Development of A Hydrologic Community Modeling System Using A Workflow Engine

A Thesis

Submitted to the Faculty

of

Drexel University

by

Bo Lu

in partial fulfillment of the

requirements for the degree

of

Doctor of Philosophy

June 2011

DEDICATIONS

I dedicate this work to my wonderful parents,

my mother Xiuyun Li

And

my father Chaogui Lu

ACKNOWLEDGEMENTS

I would like to express my sincerest gratitude to my advisor, Dr. Michael Piasecki for his insightful guidance, constant encouragement, and generous financial support. I feel so fortunate to have him as my advisor. I would like to thank Dr. Jonathan Goodall, Dr. Franco Montalto, Dr. Mira Olson, and Dr. Ilya Zaslavsky for their constructive comments on my research. I would like to thank all my friends and colleagues for enriching my life in this beautiful country. I also would like to thank my parents and sister, for their love, support and encouragement.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Development of a Hydrologic Community Modeling System Using a Workflow Engine
Bo Lu
Michael Piasecki, Supervisor, Ph.D.

Community modeling is a comparatively new paradigm that emphasizes on developing evolving modeling systems through a collective effort. It has gained growing attention within the hydrologic communities because the demand of developing more holistic-view model systems addressing chemical, physical, and biological processes within the geo volumes of the hydrologic cycle. The development of a community modeling system involves a number of technical issues including how to seamlessly integrate various models/modules especially to mediate their communications and executions, how to improve development efficiency by migrating legacy codes, and how to improve model provenance and repeatability of model runs to name just a few. The major objective of our studies is to develop a hydrologic community modeling system (HCMS) that allows constructing seamlessly integrated, workflow-based hydrologic models with swappable and portable modules for retrieving data from various data sources, pre-processing, modeling, and post-analysis. The HCMS is built on the Microsoft's TRIDENT workflow engine which assists in tackling many of the above technical issues during its development. Four libraries are incorporated into HCMS, i.e. a data retrieval, a data processing, a hydrologic computation and a data analysis library, which support to access data from numerous online data repositories using SOAP/FTP protocols or from local data stores, transform source data into model inputs, perform hydrologic modeling, and analyze model results, respectively. It can potentially be applied to anywhere in the nation due to its access to data sets of nationwide coverage, and can reduce the workload of conducting hydrologic modeling tasks to a great level. Besides its feature of supporting parallel or concurrent executions as well as distributing computations in GRID environment can improve run-time efficiency. This thesis comprises three independent papers, which present the studies on (1) the current efforts that have been or are being made for community modeling, (2) the development of the HCMS using the Microsoft's TRIDENT workflow engine, (3) the assessment on the applicability and performance of the TRIDENT-shelled HCMS by applying it to conduct hydrologic studies on the Schuylkill watershed located in the Southeastern Pennsylvania.

CHAPTER 1: INTRODUCTION

Hydrologic communities around the world have developed a plethora of codes in a multitude of programming languages over the past few decades in order to assess environmental processes and to predict changes in the hydrologic realm. While these codes represent a vast amount of knowledge, expertise and resources spent and also have been extremely useful for the purposes they were designed, increasingly the realization emerges that a better coordination and coupling of both models among each other and also to data sources that drive the models is essential. The emergence of the idea of community modeling systems such as the Community Surface Dynamics Modeling System (Peckham, 2008), the Earth System Modeling Framework(Hill et al., 2004) or the Weather Research and Forecasting model (Michalakes et al., 1998) as well as coupling frameworks such as the Open Modeling Interface and environment (Gregersen et al., 2005) and the Interactive Component Modeling System(Reed et al., 1999) have opened up new ways of thinking about how to link up legacy codes or integrate them into frameworks that allow for a more holistic modeling approach than before. Herein the community modeling emphasizes on addressing specific issues or developing modeling systems within a domain through a collective effort, while coupling frameworks are commonly designed for easing the process of model integration thus can provide technical supports and participatory platforms for community modeling.

Most existing coupling frameworks accomplish model integration by undertaking the mechanism of standardizing model interfaces, modularizing model kernels and placing models into a shared configuring and executing environment. The similar functionalities can also be found in existing workflow systems such as the Kepler (Ludäscher et al.,2006), the Triana (Taylor et al., 2003) and the Microsoft's TRIDENT (Microsoft Research Group, 2009), which utilize a workflow or a node resided in a workflow to represent a model. Workflow systems also incorporate features of recording data or model provenance information and supporting parallel or distributed computations thus allowing repetitive executions and can possibly improve running efficiency. To date a few efforts have been made for developing hydrologic community modeling systems based on coupling frameworks, for example, The Invisible Modeling Environment (Rahman, 2003) has been undertaken by hydrologic community of Australia that has incorporated most of its well-known hydrologic models. However, applying workflow systems to the development of community modeling system is still rare in hydrologic domain, thus leading to our studies of developing a hydrologic community modeling system based on a workflow system.

In this thesis, first, an overview of current efforts that have been made on the development of community modeling systems addressing processes located in the Critical Zone of the earth is provided. Second, the development of a hydrologic community modeling system(HCMS) utilizing the TRIDENT workflow system that permits seamless integration of data flows from source, to preparation, to ingestion, to model execution, to harvesting and analysis of the generated result through the design of workflow sequences is presented. The HCMS incorporates four libraries containing a number of so-called activities (execution blocks within TRIDENT) that dedicate to access source data from numerous online/local data resources, process data to meet requirements of model inputs, perform hydrologic computations, and analyzes model results etc. Third, the assessment of the feasibility and run-time performance of the HCMS is conducted by composing different hydrologic-modeling-oriented workflow sequences and applying them to model the hydrologic responses of the Schuylkill watershed located in the Southeastern Pennsylvania.

CHAPTER 2: A REVIEW OF CURRENT EFFORTS TO ESTABLISH A HYDROLOGIC COMMUNITY MODELING SYSTEM

Abstract

Numerical modeling in the water sciences has been shifting from developing single or specific purpose oriented, tightly intertwined model applications to integrated model systems addressing more complex and interlinked geo-physical, -chemical, and-biological processes across all strata of the critical zone geo-volume. This is a logical response to a number of important issues that reach from preservation of legacy code, to a higher degree of development cost efficiency, to the realization that processes in one strata depend on others (for example a mesoscale atmospheric model must be linked to a watershed model which must be linked to a sub-surface model), to harmonize code usage, and to improve code provenance and repeatability of model runs. Consequently, a number of Community Modeling Systems, CMS, have been either proposed or are being developed with typically individual communities taking the lead to develop a CMS for their constituency. While the development of CMS' is a major step forward in trying to harmonize modeling efforts and to increase predictive capabilities, typical approaches vary with numerous efforts under way to arrive at a workable and functional CMS. Hence, this review seeks to provide an overview of these efforts and tries to assess their current degree of success. The purpose of this review is to: (1) illustrate benefits and obstacles in the development of community model systems; (2) evaluate existing community modeling systems, typically on their technologies and performances; and (3) indicate potential future pathways of either improving or elevating the approaches to the next stage. This overview focuses on some general aspects of

CMS, however, it will for the most part target those efforts that address processes located in the Critical Zone, CZ, of the earth.

