19th International Congress on Modelling and Simulation, Perth, Australia, 12–16 December 2011
http://mssanz.org.au/modsim2011

Linking hydrological simulation models with workflow and optimisation software

D.J. Penton [a b], **B.P. Leighton** [a b], **M.P. Stenson** [a], **J.M. Rahman** [a b], **and M. Bethune** [b]

[a] *CSIRO Water for a Healthy Country National Research Flagship, CSIRO Land and Water, GPO Box 1666, Canberra, ACT 2601, Australia*
[b] *eWater Cooperative Research Centre, University of Canberra, ACT 2601, Australia*
Email: dave.penton@csiro.au

Abstract: The literature is rich in discussion of methods, platforms and standards that hope to ease the process of integrating models in workflows or decision support systems. The reality for practitioners is that integrating models is still complex and time consuming in most cases. In addition, the literature contains few descriptions of the practicalities of integrating models in the hydrology domain. This paper presents two case studies to explore opportunities for reducing the complexity associated with model integration. Even though the case studies were different, significant commonalities exist in the linkage requirements for both case studies. The paper finds that meeting these requirements required significant change to the existing model architecture.

The first case study involves linking the hydrological modeling system, Source IMS, with the Delft Flood Early Warning System (FEWS). The FEWS system is being trialed in the Australian Water Resources Assessment system (AWRA) that the Water Information Research and Development Alliance (WIRADA) is developing to support water assessment and water accounts products for the Bureau of Meteorology. The second case study involves linking the Source IMS with Parameter ESTimation (PEST) software for automated calibration of model parameters. These two case studies identified four key requirements that are considered common to how workflow tools would need to integrate with Source IMS. Specifically,

1. Run and retrieve results without a user's interaction.
2. Model independence – both FEWS and PEST remain independent of the model.
3. Manipulate parameters and boundary conditions; there was no need to modify the structure or configuration of the model.
4. Manage project load times and overall simulation times.

The architecture of Source IMS changed significantly to support mechanisms for simulation stepping, distributed execution, stay-alive functionality and parameter passing. Adopting The Open Modelling Interface (OpenMI) would have addressed these requirements. However, the start up costs of implementing the OpenMI made this solution unachievable in the short term with the available resourcing. This paper discusses a pragmatic solution that addressed the key integration requirements while reducing the cost of moving to OpenMI in the future. The pragmatic solution involved changes such as:

- Refactoring persistence functions (load and save), data model parameterisation and simulation methods so they were brokered through the application layer interfaces;
- Decorating the application layer interfaces with Windows Communication Foundation (WCF) to enable inter-program, cross-process and cross-computer communication; and
- Developing a command-line client and server application for providing application layer services to external programs.

Practitioners and software developers solving model integration problems desire to leverage standards such as OpenMI. This paper contributes to our understanding of the precursor steps required to implement some of these standards. The precursor to integrating hydrological models in Source IMS was a reengineering of the software architecture. The reengineered architecture supported requirements of both case studies due to commonalities of PEST and FEWS. This paper encourages practitioners to consider major architectural decisions that might simplify or complicate the integration with other products.

Keywords: *Workflows, Decision Support Systems, River Modeling, Water Quantity Modeling, Optimization*

1. INTRODUCTION

The literature is rich in discussion of methods, platforms and standards that hope to ease the process of integrating models in workflows or decision support systems (e.g. Gregersen (2004)). Good design and significant effort is required to realise the benefits of such integration systems. Time and budget constraints typically limit the extent to which domain models implement integration support systems and methods. Consequently, the reality for most practitioners is that integrating models is still complex and time consuming.

The eWater Cooperative Research Centre is building a new integrated modelling platform, called the Source Integrated Modelling System or Source IMS. Source IMS represents the physical and managements aspects of water movement and storage within a catchment, including runoff generation in catchments, storage in reservoirs and demand modelling. Source IMS includes functionality to model the complex management rules associated with regulation in Australian rivers (Penton and Gilmore, 2009). This functionality will enable Source IMS to underpin water sharing plans that are currently modelled in Australia using Integrated Quantity Quality Model (Podger et al, 2004), Resource Allocation Model (Perera et al, 2005) and Monthly Simulation Model-Bigmod (Close, 1996).

