRAN model, otherwise no legitimate claim could be made that EXTRAN results produced through SewerCAT had the same level of veracity as those produced by running the EXTRAN model as a batch program. Another goal was to preserve as many of the unique features of EXTRAN available to the SewerCAT user as possible. Thirdly, this had to be accomplished without changing the SewerCAT user-interface in any significant way that would produce an additional 'learning-curve' for users striving to get up to speed on using EXTRAN from within SewerCAT. Finally, to enhance the long-term viability of EXTRAN in this form, the EXTRAN code needed to be modular and adhere to modern software engineering standards. This makes it easier to debug, maintain and extend in the future.

In a sense, the EXTRAN model can be thought of as a 'black-box' software component. It has a set of well-defined inputs: network data, inflows, and downstream boundaries, and a set of well-defined outputs: the heads and flows in the network at each time step of the simulation. It comprises a conceptual module that can be interchanged with other modules of the same type. In its SWMM implementation however, EXTRAN is not a 'black-box' but is instead tightly bound to the SWMM interface and very dependent on the internal workings of the SWMM program. Because of this tight binding it would be very hard to take the SWMM implementation of EXTRAN and use it separately. The first task in creating the EXTRAN Dynamic Link Library was to loosen this binding between SWMM and EXTRAN, in other words, to separate the code that was really part of the EXTRAN hydraulic engine from the code that was performing input/output tasks such as reading the SWMM interface files, reading the SWMM input file, and writing out the ASCII text results file. First, the code that does reading of data from the input file was separated from the error-checking and initialization code (the code that massages the user's input data into a form more amenable to the internal workings of EXTRAN). Secondly, both of these sets of code were broken down into logical subunits. For example, the reading of data from each 'card' in the input file would now be handled by a separate subroutine. This new, modular version of EXTRAN could still be run from within SWMM, but its internal structure was now very different. To verify the results of this task, the new version of EXTRAN was run against a number of test cases, and made sure the results matched those produced by the original.

The next step was to use the new, 'modularized' version of EXTRAN to produce a Windows® Dynamic Link Library (DLL) that adhered to specific

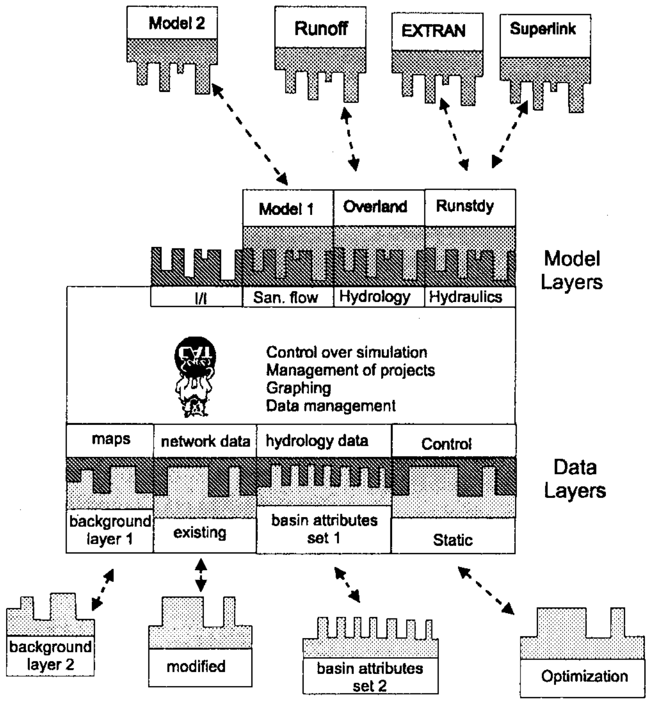

Fig.1 SewerCAT Conceptual Framework

calling conventions. Simply put, EXTRAN would now be a separate software component that could be interfaced to any Windows application with a minimum of coding effort. This hypothetical application would gather input and mediate all interaction with the user, and let EXTRAN do what it does best: solve for the heads and flows in the network. Of course, the intention from the beginning was for SewerCAT to fill this role, and the interface of the EXTRAN DLL was designed to match the interfaces of the other hydraulic engines that were currently being used with SewerCAT. In this standard interface, each type of network object is encapsulated in a data structure that can be sent to the DLL as a single parameter. Functions were added to accept these structures (or arrays of them) and move the data into EXTRAN's internal arrays and common blocks. To simplify matters, the hydraulic model DLLs use only metric units and all conversions between unit systems are handled by the calling program (in this case, SewerCAT).