2.1 Introduction

The focus of the hydrologic modeling community has been shifting over the past few years away from developing new models that address isolated aspects of water movement to more holistic-view-models that also include chemical, physical, and biological processes within the geo-volumes (while there is no clear definition of where the boundaries of this geo-volume are, in our definition the geo-volume is best aligned with that of the “critical zone” of the earth, which extends from a layer approximately 100 meters in the sub-surface to the 500 mBar layer in the atmosphere) of the hydrologic cycle (Abbott and Vojinovic, 2009; Wagener et al., 2009). Integrated modeling approaches are especially needed for making well informed decisions, i.e. in the context of regulatory actions (Hewett et al., 2010). Development of such integrated modeling systems is typically beyond the knowledge scope of individual researchers, instead requiring team efforts to assemble codes being able to address processes at the envisioned complexity levels. Community modeling, a promising paradigm to develop complex evolving and adaptable modeling systems through a collaborative partnership, has thereby gained more attention in current years (Voinov et al., 2008). With the emphasis on the shared infrastructure and commonality in codes and data, community modeling can for one, improve the efficiency of model development, and for the other, broaden the possible research applications of individual models through integration (Dickinson, 2002). Hence, the concept of Community Modeling System (CMS) brought forward is one that represents the collection of models, pre- and post-processing tools, benchmark data sets and the computing infrastructure such as high performance computing environments or “big iron” storage facilities.

From the perspective of modelers, essential issues coming to fore in the development of CMS are the proliferation of individual models and modules, the credibility of model/legacy codes, the integration of independent models, the interoperability of both model components and accompanying data sets, the infrastructure maintenance, and the question how to best provide a means for the community to add to the CMS. For example, in the area of hydrology, a plethora of models have been developed over the past few decades that are employed in a wide spectrum of areas ranging from watershed management to engineering design (Singh, 2002). Since those code stacks are mostly standalone, and were not considered to be linked with other computational kernels at the time of their creation, it is typically quite difficult to integrate two or more code implementations together, and even more challenging to pull components from different modeling environments and assemble them into a brand new code assembly. The technical impediments includes lack of modular model structure, intertwining of user interfaces and computing kernels, varying computer languages used to encode the modeling kernel, distinct input and output data structures, and poor documentation of source codes to name a few (Rizzoli et al., 1998).

Some of these legacy codes may have proven to be quite popular over the years such as the models developed by the Hydrologic Engineering Center (HEC, 2010) at the US Army Experiment station in Vicksburg, Mississippi (for example HEC-River Analysis System, HEC-Hydrologic Modeling System), which have been widely used in the hydrologic community, however, there are also a vast number of research legacy codes that have only been used for individual research applications and that never had exposure to formal verification procedures (the authors themselves have quite a number of those). While Argent (2004) argues that harvesting and incorporating legacy codes, which represent countless man hours of effort in addition to being a very rich knowledge source, is an appropriate and even necessary step, it is also clear that this is not straight forward because of a typical lack of documentation, lack of credibility of the algorithms or methods encapsulated in the codes, incompatible programming languages, and a general lack of “good coding practices”, for example avoidance of (FORTRAN based) GOTO statements, hardwired constants, or structures that defy modularization and parallelization. Hence, a dilemma often faced is whether to put effort to vet and restructure legacy codes for common use (these codes represent a fairly rich knowledgebase after all), or to launch the creation of new codes and leave the legacy codes behind. Usually this decision will depend on the quality of the codes (and its documentation) versus the difficulty of recreating the code contents.

It stands to reason then that new code developed for use in a CMS environment should feature a high degree of portability and modularity. To meet these demands Object-Oriented (OO) code design has been adopted by many modelers when writing new code environments (Rumbaugh et al., 1991). The OO paradigm has been contrasted with previous modeling methods such as structured analysis or procedural methods in terms of communication, encapsulation, inheritance and polymorphism by Pressman (2001), who found that the modularity and decomposability of new OO based models make their integration with other models much easier. For example, the fact that the OO approach separates the compute kernel from the data allows the independent development of tools for data manipulation, transformation and visualization which can then be reassembled with the modeling code. It relieves researchers and programmers from repetitive programming work and enables them to only focus on encoding model kernels. However, interfaces and code structures are still required to be somewhat standardized by the community in order to ensure communications between models.

In addition to the need for model linking or module selection at run time within a larger modeling framework data interoperability constitutes a second important aspect that is crucial for developing a CMS because of the need i) to access data from disparate data sources, and ii) to ingest these data into the model kernels. Regarding the former issue, some communities and institutions have been set out to provide data access and retrieval systems that can search and access data from national, state, institutional or individual data repositories, and then deliver data to users in the community-defined data formats. Notable attempts have been made by the Consortium for the Advancement of the Hydrologic Sciences, Inc (CUAHSI, 2010) and government partners in the US, the INSPIRE program in Europe (INSPIRE, 2007), and also the Australian Commonwealth Scientific and Research Organization (CSIRO, 2010) all three of which are engaged in developing data models for the exchange of hydrologic data. A CMS can (or better must) take advantage of those systems for retrieving data, even though the respective data models designed by those efforts need to be harmonized (at the time of the writing of this article the World Meteorological Organization, WMO, and Open GeoSpatial Consortium, OGC, have formed a Hydro Domain Working Group to tackle exactly this issue), in addition to sharing data access interfaces and semantic systems within the communities, or alternatively, by developing a set of data transformation tools.

Approaches for CMS development thus far can be broken down into two major classes: firstly those using what we call a monolithic code (of various complexity) class, and secondly a class that features link- or coupling interfaces that permit standalone codes to communicate with each other during execution. Our intention is to provide an overview of the approaches used in the hydrologic community in the context of these two classes, even though we will occasionally make reference to efforts developed in other geoscience communities when appropriate. Also, while advancing the idea and development of a CMS also has a cultural dimension within each community, i.e. the unwillingness of sharing data and individual code preferences that have developed over time, this paper only concentrates on the more technical challenges and aspects of CMS development. We first attempt to classify the systems we know of, which will be followed with the discussion of technologies that can facilitate the development of CMS. We will then focus on a range of code coupling approaches that are currently under development and that lend themselves for forming a CMS. We will also briefly discuss data interoperability, and finally try to give an outlook at future approaches that could be undertaken for building a CMS.