Many real world applications require embedding the hydrological model within quality assured workflows, to support decision-making and for tasks such as automated calibration. Practitioners are using Source IMS in two current applications:

1. Water assessment and accounting. Under the Water Act 2007, the Bureau has a legislative requirement to deliver national scale water assessments and creating water accounts. The modelling system that underpins the assessment and accounts is the Australian Water Resources Assessment system (AWRA) (Van Dijk et al. 2011).
2. Calibration of rainfall runoff models to support landscape planning decisions.

While some tools exist in TIME to support these tasks, there are specialised, highly developed and supported tools in the market designed to address these requirements. Support for integration with such external applications is required to ensure Source IMS can be used in workflows to support business requirements.

This paper presents two case studies that demonstrate the complexity of integration. The first case study involves linking Source IMS with the Delft Flood Early Warning System (FEWS) for conducting water assessments and creating water accounts. The second case study involves linking Source IMS with the Parameter ESTimation (PEST) software for calibration of model parameters. Common requirements to both case studies are identified and revisions to the architecture of the Source IMS are discussed.

2. EXISTING SOURCE ARCHITECTURE

Source IMS has evolved from the E2 catchment modelling system (Argent 2005), which was built using The Invisible Modelling Environment (Rahman et al., 2003). E2 primarily represented the hydrology and water quality of unregulated catchments (Argent et al., 2009). The Source IMS architecture evolved to have six major logical components as shown in Figure 2–1:

- User Interface Component, which provides the user interface controls for building, parameterising, loading, saving and running a model. Modellers that use Source IMS conduct most of their work through Source IMS's graphical user interface. This component became the controller of most tasks.
- Persistence Component, which provides methods for saving and loading projects. Source IMS saves projects as a distributable binary file format.
- Data Model Component, which describes the attributes of a project and scenarios (e.g. the number of catchments and how they connect to each other)
- Simulation Engine Component, which provides methods for managing the simulation (e.g. scheduling the execution of various components) and returning results.
- Command Line Component, which provides a non-graphical way for users to build, parameterise, load and run a model.
- System Test Component, which provides methods for an automated build system to test that model output is as expected.

While these components are logically separate, the reality of a decade's development in Source is that strong coupling developed between components. While it would have been possible to work around the existing architecture for PEST through user interface tasks, supporting the FEWS and PEST directly through architectural changes was highly desirable.