Early in the process of testing the DLL, a problem arose regarding the initialization of data structures. Since SWMM EXTRAN was designed to run only once per loading of the program, it was designed with the assumption that there would be no previous non-zero values stored in the memory it used for its arrays. When a simulation was run multiple times without restarting SewerCAT, a residual effect was noticed, caused by the values left over from the previous run. In other words, because some non-zero values lingered in the memory space that was assigned to the DLL, the results of the second run of a simulation would not exactly match those of the first. The solution was simple: write a routine that would set to zero (or some other base value) literally every variable in the program. This 'reset' routine is called before the beginning of every simulation, so that EXTRAN starts every time with its memory in exactly the same state.

With the DLL now producing predictable results, the next task was to add some features to the SewerCAT graphical user interface to make the features of EXTRAN available to the SewerCAT user. To this end, a new 'Extran Parameters' object was added to SewerCAT. It gives the SewerCAT user access to some of the B Card parameters for controlling EXTRAN. Other parameters, such as METRIC (which indicates which unit system is being used), and EXTRAN's time and date parameters, are set automatically by SewerCAT.

Additional work was required to reconcile the capabilities of SewerCAT and EXTRAN. For example, tidal boundaries using coefficents of sine and cosine functions were available in EXTRAN. This capability was added to the SewerCAT interface, which now makes boundary conditions with tidal coefficents available to all SewerCAT hydraulic engines. Likewise, while EXTRAN allowed the user to create storage junctions with a power function cross section, there was no similar capability in SewerCAT, so facilities had to be added to the SewerCAT interface that allowed the user to create this type of storage junction.

One of the functions of EXTRAN's venerable ASCII output file, besides printing results, is to report any errors originating from the input data. Because results are handled very differently in this architecture, the creation of this file was removed from the DLL implementation. However, the error reporting function was still required. SewerCAT had to acquire the capability of detecting any problems with the network data and alerting the user to them before starting the simulation. Removing all reliance on the ASCII file in turn allowed the removal of all dependence on scratch files.

Under certain circumstances, the EXTRAN model is vulnerable to instability, overflow and underflow, and other numerical problems that cannot be foreseen. Because the EXTRAN DLL faithfully reproduced the behavior of the SWMM version of EXTRAN, these behaviors have not gone away. To prevent users from crashing SewerCAT, and potentially corrupting their data by running an unstable simulation, a basic floating-point exception trapping ability was added to the DLL. This means the DLL will stop if it performs an illegal floating-point operation, such as dividing a quantity by zero. The user can then adjust the parameters or try in some other way to resolve the instability before attempting to run the simulation again.

Results

Results from batch versions of EXTRAN and the EXTRAN DLL were compared to ensure that the hydraulic calculations produced the same results. The network for Philadelphia's Southwest Drainage District provided an excellent proving ground for this task. This model, developed by PWD and Camp Dresser and McKee, contains 2094 conduits, 2077 nodes, 71 storage junctions, 56 orifices, 3 pump stations, and 173 flap gates. This covers approximately 50 square miles of densely populated areas within the city limits of Philadelphia.

In order to verify that the results produced with the EXTRAN DLL are comparable with the output from the Batch version of EXTRAN, a storm event (March 8-9, 1998) was simulated in both environments. The verification process is relatively straight-forward: with the same EXTRAN network and the same inflow hydrographs, the resulting flow rates or levels can be compared from the output. The results of flow from three different locations are shown in Figure 2. The results are identical, except during brief periods when numerical instabilities occur during the simulation (e.g. conduit IC16-000 near the end of the simulation period). In this case the instabilities are relatively brief and have been traced to solution switching between supercritical and subcritical flow regimes, which can occur in this erratic manner when flow conditions are maintained near critical (Froude numbers near 1.0). This is a fairly common and minor anomaly in EXTRAN simulations.