2.2 Classification of Current CMS

It is interesting to note that actually only very few CMS have been either proposed or are being developed by specific communities. Nevertheless, those CMS that have been identified can roughly be classified into three categories. The first type of CMS is regionally limited and only targets the study of processes inside a particular region. It commonly involves a collection of independent third-party model systems or tools along with regional data sets that drive these models. In this case community members enrich the CMS by submitting their models and accompanying data that have been collected and compiled for specific purposes and time frames. A typical representative of this group is the Chesapeake Community Modeling Program (CCMP), hosted by the Chesapeake Bay Research Consortium (CRC, 2010), an open source system of watershed and estuary models that are dedicated to the study of the Chesapeake Bay region on the eastern shore of the US. It contains an assembly of watershed, hydrodynamic, biogeochemical models and additional modeling tools, along with the Chesapeake Bay Environmental Observatory (CBEO) data. However, the involved models are standalone executable programs, sometimes are license restricted, and in general do not provide a true “side-by-side” placement having used similar grid or mesh assemblies, model run timeframes and identical data sets to drive the model runs. Rather, the current focus of the Chesapeake Bay modeling community is more on the application and amelioration of those models, than the construction of more comprehensive model systems via integration. Yet, a set up of this structure permits the addition of other modeling efforts and the community can easily agree on a certain event or time frame that needs attention with several spawned independent modeling efforts running side by side.

Alternatively, a CMS in the second category centers on the use of one specific or monolithic computational code often organized in a modular software architecture. While this model architecture typically features some degree of flexibility by allowing the on- and off-switching of a set of predefined modules it makes the extension more cumbersome as change requests need to be submitted which are then integrated (or not) (Kuo et al., 2004). In this case the development group will determine the usefulness of the request and then extend the code, which makes this a potentially time consuming process. Of course, the advantage lies in the fact that the development team can control version creep as well as test and validate prior to release. A typical example of this type of CMS can be found in the meteorological community in which the MM5 (PennState/UCAR Mesoscale Modeling 5th Generation, MM5, 2003) or the more recent Weather Research and Forecasting model (WRF, 2010), have been designed for mesoscale numerical weather prediction (Michalakes et al., 1998; 2001). The WRF model development is conducted through a set of sixteen lead groups, each of which concentrates on one particular task, such as the development of numerical software, the maintenance of model architecture, and the integration of models from related domains. A similar case is the family of three dimensional ModFlow codes (USGS, 2009) used to model sub surface flows (another popular system is ParFlow; LLNL, 2010). Its modular structure has enabled the integration of some additional simulation capabilities, for instance, simulation of surface-water, solute transport, aquifer-system compaction and land subsidence.

There is an entire other class of community type systems that could also fall into this category. These concern codes for the simulation of river, estuarine and coastal process and encompass quite a long list with codes such as DELFT3D (the newest member added in January 2011 from DELTARS, http://www.deltares.nl/en), ROMS, TOMS, EFDC, SELFE, CH3D to name just a few from this list. It is clear that each of these codes has its group of followers and strictly speaking once it is more than a handful then one could actually think of it as a user community. However, we do not seek to explore this line of reasoning because there are quite a number of codes in this sub-domain (unlike for example ModFlow/ParFlow or MM5/WRF), and the estuarine/coastal modeling community is in fact quite divided on whether a specific code is better than another. Also, for many of the codes the community is not engaged in a coordinated effort to improve these codes (this is often left to the original developer or an institution that adopted the code) that are either left alone as is, or are being “bastardized” into a myriad of derivatives as the original source code is altered by each individual who downloads the latest known documented version. Because this specific code landscape is so diverse we decided not include it into our discussion, however felt it worth mentioning.

While the idea of forming interest groups to develop a new monolithic code structure has the advantage of bringing many minds to bear on the development thus ensuring substantial intellectual focus and breadth, it is a fairly time consuming task because of the large development group and the need for an organized versioning system. In addition, the tight source code control typically delays the transfer to other operating systems and also prevents the harnessing of a much broader community for code contributions. This in turn limits the incorporation of modules and externally developed code (including legacy code) and also the porting and integration of code segments written in other programming languages.

The third type category CMS has a generic component-based modeling framework that can integrate models and build up multi-component model systems thus permitting a substantial degree of flexibility. Examples are the Community Surface Dynamics Modeling System (CSDMS) (Peckham, 2008), the Partnership for Research Infrastructures in earth System Modeling (PRISM) (Valcke et al., 2006) and the Earth System Modeling Framework (ESMF, 2010). The CSDMS uses a strategy called Common Component Architecture (CCA), which involves a set of tools and standards for modularizing component modules (Bernholdt et al., 2006). It also contains a language-interoperability tool called BABEL (Dahlgren et al., 2007), and a Graphical User Interface (GUI) for linking component modules within the high performance computing environment called Ccaffeine (http://www/cca-forum.org/ccafe). BABEL can generate "glue codes" for component modules written in different programming languages, including C, C++, Java, Fortran and Python. The CSDMS currently involves a variety of terrestrial, marine, coastal and hydrological modules that originate and were submitted by community members, and have been or will be modularized as linkable component modules. The PRISM framework employs a stand-alone coupler called OASIS to handle synchronized exchanges of coupling information between numerical codes, a Standard Compile Environment (SCE) to retrieve and compile source codes, and a Standard Running Environment (SRE) to maintain model execution. A child model of PRISM is portable, usable independently and interoperable with siblings, and freely available for research (Valcke et al., 2004). Finally, the ESMF framework is a hierarchical collection of components that can be combined to form larger scale models such as the atmospheric circulation model (GEOS5) that NASA deploys at its Goddard Space Center. Components can be comprised of physical domains on the earth surface such as hydrologic domains (lakes, rivers, etc.), but also chemistry, vegetation and catchment processes, as well as atmospheric turbulence and radiation models. The system permits the use of parallel computing environments and through its coupler functionality the module execution in sequential or concurrent mode (Hill et al., 2004; Collins et al., 2005). Coupling frameworks on the other hand seek to provide code into which external code can be embedded or linked to in an attempt to overcome difficulties that one encounters during the process of model integration, i.e. disparate model interface definitions, mixed programming languages, difference in data semantics, and incompatible spatial and temporal scales etc (Holzworth et al.,2010). Since they can provide technical supports and participatory platforms for community modeling, they represent a promising alternative path to develop a CMS and as such warrant a separate section in our review.

2.3 Using Coupling Frameworks in Developing CMS

2.3.1 Components of Coupling Frameworks

The literature is quite rich on approaches of how coupling frameworks could be designed and it has been reported that the advancement of coupling frameworks and increased code commonality facilitate the creation of more cohesive and collaborative communities (Killeen et al. 2006). We will try to give an overview on those that seem to be the most prominent ones currently being pursued. In a somewhat simplistic view, coupling frameworks are software layers that "glue" together component or modules during run time in such a fashion that data can be moved in and out of these components together with time (time manager) and spatial (re-gridding or spatial interpolation) control. In other words, as long as each of the components or modules abides by the rules and protocols set forth by the interface (or "glue") definitions any code can be linked to another code during run time. When trying to incorporate legacy codes this leads to the need of writing so-called wrappers that mimic the coupling framework interface and hide the legacy codes' historic I/O definitions. While the range of features of a coupling framework largely depends on the requirements of its problem domain, we have tried to identify the most common ones that can be summarized as follows:

- **Model standard or protocol**: is the kernel of coupling frameworks that commonly comprises standard interfaces that component modules should comply with, descriptions of model structure, data model, metadata tags, and some other abstract standards.