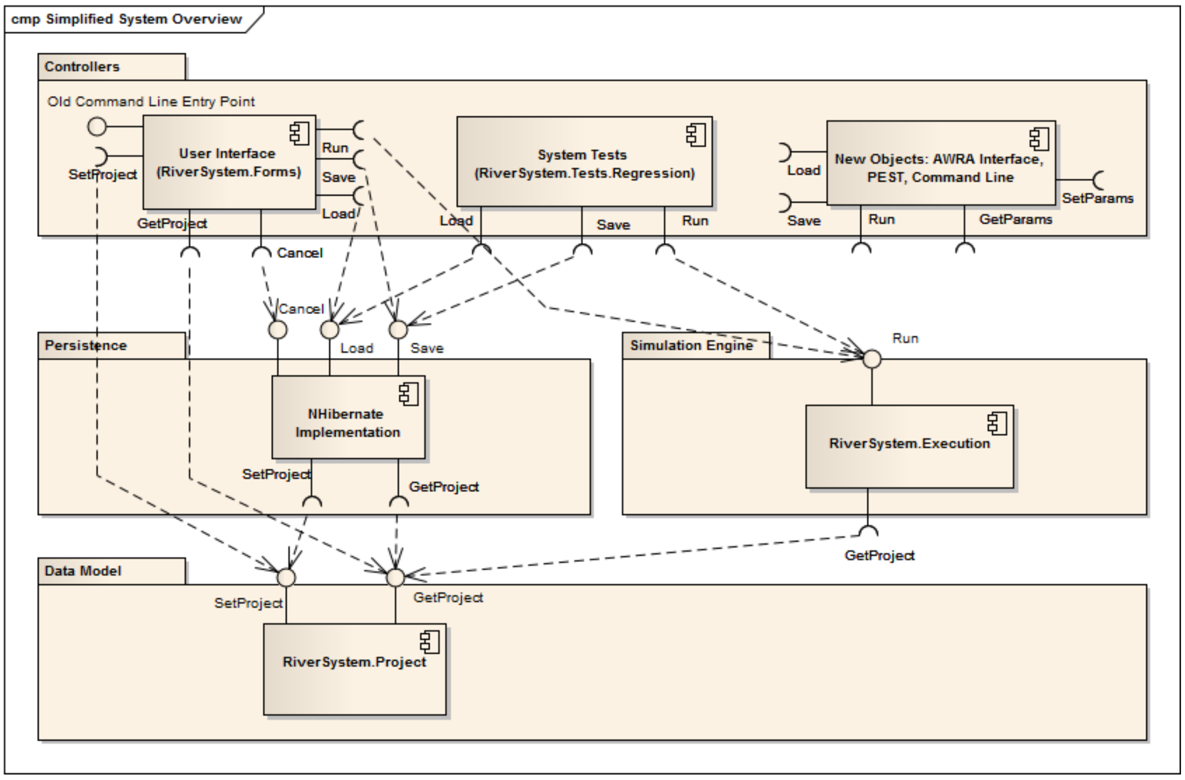

cmp Simplified System Overview
Controllers
Old Command Line Entry Point
User Interface (RiverSystem.Forms)
SetProject
GetProject
Run
Save
Load
Cancel
System Tests (RiverSystem.Tests.Regression)
New Objects: AWRA Interface, PEST, Command Line
SetParams
GetParams
Persistence
NHibernate Implementation
Simulation Engine
RiverSystem.Execution
Data Model
RiverSystem.Project

Figure 2–1. Component diagram showing Source prior to integration with AWRA and PEST.

3. DELFT-FEWS CASE STUDY

The first case study involves linking the hydrological modelling system, Source IMS, with the Delft Flood Early Warning System (FEWS). The FEWS system is being trialed in the Australian Water Resources Assessment system (AWRA) that the Water Information Research and Development Alliance (WIRADA) is developing to support water assessment and water accounts products for the Bureau of Meteorology. The objective of AWRA is to report the quantity of water at the national scale on a regular basis. The Bureau of Meteorology hopes that temporal analysis of water quantities will highlight patterns, trends and variability in water quantity, which will inform public policy and program development.[1] This requires AWRA to measure the terms of the water balance such as reservoir storage and inflows. In places where direct observations are unavailable or unreliable, AWRA must estimate terms of the water balance retrospectively.

The nation-wide scope of the data gathering for AWRA requires that the process for generating water balance terms and reports be automated and easily repeatable. The platform under trial for this process is the Delft-FEWS software. Delft-FEWS provides a software infrastructure for operational water management and forecasting (Gijsbers 2008). It provides native abilities to extract data from remote sources and transform this data ready for further use. Gijsbers (2010) discusses using Delft-FEWS as a scientific workflow, which is how AWRA are using it. Gijsbers also highlights that Delft-FEWS provides workflow functionality for a subset of problems and its user interface does not provide a suitable environment for discovering new workflow items.

WIRADA have developed a FEWS module for landscape modelling called AWRA-L. Van Dijk (2010) reports that the AWRA-L component uses a 0.05° grid-based structure to cover continental Australia with rainfall run-off calculated for each cell. In FEWS, once AWRA-L has run for a time-step, the AWRA-R module needs to model river reach processes to generate terms of the water balance. AWRA-R models reach processes with Source IMS to simulate the routing processes. AWRA-R wraps the Source IMS executable; it sends reach parameters to Source IMS, which returns river flows and other water quantities. Figure 3–1 shows how the AWRA components execute sequentially as part of a workflow within the FEWS engine on a daily time-step.

[1] Retrieved from http://www.bom.gov.au/water/awra/index.shtml on the 5th July 2011.