1 Conclusions

The result of this effort has achieved many of the goals that were set at the commencement of this project. The software component works seamlessly within the SewerCAT graphical interface, and adheres to the same interface as SewerCAT's other hydraulic modeling engines. The resulting EXTRAN DLL component accurately reproduces the behavior of the original EXTRAN model, and has the following features:

- Simulates at nearly the same speed as the SWMM 4.4 EXTRAN. (Note that the SewerCAT user has the option of creating any number of continuously updating windows, such as hydraulic profiles and time-series graphs, which will slow the rate of simulation).
- The EXTRAN DLL is interchangeable with the other DLL engines in SewerCAT.
- The DLL can simulate the following network features (new to EXTRAN):
    a. Sluice Gates controlled through a variety of algorithms
    b. Several new conduit types, including trapezoid pipe, cunette pipe, Runstdy-

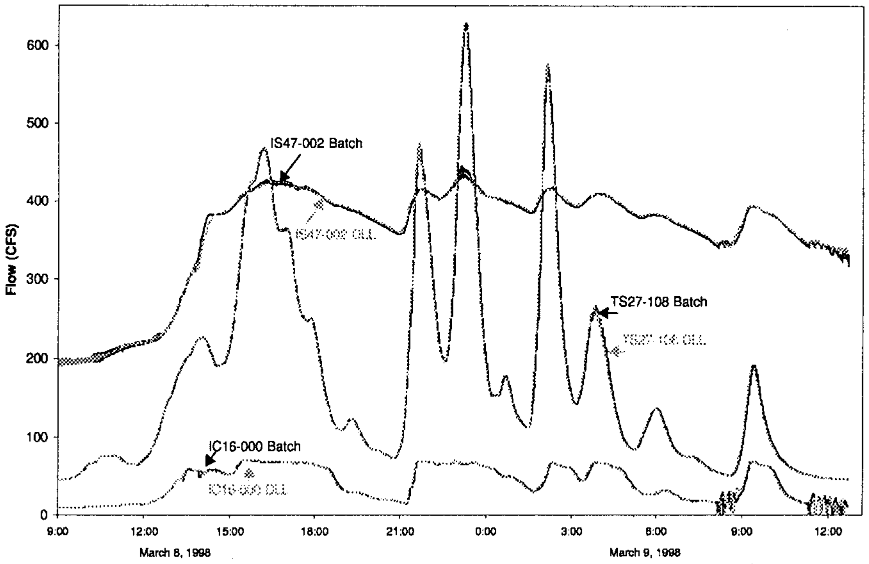

Fig.2 Comparison of Output: EXTRAN Batch vs. EXTRAN DLL

style user defined shapes, and rectangular channels.

c. Pump stations with multiple pumps, each with its own pump curve and on/off settings

- The SewerCAT interface controls the EXTRAN computations in such a way that users are now enabled to:

  a. Use all the various network control structure control strategies implemented in SewerCAT with EXTRAN (Direct Control, Node-Depth Control, etc.)

  b. Combine inflow data from any number of sources, including other SWMM blocks.

  c. Make full use of SewerCAT's dynamic simulation playback and results analysis capabilities.

  d. Easily compare results from simulations run with different sets of control strategies.

  e. Easily compare the results produced by EXTRAN to those produced by other engines running on the same network.

  f. EXTRAN users will be able to take advantage of many more features as they are added to SewerCAT.

These added capabilities are a direct result of SewerCAT's component based nature, and the way in which it separates user interface and control functions from the solving of the hydraulic flow equations.

The re-working of the EXTRAN architecture, without impacting its proven computational hydraulic capabilities, will benefit the user community. In addition to the full real-time control capabilities that are c urrently available through the SewerCAT framework, future EXTRAN enhancements can be made to a more modern and maintainable code base.