- **Module Library**: Modules represent context-independent software units that can be separated from their original code base and turned into standalone executables with moderate change (Ciupke and Schmidt, 1996). They are standardized, portable, and usually are made available in form of a dynamic-link library (DLL) or a COM object. A module can encapsulate scientific concepts and algorithms or just be a service module.
- **Data Analyst**: It contains tools for data analysis, for instance, geospatial data processing, data statistics, data interpolation or extrapolation etc.
- **Toolbox:** It contains sophisticated tools and utilities that facilitate the development of component modules, such as optimizer, ODE solver, unit converter, tools for wrapping or converting legacy codes, data flow monitors etc.
- **Workbench**: is a platform for model linkage, execution and management, which usually supports graphical, icon-based model construction (Maxwell and Costanza, 1996).

Studies on coupling frameworks can be traced back to the 1990s, and the Modular Modeling System (MMS, Leavesley et al., 1996a) is an early attempt. The MMS represent a hybrid approach between a traditional stand-alone model system and a component-based coupling framework. It is similar to the former where modules stay as source code files and will be compiled and linked as executables during the linking process, and on the other hand, incorporates most features of the latter as summarized above. The only exception is the absence of the standard interfaces that contribute to make modules compatible. Bongartz et al. (2003) pointed out that the MMS is an objected-based rather than an object-oriented coupling framework, which does not support features such as abstraction, inheritance, and encapsulation. David (2002) then adopted the basic idea of MMS, and presented a pure Java, object-oriented framework called Object Modeling System (OMS). One highlight of the OMS is that it takes advantage of the introspection feature of the Netbeans (http://netbeans.org/) Independent Development Environment (IDE), thus supporting integration of component modules via metadata-tagging and reflection techniques (David et al., 2004). The same idea can also be found in the Interactive Component Modelling System (ICMS, Reed et al., 1999; Cuddy et al., 2002) and The Invisible Modelling EnvironMent (TIME, Rahman et al., 2003; 2004a). While the former is built by a self-developed C-like language along a debugger called ICMSBuilder, the latter is born from the .NET IDE (Meyer, 2001). The introspection mechanism embodies the concept of inducting the declarative language into model development. Fekete et al. (2009) presented the idea of developing a declarative framework called the Next generation Framework for Aquatic Modeling of the Earth System (NextFrAMES), which attempts to provide a high level abstraction of the scientific tasks. It provides an eXtensible Markup Language (XML) schema for describing model structure, along with a run-time engine that interprets the modeling XML, loads the modules, establishes the linkage, and executes the model (Lakhankar et al., 2008). The NextFrAMES centers on using the declarative language to integrate component modules, whereas the Spatial Modeling Environment (SME, Maxwell and Costanza, 1994), another early attempt of coupling framework, focuses on integrating component modules encoded in declarative language. It employs commercial declarative modeling environments like STELLA (www.iseesystems.com/softwares/ Education/StellaSoftware.aspx) to create modules that perform certain computations over a spatial unit, e.g. a grid cell. Those modules can then be loaded to the library, converted to C++ objects and executed within the geospatial context (MaxWell and Costanza, 1995; Maxwell, 1999). The declarative modeling approach (or system modeling) is advantageous in linking elements declared in the model with entities declared in a distributed knowledge ontology, however, the absence of a standard and common declarative modeling language often slows its reuse in other applications (Argent and Rizzoli, 2004; Argent, 2004).

The most representative coupling frameworks that incorporate the common features summarized above are the Open Modeling Interface and environment (OpenMI, Gregersen et al., 2005; 2007; Moore and Tindall, 2005), the ModCom framework (Hillyer et al., 2003) and the Tarsier environment (Watson et al., 2001; Watson and Rahman, 2004). These linking kernels represent a set of standard interfaces that describe, link and run compatible models (Knapen, 2009). Since integration work is handled solely by the interfaces and such remains de-coupled from the “scientific” modules, module development is fairly uncomplicated with few and manageable constraints. Another alternative has been presented by Campbell et al. (1998) who introduce the Dynamic Information Architecture System (DIAS) that uses standard abstract classes to specify “entity objects” and their dynamic behaviors. The “entity objects” conceptualize the real-world entities in ecological systems such as atmosphere, ocean and fish, and the dynamic behaviors represent simulation models. The DIAS allows building, manipulating and simulating complex ecological systems, in which multiple objects interact via multiple dynamic environmental and ecological behaviors (Hummel and Christiansen, 2002).

We briefly introduced the architecture of a group of coupling frameworks above. They are selected because they represent different efforts made for the progress of coupling frameworks. In the following section, we will discuss in more detail on associated techniques of coupling frameworks, as well as the commonalities and divergences of those coupling frameworks.

2.3.2 Modes of Integration

Bulatewicz (2006) summarized four approaches based on how model codes are integrated: the monolithic approach, the scheduled approach, the component approach, and the communication approach. In the monolithic approach, pieces of code are taken from different programs and merged together to form a new program. As the result of introducing the concept of modular and hierarchical decomposition of models and the subsequent emergence of the object-oriented programming design as the dominant programming methodology, models have become much more modular which has served as a successful base for the advancement of integration approaches (Padulo and Arbib, 1974; Gamma et al., 1995). The other three approaches are similar in their using of code blocks and as such resemble in the modular structure of the monolithic code approach but they differ in their code functionality and arrangement. In the scheduled and communication approach, the software components are independent models conducting scientific computations, and execute in a scheduled order and through messaging passing. The referred models can be further decomposed as a set of fine-grained software components each on which would be responsible for a specific function; this sub-division of a sub-division represents the main idea of the component approach. Among the coupling frameworks we mentioned here, the NextFrAMES falls into the scheduled approach group, while the SME, DIAS and Tarsier adopt the communication approach, and the rest utilizes the component approach.

In contrast to the previous classification approach, Brandmeyer and Karimi (2000) presented a classification of methodologies using a five-layered pyramid with ascending order of complexity and sophistication: one-way data transfer at the pyramid base, loose coupling, shared coupling, joined coupling, and finally a layer on top for the tool coupling approach, as shown in Figure 2.1.