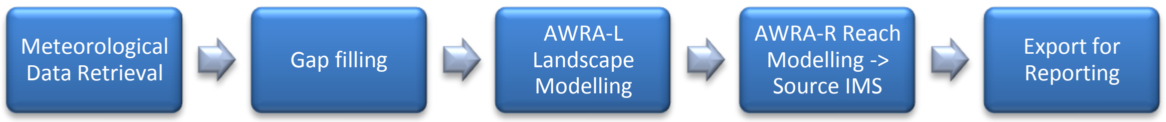

Meteorological Data Retrieval
Gap filling
AWRA-L Landscape Modelling
AWRA-R Reach Modelling -> Source IMS
Export for Reporting

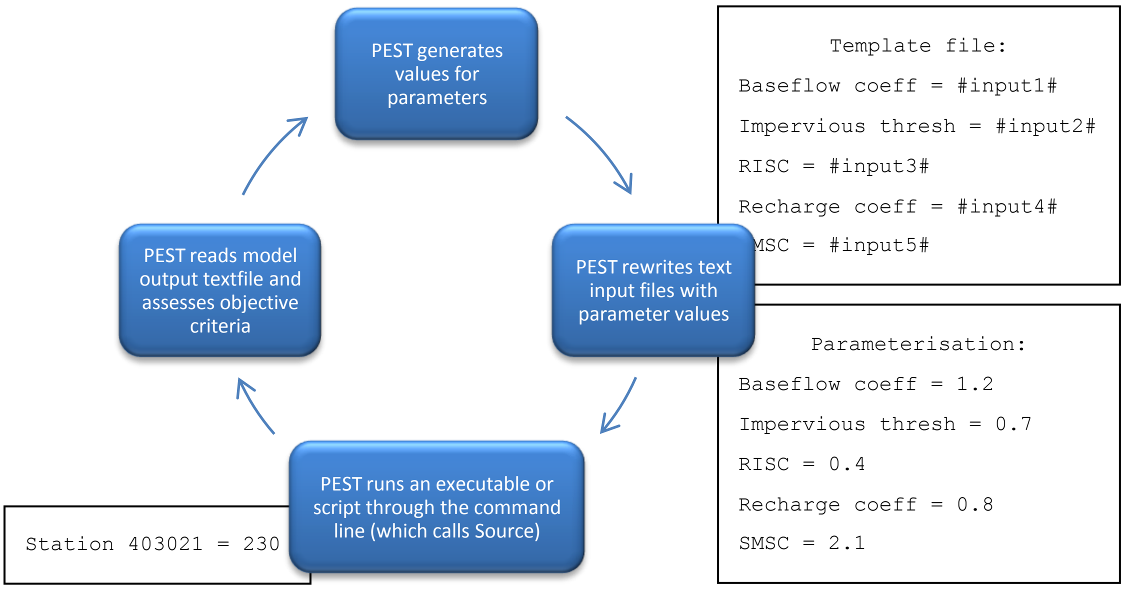
Figure 3–1. AWRA Hydrological Workflow executed by FEWS version 0.1.

The key constraints of the FEWS environment are:

- The executable that FEWS runs must not require user input to run a simulation.
- The simulation engine of the target model must run a time-step at a time
- The AWRA-R module must handle cross-computer communication required by performance constraints.

4. PEST CASE STUDY

The second case study is the linking of Source IMS with PEST for calibration of rainfall runoff models in long-term catchment planning scenarios. PEST software provides a model independent mechanism to identify parameter sets that minimise or maximise an objective. For example, PEST can search for the rainfall run-off parameters that minimise the difference between modelled flow and observed flow at a downstream gauging station. PEST does this by calling a command line application with different values for each parameter then running an objective function against the outputs (See Figure 4–1).


PEST generates values for parameters
PEST rewrites text input files with parameter values
PEST runs an executable or script through the command line (which calls Source)
PEST reads model output textfile and assesses objective criteria

```
Template file:
Baseflow coeff = #input1#
Impervious thresh = #input2#
RISC = #input3#
Recharge coeff = #input4#
MSC = #input5#
```

```
Parameterisation:
Baseflow coeff = 1.2
Impervious thresh = 0.7
RISC = 0.4
Recharge coeff = 0.8
SMSC = 2.1
```

```
Station 403021 = 230
```

Figure 4–1. PEST executes in the sequence shown above for calibration or uncertainty analyses. A template file (top right) is updated with the parameters on each iteration (bottom right). Then the objective function is evaluated after the executable or script is run (bottom left).

Ellis (2009) reports that using PEST to calibrate Source Catchments rainfall runoff models for a landscape planning model provided an increase in average coefficient of efficiency for daily flow at 20 locations from 0.50 to 0.79 compared to calibration external to the Source Catchments model. This improvement occurs through the ability to calibrate multiple catchments and routing reaches together. Ellis (2009) also reports that despite simplifications and subjectivity he is able to produce a transparent and quantifiable measure of uncertainty. PEST calls its target (in this case Source), through the command line.

The key constraints of PEST are that:

- There must be at least one input ASCII text-file that PEST can modify.
- The executable must not require user input to run a simulation.
- There must be at least one output ASCII text-file that PEST can use to calculate the objective value.

Another factor to take into consideration is that the target executable of a PEST calibration is loaded afresh for every parameter set.

5. REQUIREMENTS COMMON TO THE CASE STUDIES

First, FEWS and PEST both need to run and retrieve results without a user's interaction. Second, FEWS and PEST both deal with a model, but they remain independent of the model. Third, FEWS and PEST manipulate parameters, but they do not drastically modify the structure or configuration of the model.

These two case studies demonstrated overlapping requirements from workflow and optimisation software on the Source IMS, or more generally, any consumed model. Four requirements were common to the two use cases:

1. Run and retrieve results without a user's interaction.
2. Model independence – both FEWS and PEST remain independent of the model.
3. Ability to manipulate parameters and boundary conditions, there was no need to modify the structure or configuration of the model.
4. Improvements to project load times and overall simulation times.

6. ARCHITECTURAL CONSIDERATIONS

The Source IMS had to support mechanisms for simulation stepping, distributed execution, stay-alive functionality and a generic mechanism for parameter passing. FEWS required Source IMS to progress the simulation one time-step at a time. The underlying Source engine already supported step by step execution, though some of the logic required refactoring to expose this to external tools such as FEWS. Exposing the step by step execution to external tools provides the ability to adopt the Open Modelling Interface (OpenMI) subsequently, as discussed in Section 8.

Integrating multiple software packages in a distributed execution requires support at the architectural level. Goodall et al (2011) introduces a standards-based service-oriented architecture (SOA) for hydrologic modelling with OpenMI. A SOA is a method for connecting services – a service being a long-lived process that conforms to a contracted interface. Schmutz et al (2010) provides a taxonomy of integration architectures and communications protocols for SOAs. They identify five middle-ware communication methods: conversational, request/reply, message passing, message queuing, publish/subscribe and four integration architectures: point-to-point, hub-and-spoke, pipeline and service-oriented architecture. A message passing method was adopted in order to provide client components with feedback on the progress of long running simulation tasks. The point-to-point architecture, while having drawbacks, provides the AWRA-R component with the lowest start-up costs. Schmutz et al (2010) provides a more detailed break-down of the trade-offs.

Calibrating Source IMS through PEST is faster if the target model is loaded once and the same instance is used for all simulations. Although Ellis (2009) use background processes that communicate via touching shared files, the typical implementation for this is to use services (in this case windows services).

There were alternate ways to support the retrieval and modification of model parameters:

- Historically, many models have their entire structure described in text files. This simplifies some tasks, such as the connection to PEST, but it has implications for maintenance as the models evolve.
- Defining a simple 'language' (really a format for strings) that can identify any numeric model parameter based on where it appears in the model hierarchy – eg 'Subcatchment 24' | 'Forested Areas' | 'Rainfall Runoff' | 'Soil Moisture Store Capacity'. This is somewhat more robust than capturing the entire model structure in text, but is still subject to change as the model evolves.
- Providing a mechanism for the user to 'tag' certain model parameters with simple names, with the model responsible for maintaining the link between the user's tags and the desired parameters as model versions evolve.

Each approach has advantages and disadvantages, but for a balance of flexibility and robustness, the approach of tagging variables was chosen.

7. REVISED ARCHITECTURE

The core Source simulation engine provides a mechanism for modelling a variety of long-term planning scenarios in regulated and unregulated rivers and catchments. However, this functionality was not easily accessible to external applications. The following changes were made:

- Persistence functions (load and save), data model parameterisation and simulation methods were refactored so they were brokered through the application layer interfaces;

- The application layer interfaces were decorated with Windows Communication Foundation (WCF) tags to enable inter-program and cross-computer communication (WCF is a common platform for Microsoft .NET application development, alternatives would probably perform as well); and
- A command-line client and server application was developed that provided methods to external programs. The client included options for text-file parameterisation of the model.