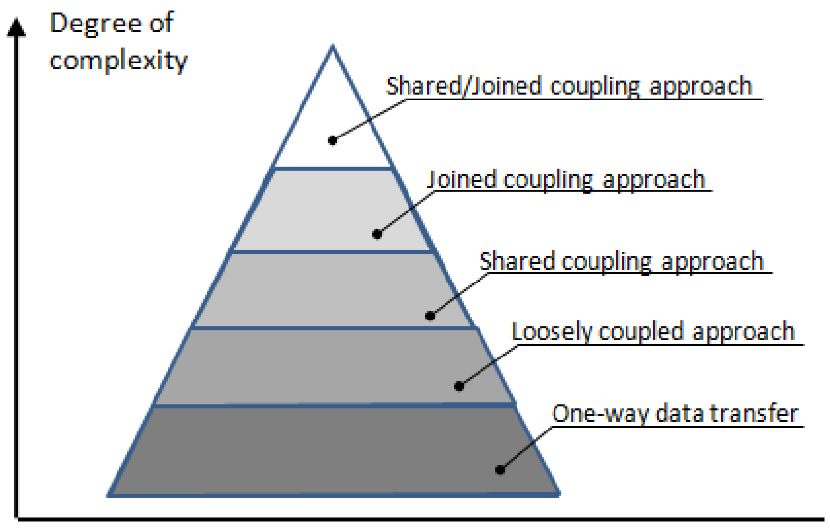


Figure 2.1 Classification of Coupling Methods according to Brandmeyer and Karimi

The lowest level, i.e. the one-way data transfer, in which two completely separate models have a single directional data transfer requiring manual user control, may not be even taken as a genuine coupling method. In this approach it is the users' responsibility to access the output of one model and adapt or re-format it to be the input of another model; an approach that is very common. The next level upgrade of this method is the loosely coupled approach, which supports bi-directional data transfer and permits users to be entirely hands-off on the data transfer. This approach works off a list of standardized file structures in which one program is expecting a certain file to appear in a specific sub directory at some point during run time, and vice versa. In the shared coupling approach, models are integrated with the aid of a shared component, for example through a GUI or a database. While models in these first three approaches stay and execute as independent applications, the two remaining higher-level coupling schemas feature a higher degree of integration. In the joined coupling method, one model takes the dominant role and the other models are integrated via plugins, thus becoming secondary code insertions into the main or lead code. The concept of the top-level tool coupling is a hybrid approach representing a combination of the ideas embedded in the shared and joined couplings.

In our opinion, either classification scheme is as valid as the other as they provide an attempt to systematically list the various approaches. Inevitably, many of the model codes, frameworks, and coupling approaches mentioned in this manuscript can be classified according to these two schemas and it is less relevant to find a place for each and everyone in the schemas presented, than to simply outline what the underlying ideas are.

2.3.3 Component Model Standardization

Ideally component modules should be independent standalone code entities that can be analyzed separately and then be merged to form more complicated model systems (Voinov et al., 2008). For some of the coupling frameworks introduced earlier, i.e. OpenMI, Tarsier, and ModCom, component modules are compiled into dynamic link libraries (DLLs) which can be loaded by the execution manager at run time. In other coupling frameworks, component modules can be pre-complied source codes (as in MMS) or metadata-based models specified by declarative languages (in SME) (Abel et al. 1994). While a component module commonly remains as a “black box” to the external environment, some degree of standardization can be achieved fairly easily if the component module is modified to expose an agreed upon set of information. There are two requirements for this type of standardization. First, while component modules of an integrated model typically execute in sequence or parallel being controlled by a run time manager, the underlying technology is based on a small set of designated methods embedded in the source codes, which are called to perform initialization, main computation and termination. In this approach each component module belonging to a coupling framework must use consistent method signatures which the run time manager can recognize even it requires customization. For example, the standard interface of the OpenMI (OpenMI, 2010) contains the methods *Initialize()*, *Prepare(), GetValues()*, *Finish()* and *Dispose()* etc., whereas that of the ICMS contains *initialization(),main()* and *finalization()*. We have listed a summary of these and other methods in Table 2.1.

The second requirement is that component modules should expose their input and output definitions so that a linkage can be set up via mapping the output of one module to the input of another explicitly or implicitly. Some coupling frameworks adopt a metadata-based approach, in which component modules are required to self-document their source codes, where input/output properties will be assigned to certain variables along with other metadata. The TIME and the ICMS frameworks introduced earlier feature this type communication approach. While TIME relies on the metadata-tagging feature given by the third-party .NET environment, ICMS achieves this through a proprietary declarative programming language called MickL (Rahman et al., 2004b). The OMS and the NextFrAMES approaches are similar cases but instead of embedding metadata annotations in the source codes they define input/output quantities using XML-encoded declarations in external files. A more common approach is to specify exchange items in standard interfaces. For example, component modules using OpenMI should implement the *IExchangeItem* interface, which specifies exchanging items as *Quantity* and *Elementset*. A *Quantity* contains metadata of a variable, like ID, description and unit, etc, while an associated *Elementset* provides spatial information of the *Quantity*. ModCom uses an approach that is quite similar to OpenMI, where component modules need to implement the *SimObj* interface. The Input/output variables are defined as SimData class and can be exposed via the *ISimObj.Input* or *ISimObj.Output* properties (Hillyer et al., 2003).

Table 2.1 Summary of technologies of the coupling frameworks

Short Name	**Model Builder /Simulation control**	**Component model format (Acceptable Source codes)**	**Methods in standard interfaces**	**Input/output specified in component models**	**Communication mechanism**
MMS	GUI Xmbuild	Executables/ Source codes file(Fortran/C)	declare(), initialize(), run(),main()	N/A	File transfer or sharing central database
SME	Configuration file	SMML Object(C++/ Java,Fortran,C)	N/A	Specified in Frame classes	Message passing
ICMS	GUI ICMSBuilder	plugin(MickL)	initialization(), main(),finalization()	Variable/fields attributes	Input-Output mapping
DIAS	GUI(GeoViewer)/Context Manager	N/A(Any)	N/A	Register input/output parameters	Indirect communication via domain objects
Tarsier	GUI(Tarsino)	DLL(Bortland C++)	execute()	RegisterFields function	Data sharing and message passing
OMS	GUI(Model Editor)	Jar(Java/ Fortran/C)	init(),run(), cleanup()	Attribute Editor	Input-Output mapping
ModCom	GUI/ISimEnv interface	DLL(Any supporting COM)	StartRun(), EndRun(), HandleEvent()	Input/Output Class	Message passing interface and I/O actions from/to disk files
TIME	GUI	DLL(VB/C#/ Fortran/C++/ Java)	runTimeStep()	Variable/ fields attributes	Input-Output mapping
OpenMI	GUI Configuration Editor	XML file(.omi) + DLL(Any)	initialize(),Prepare(), GetVaues(), Finish(), Dispose()	Input/Output Class	Request-reply mechanism
NextFrA-MES	XML file	plugin(C/C++)	initialize(),execute()	XML statements	Input-Output mapping

The above requirements for standardizing component modules are discussed under the assumption that they are written using a programming language that is supported by the coupling framework. For example, for the .Net version of OpenMI, component modules can be encoded using VB.Net or C# programming languages which allow the run time manager to invoke the embedded methods and parse input/output items. In contrast, for a large number of modules that are written in other languages OpenMI is not directly usable. The mixed-language issue could be addressed by performing language translations and re-coding, either manually or using translation tools (e.g. JNBridge, www.jnbridge.com). However, neither manual transformations, which can be complicated and prohibitively time-consuming, nor the translation tools, which often have difficulties to adequately translate complex programs, provide for a high degree of success. Alternative approaches seeking to overcome of the aforementioned difficulties center on the design of “glue” code interfaces (Argent and Rizzoli, 2004) written in conventional programming languages with the intent to wrap the original programs. This idea has been adopted by the OpenMI, which provides a set of wrapper interfaces assisting the migration of legacy code (note that this works only for those codes that can be executed in compatible batch mode).