Figure 7–1 shows the application layer as the broker for calls to the persistence, data model and simulation components. The user interface uses the application layer for persistence and simulation functions, but user controls outside the application layer directly modify the data model. The application layer only has the ability to deal with existing models, not to build models from scratch. Figure 7–2 shows how this looks from the point of view of external products. The resulting architecture supports both case studies: FEWS can run Source scenarios; and PEST can calibrate time-series demands.

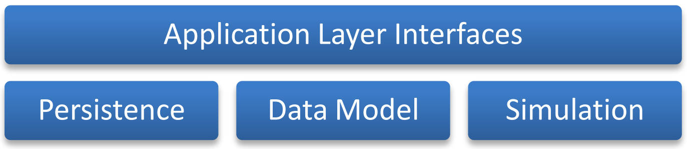


Figure 7–1. The revised architecture introduces an application layer to broker calls to the persistence, data model and simulation components.

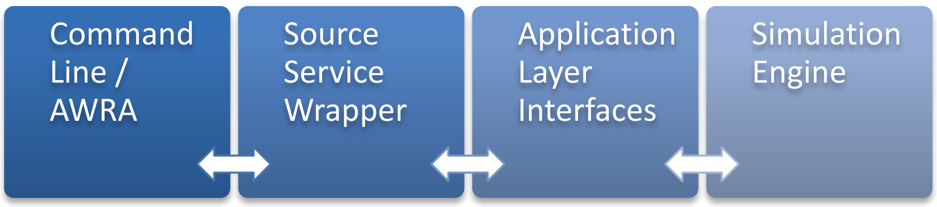


Figure 7–2. The command line and AWRA components interact with the Source service through calls to the application layer. This published interface could be by consumed any other controlling service.

8. OPPORTUNITES TO REDUCE THE MODEL INTEGRATION PROBLEM

Technologies such as WCF and equivalents continue to simplify the generic problem of integrating models across heterogeneous software and infrastructure environments (e.g. inter-program, cross-process and cross-computer communication). The broader modelling and software engineering issues such as resolving different software architectures and working through different interfaces were more complicated than inter-process communication. This type of model interoperability problem is not new. It has motivated generalised solutions including service-oriented architectures, workflows and standards such as OpenMI.

OpenMI defines a software interface that links models in a similar time-stepping framework. According to Gregerson et al (2007) this interface can link OpenMI compliant hydrological and hydraulic models. Donchyts (2010) also reports that improvements in OpenMI 2.0 make the standard more intuitive across a wider range of scientific modelling domains. Christensen et al (2004) show that OpenMI can in practice link river basin models like MIKE BASIN with 3D Hydrological Model MIKE SHE.

In the case of service-oriented architectures, the startup costs of the software development were prohibitive given the available resourcing. However, the creating the clean façade in Source IMS makes the addition of an OpenMI compliant interface much simpler than it would previously have been.

9. CONCLUSION

The literature is rich in discussion of methods, platforms and standards that hope to ease the process of integrating models in workflows or decision support systems. The reality for practitioners is that integrating models is still complex and time consuming in most cases. The two case studies demonstrated that the decoupling of the simulation engine and simplifying the published interfaces in Source IMS increased the potential uses of the model. Even though the case studies were different, significant commonalities exist in the linkage requirements for both case studies.

Practitioners and software developers solving model integration problems desire to leverage standards such as OpenMI. The precursor steps required to implement OpenMI or SOA involved the reengineering of the software architecture. The reengineered architecture supported requirements of both case studies due to commonalities of PEST and FEWS. As reengineering is expensive, this paper encourages practitioners to consider major architectural decisions that might simplify or complicate the integration with other products as early as possible.

ACKNOWLEDGMENTS

We are very grateful for the support of the eWater CRC and the Water Information Research And Development Alliance, a collaboration between the Bureau of Meteorology and CSIRO's Water for a Healthy Country flagship.

REFERENCES

Argent, R.M., Grayson, R.B., Podger, G.M., Rahman, J.M., Seaton, S., and Perraud, J.-M. (2005). E2 - A Flexible Framework for Catchment Modelling. In: Zerger, A. and Argent, R. M., (Eds.), *MODSIM 05 International Congress on Modelling and Simulation*: Melbourne, Modelling and Simulation Society of Australia, p. 594-600.

Argent, R.M., Perraud, J.M., Rahman, J.M., Grayson, R.B., and Podger, G.M. (2009). A new approach to water quality modelling and environmental decision support systems, *Environmental Modelling and Software*, 24(7):809-818.

Christensen, F.D , Coupling Between the River Basin Management model (MIKE BASIN) and the 3D Hydrological Model (MIKE SHE) with use of the OpenMI System, 6th International Conference on Hydro informatics (2004).

Close, A. F. (1996). A new daily model of flow and solute transport in the River Murray. 23rd Hydrology and Water Resources Symposium, Hobart, Australia, 21-24 May 1996, 1066 p. 173-178.

Doherty, J. (2010). *PEST: Model-Independent Parameter Estimation*. Watermark Numerical Computing, Brisbane, Australia. http://www.pesthomepage.org/getfiles.php?file=models_and_decisions.doc. Accessed 5th July 2011.

Donchyts, G., Hummel S., Vaneçek S., Groos J., Harper A., Knapen R., Gregersen J. Schade P., Antonelli A, and Gijsbers P. (2010) OpenMI 2.0 What's New. This conference, iEMSs 2010 Fifth Biennial Meeting, Ottawa, Canada

Ellis, R.J., Doherty, J., Searle, R.D., and Moodie, K. (2009). Applying PEST (Parameter ESTimation) to improve parameter estimation and uncertainty analysis in WaterCAST models: In Anderssen, R.S., R.D. Braddock and L.T.H. Newham (eds) 18th World IMACS Congress and MODSIM09 International Congress on Modelling and Simulation. Modelling and Simulation Society of Australia and New Zealand and International Association for Mathematics and Computers in Simulation, July 2009, pp. 3158-3164.

Gijsbers, P. (2010). Opportunities and limitations of DelftFEWS as a scientific workflow tool for environmental modelling: In Proceedings of the iEMSs Fifth Biennial Meeting: International Congress on Environmental Modelling and Software. International Environmental Modelling and Software Society, Ottawa, Canada, July 2010.

Gijsbers, P.J.A., Werner, M.G.F., and Schellekens, J. (2008). Delft-FEWS: A proven infrastructure to bring data, sensors and models together: In Proceedings of the iEMSs Fourth Biennial Meeting: International Congress on Environmental Modelling and Software, Barcelona, Catalonia, July 2008, pp. 28-36.

Goodall, J. L., Robinson, B. F., and Castronova A. M. (2011). Modeling water resource systems using a service-oriented computing paradigm, *Environmental Modelling and Software* 26 (5), pp. 573–582.

Gregersen, J.B., Gijsbers, P.J.A., and Westen, S.J.P. (2007). OpenMI: open modelling interface, *Journal of Hydroinformatics* 9, pp. 175–191.

Perera, B.J.C., James, B., Kularathna, M.D.U. (2005), Computer software tool REALM for sustainable water allocation and management, *Journal of Environmental Management*, 77(4):291–300.

Podger, G.D., and Beecham R. (2004), IQQM Reference manual, Department of Infrastructure, Planning and Natural Resources.

Rahman, J.M., Seaton, S.P., Perraud, J.-M., Hotham, H., Verrelli, D.I., and Coleman, J.R. (2003). It's TIME for a new environmental modelling framework. *Proceedings of MODSIM 2003*,4: 1727–1732.

Schmutz, G., Liebhart, D., and Welkenbach, P. (2010). *Service Oriented Architecture: An Integration Blueprint*. Packt Publishing, Birmingham, U.K.

Van Dijk, A. I. J. M., Bacon, D., Barratt, D., Crosbie, R., Daamen, C., Fitch, P., Frost, A., Guerschman, J. P., Henderson, B., King E. A., McVicar, T., Renzullo, L. J., Stenson, M. P. and Viney, N. (2011) Design and development of the Australian Water Resources Assessment system. *WIRADA Science Symposium*. Melbourne.

Van Dijk, A. I. J. M. (2010), AWRA Technical Report 3, Landscape Model (version 0.5) Technical Description, WIRADA/CSIRO Water for a Healthy Country Flagship, available at: http://www.clw.csiro.au/publications/waterforahealthycountry/2010/ wfhc-aus-water-resources-assessment-system.pdf (last access: 5 July 2011) Canberra, 2010c,.