Another alternative approach is the use of web services which support machine-to-machine integration and interoperation of web-based applications. This idea would entail the creation of web services based modules that can be accessed via standard interfaces, such as the Web Processing Service (WPS), an Open GeoSpatial Consortium standard, or customized web services designed for model communications (Horak et al., 2008). Figure 2.2 shows a simple example, where Model A is a standardized model located in one machine, and Model B represents a web service based model located in one remote machine. On the local side, an additional wrapper would be needed in order to tap into the web services of model B.

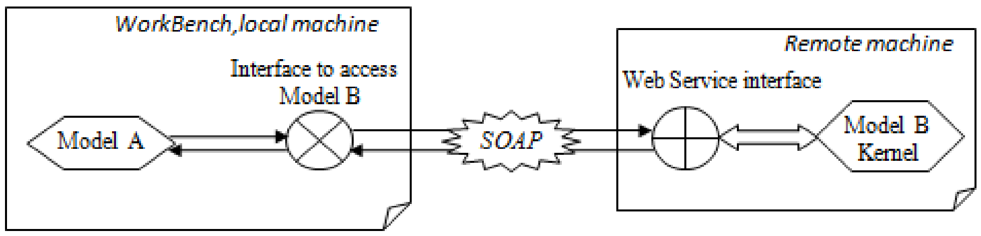


Figure 2.2 Communication of Web Service based Models

2.3.4 Communication mechanisms

Data exchange is the primary communication task between modules within a modeling system. In some early integrated models, modules achieved integration through transferring data files or sharing databases only. For example, Leavesley et al. (2005) coupled the BOR RiverWare model (Fulp et al., 1995) with the MMS via a shared relational database. The MMS simulated streamflows and wrote the results into a database, while the RiverWare model read them and proceeded to evaluate reservoir-management strategies. To facilitate this setup customized data management interfaces (DMIs) need to be written for assisting the database to bridge the communications among the participating actors (Leavesley et al., 1996a; Leavesley et al., 1996b). This repetitive reading from and writing into a database approach mostly suffers from slow execution times slowing down the overall model progression. As a result of IT infrastructure growth on data pipelines and improved data and machine communication protocols, most coupling frameworks now seek approaches that allow component modules to communicate dynamically and seamlessly in addition to using new paradigms of how data collections can drive physical models (Gourbesville, 2009).

One approach adopted by some coupling frameworks is based on a *request-reply mechanism*. An example for this type of approach is again the concept behind OpenMI. During the initialization phase, it requires each compliant model to implement the “GetValues” method which allows the retrieval of data from another OpenMI compliant module during run time. The OpenMI uses a purely single-threaded architecture where only one data request is handled at any time (Fortune et al., 2008), however it now includes both quantitative (numbers) and qualitative values (such as “dry” or “wet”) as descriptors.

Another approach can be described as *input-output mapping*. In some coupling frameworks the linkage between two component modules can be explicitly specified by matching the output of one module with the input of the other using GUI tools. For example, the ICMS provides a dragging-and-dropping platform where component modules or model objects (e.g. sub-catchment, stream) can be connected via a link-arrow. The underlying connection is then defined by configuring the link properties. As a modification to this approach some coupling frameworks such as NextFrAMES specify those linkages implicitly; for the case of NextFrAMES via defining input/output components separately in a modeling XML file including the definition of variables passed. The variable names are then mapped to related variables declared in the sources codes of the plugin embedded in a component model (Fekete, 2009). This type of semantic link enables the data transfer from one module to the other, with the added advantage that point-to-point mappings often avoid semantic ambiguity and confusion. Note that this approach requires the need for a sequential execution of the component modules to ensure that a data provider model is executed before the next-in-line data consumer. From this point of view, this approach lacks some of the flexibility inherent to the request-reply mechanism, which supports triggering the execution of a data provider model whenever it is needed. The last approach in this line is based upon *in-memory data sharing and message passing mechanism*. The Tarsier framework is an example, in which the model structure and communication protocols are based on the 'observer' pattern of client-supplier computing (Gamma et al., 1995). Data can be registered as "*usees*" and shared among models and tools called "*users*". If two *users* use the same *usee*, they are implicitly linked. When one *user* changes the *usee*, it will send a message to the other *user*, which will respond immediately to the data changes (Watson et al., 2001). The message passing is handled by the *SendMessage* and *ReceiveMessage* methods implemented by *users*.

A key prerequisite of establishing valid communications between modules is that these modules are interoperable. Howie et al. (1996) defined interoperability as the ability of different programs to share and process information irrespective of their implementation language and platform. In the hydrologic domain, the component modules should also be interoperable with respect to spatial and temporal scales which could differ in format, resolution and reference system. For example, regional climate models mostly provide meteorological estimates (precipitation, temperature, air humidity and wind speed) with a spatial resolution of a few kilometers, whereas distributed hydrologic models are normally built up for higher-resolution analysis with grid sizes in the tens of meters range. When coupling them together, a step of downscaling transformation should be put in the middle in order to reconcile the scale difference. The downscaling could be carried out via different approaches. For example, Marke et al.(2001) presents a statistic approach that introduces the correction of biases into a pragmatic approach for the downscaling of precipitation (Früh et al., 2006), and Cubasch et al. (1996) investigates approaches of direct interpolation of the nearest grid points, time-slice and statistic downscaling experiments.

It is clear that the degree of interoperability will therefore hinge on the degree of sophistication that has been implemented in the “interoperability tool box”. Unit conversions, syntactic (format) transformations, temporal and spatial interpolation and extrapolation capabilities, semantic mediation (for example of keywords and variable names) are all services that are ideally embedded into this tool box which should be able to automatically act whenever it detects an incompatibility. Currently, the majority of the coupling frameworks feature only a limited set of transformation tools.

2.3.5 Coupling Frameworks and their Use for CMS

As pointed out in the previous sections coupling framework exist in a number of flavors using different implementation strategies. They basically fall into two categories; those that use declarative statements in which the number and type of modules are recorded as well as the sequence in which the modules will be called, and then those that use a graphical user interface of some sort allowing the construction of module execution sequences using visual aids. The former typically uses a set of configuration files to describe the overall model structure, i.e. component modules and their connections. An engine accesses the configuration file at runtime, parses the model hierarchy, loads specified component modules, invokes the model computations, and performs the model I/O. Into this category we can group NextFrAMES, SME and PRISM. For large model composed of many component modules, however, the preparation of the configuration file(s) can be quite complicated and time-consuming in addition to requiring users to have a clear understanding of the coupling hierarchy.

Working with a GUI workbench is a lot more user-friendly of course and from a users' point of view visual aids for dragging and dropping connections supported by drop down menus to navigate module libraries is quite simpler than encoding configuration files. Examples for coupling environments providing GUIs include OpenMI, ICMS, and OMS. In most of these systems, an integrated model can be exported as a project file, which can be re-loaded to the workbench, thus providing some degree of repeatability and provenance.

Whatever the adopted approach is, we believe that coupling frameworks can serve very well as the backbone of a CMS. One of the key advantages lies in the fact that legacy code can be migrated into the environment, admittedly with some work, so it complies with the necessary I/O and interface definitions. However, the convenience of legacy migration is not the only aspect to consider as there are also the frequency with which the coupling framework may experience upgrades and changes as it matures further (hence, how mature is the chosen one?). Additional points to consider concern the extent of the existing library of compliant modules and codes, whether or not the system enjoys a large group of developers or an active user community from which to draw support, a rich support library that contains peripherals and visualization applications, and if the framework supports high performance computations on parallel machines or the cloud.

Among the discussed coupling frameworks, NextFrAMES is still under development, and some of the others such as the OMS and the DIAS only have few applications to date (Sydelko et al., 2001; Kralisch et al., 2005). The ModCom framework while developed by a (small) group of researchers and therefore having some manpower behind it appears to have found little acceptance in practice; we found little evidence in -form of published references during the literature search that would demonstrate use of it in the wider modeling community. Also, being built on a Unix-based platform, MMS and SME may not have the chance to gain extensive popularity in the largely Windows-based hydrologic user community. Those frameworks developed using less compatible languages such ICMS (using MickL) and Tarsier (Borland C++) may be hindered in their degree of utilization and spread in the community. However, TIME and OpenMI have attracted a fairly large group of researchers, with TIME having incorporated most of the well-known hydrologic models in Australia (Bari et al., 2009; Rassam et al., 2009).

2.4 Summary

In this paper we attempted to review current efforts in building modeling systems for the hydrologic community and to summarize some of the salient points, both advantages and disadvantages, of the systems that we have been able to review. We deliberately focused on systems that are directed towards the hydrologic community with some efforts on the periphery, such as atmospheric or estuarine/coastal modeling systems. We felt that these efforts have a place in this review as they are related to the hydrologic realm and also demonstrate some basic features that are potentially common to any community modeling system regardless the specific community. The paper focuses on the use of two distinct strategies to establish a Community Modeling System: the first focuses either on a geospatial extent as a defining frame, or uses a single monolithic code, or uses a single complex multi-module code base; and the second focuses on coupling (or “glue”) frameworks bringing together codes under one umbrella.

We classify current CMS into three categories. The first one uses a specific region as an organizing principle for which any number of models can be developed with no particular requirement as to what these models should be. This somewhat lose organizing principle has the advantage, however, that additions are quite easily accomplished and exchange of data, information and results is fairly straightforward. The real strength lies in the ability to execute parallel efforts and to compare results that were arrived at using different methods. The next group uses a single code structure as the underlying organizing principle, i.e. instead of single-region-multiple-codes, it organizes around the idea of single-code-any-region. A key advantage in this system is that a single code used by the entire community allows for a high degree of sophistication of this code as the pool for suggestions and improvements is vast. In addition, it limits the need for a large set of pre- and post processors as the same utilities are used by everybody and also allows easy exchange of I/O data among fellow users. The third group concerns multi-module modeling frameworks that seek to combine the computations of processes in many geo-volume strata addressing the linkages that exist between the movements of water and the many bio-, chemical-, and physical (and even economical and life cycle) processes that are present in the hydrologic realm. The key advantage in these systems clearly lies in their potential to provide more holistic views of the environment in which the complex melee of coupled processes are accounted for. Community contributions to code development or benchmark case studies are possible in these systems as many of them feature gateways for externally developed code insertions in case the default modules need to be swapped out.

Our second focus was on highlighting the features of coupling frameworks and how these features pertain to CMS development. In short, it is fair to say that they offer a number of advantages for the development of a CMS. The advantages of using existing coupling frameworks include: 1) The component-based architecture enables the CMS to incorporate new component modules easily. This feature can make a CMS a powerful tool and also create the spirit of a true community environment. 2) Coupling frameworks provide infrastructure for mediating the execution and communication between quite disparate models, thus community members only need to focus on the migration of their own models. 3) The portability and reusability features of compatible modules enable users to construct more complex code assemblies that have a wide range of applications. 4) Legacy codes can be wrapped to become compliant modules typically without the need for extensive code modifications by using development tools. 5) Some coupling frameworks support parallel computations, which can be possible venue for improving modeling efficiency. 6) A potentially large user community provides a good feedback pool that can be used to improve the coupling framework.

There are, however, a few disadvantages as well. First, some of the frameworks are established at a basic computational level, often requiring a somewhat steeper learning curve for using the chosen coupling framework. For example, a coupling framework may provide a standard set of interfaces encoded in certain programming language, which makes it difficult to adopt or use if the user has little to no experience using this language. Second, when wrapping a piece of legacy code the effort to wrap the code increases dramatically if the code is complex and monolithic in its structure and features complex data models that are not easily modified to work well with the chosen interface definitions. Additional work may be necessary to separate intertwining interfaces and to partition and modularize monolithic codes. Third, coupling frameworks have been criticized to miss out on the opportunity to formulate a new modeling paradigm that seeks to couple governing equations of the hydrologic processes present in each geo-volume strata in a holistic fashion. This is to say that the coupling framework may perpetuate the existence of inadequate models (and the errors they produce) by coupling one inadequate model to another, thus producing a seemingly better outcome while in reality only adding faulty results of one model to the faulty results of another. In other words, a coupling framework may facilitate data exchange as run time, but does not link these models on a more conceptual and theoretical platform.

We pose that future work on the CMS development is likely to happen along a more Darwinian evolutionary track: many of the systems presented here are still in a process of being developed, while others already have some degree of acceptance, while others may or may not survive in the future. In other words, future work will be spread across many different pathways and will involve try-outs and testing, as well as observing what happens as developments move along. In light of this, it is consequently quite difficult to arrive at a verdict as to what is best, or most promising, versus what has little chance of surviving or meeting community expectations. For this much of the work currently carried out is still too much in its infancy. However, it is also clear that it is hard to predict what technological developments will bring in the future and how these developments may impact one CMS versus another. For example, the increased use of internet technologies such as web services or cloud computing may open up opportunities hitherto unknown. There are also other frameworks that could be explored for CMS development such as workflow engines. These are relatively new arrivals on the computing scene (at least for hydrologic modelers) and have not yet been investigated for their suitability to serve as a backbone for a CMS, despite obvious advantages such as: a means to record provenance, automatic versioning system, smart connections to web services and any type of data store, ease of library development, coupling of workflows from pre- to post processing, to name just a few. The questions that could be answered here are how easily legacy could be transferred into workflow activities (or actors), or how computing performance would suffer if executed through workflow engines, or how applicable or universally transferrable is a CMS when attempting to use it any location across the world (or initially perhaps the US). Future studies could also focus on developing a system of CMS in which CMS could be linked through middleware, or middleware that permit to push any CMS computation out into the high performance computing arena, or frameworks that help to overcome semantic disparity between models in general and CMS in particular.

CHAPTER 3: DEVELOPMENT OF A HYDROLOGIC COMMUNITY MODELING SYSTEM (HCMS) USING A WORKFLOW ENGINE

Abstract

Hydrologic models can be conceptualized as a workflow sequence carrying out a suite of data-driven operations at various levels of complexity. It involves functionalities of data accessing, harvesting, digesting, processing and analysing. In this paper, we present our development of a workflow engine (TRIDENT) shelled hydrologic community modeling system, which supports customizing hydrologic-modeling-oriented workflows by means of sequencing independent and swappable modules. We focus on developing four libraries: a data retrieval, a data processing, a hydrologic computation, and a data analysis library of activities. The data retrieval library allows for accessing data from online repositories via SOAP or FTP protocol based machine-to-machine communications, as well as from varied types of data files such as Excel, SQL database, and NetCDF among others. The data processing library is designed to bridge the gap between source data and model input data, which performs a number of data transformations including geospatial data analysis, interpolation/extrapolation and unit conversion. The modeling library consists of a number of activities that encapsulate traditional hydrological methods for simulating single hydrological process such as evapotranspiration, surface runoff and channel routing. In addition, the modeling library includes a semi-distributed hydrologic model, namely the TOPography based hydrologic MODEL (TOPMODEL) along with the hydrologic components of the Soil and Water Assessment Tool (SWAT). The fourth library encompasses activities for post-processing such as uncertainty analysis, data storage and visualization. To demonstrate the applicability and feasibility of this modeling environment, a simple hydrologic modeling

workflow is developed and applied to Furnace Creek watershed (located in south eastern Pennsylvania) for simulating a flood event.

3.1 Introduction

Hydrologic modeling systems are evolving along the same lines of innovations that accompany the increase of computational power (with decreasing costs), internet connectivity (volume and speed), and the realization that hydrologic modeling is a much more complex task than previously considered because it requires accounting the water pathways and stores in all geo-volumes simultaneously. What we mean by this is the need to overcome the conceptual separation of subsurface water, surface water, and atmospheric water modeling approaches and all the processes that govern and influence the flux of water through the environment, and instead to take a holistic view in which traditional horizontal layers in the geo-volume are coupled in vertical directions as well. Hence, there is an increasing need to develop coupling frameworks for traditional (layered) hydrologic models in the context of community modeling systems or as comprehensive modeling frameworks. These have been introduced at various levels over the past few years and some success has been achieved (Bo and Piasecki, 2011, in review); we refer to the referenced paper for a detailed review of community modeling systems and strategies to link models together. Some of these approaches are quite complex in their structure, such as the Earth System Modeling Framework (ESMF, 2010), or the Community Surface Dynamics Modeling System (CSDMS, Peckham, 2008) attempting to address as comprehensive a representation of the environment as possible. Others focus on establishing a coupling framework in which legacy codes are coupled at runtime and progress through simulations hand-in-hand (such as OpenMI, 2010), while yet others try to build monolithic codes in which a multitude of equations governing flow in the vertical and horizontal directions are numerically integrated over the geo-volume (such as PIHM, 2010). However, in none of them important issues such as data provenance, repeatability of model executions, and ease-of-adding-to-the-modeling-environment, linking execution chains from preparing to analyses has been addressed satisfactorily so far. The reasons for this are manifold: this could be the result of proprietary code and resulting inability to add new pieces, lack of general pre- and post processing components, absence of proper model execution and data preparation documentation, inadequate code maintenance, or simply being “stuck” with a model development strategy that was formed years back and that is not easily changed.

A workflow environment on the other side, presents numerous advantages and possibilities that can be utilized when trying to address some of the shortcomings mentioned above. For example, the ability to create detailed provenance information for each model run allows not only for model run documentation but also the possibility to completely record this workflow sequence as a file. In addition, the ease with which components can be added and modified is ideal for making this a community modeling environment in which formalized testing of new components can take place in addition to easily building libraries that contain various versions of the same process encoding, or alternative formulations that can be easily swapped in and out of a modeling sequence. Other advantages include the coupling with cloud computations and also a plethora of auxiliary services that range from data format conversions, to automated data annotations, to unit conversions, to the automate degeneration of added-value products during post processing to name just a few. Hence, in this study we seek to explore the advantages of using a workflow environment for creating a hydrologic modeling framework, which is functionally analogous to a coupling framework for modularizing and integrating models and at the same time addressing some solutions to the aspects raised above. We will provide an overview of workflows in general and then the TRIDENT workflow engine, followed by a section in which we describe our libraries, and then a small case study to show the utility of the developed system. The paper closes with a summary of the current state of the hydrologic workflow environment and presents an outlook on future activities.

3.2 Workflow Engine

3.2.1 Overview

We will use the “workflow” paradigm as a conceptualization for the representation and management of distributed scientific computations where we can either encapsulate an entire numerical kernel into a single computing node or install the modeling components and its sub-components in a sequential execution string (Gil et al., 2007). Workflow engines are software applications that can facilitate composing, executing, archiving and sharing scientific workflows. Popular and widely used scientific workflow engines for scientific computations include KEPLER (Ludäscher et al., 2006), Pegasus (Deelman et al., 2005), Triana (Taylor et al., 2003; 2007), Taverna (Oinn et al., 2004) and Microsoft’s TRIDENT (Microsoft Research Group, 2009). The latter in particular supports high-performance computing (on MicroSoft’s “Azure” cloud environment), while it is common to all engines to feature the options of parallel or concurrent execution, and distributed computations in the Grid environment (Pautasso and Alonso, 2006). All engines provide the service of recording provenance, a detailed history of each execution of a workflow, including the diagram of the workflow, workflow input/output data, and some other metadata associated with the workflow and data products, thus guaranteeing the repeatability of specific scientific execution sequences. Additional features like scheduling workflow executions, monitoring workflow progress at runtime, tracking sources of errors, determining data quality, and interrogating histories of workflow executions are also incorporated (Simmhan et al., 2005; Goderis et al., 2008). All workflows can be shared through publication mechanisms or repositories such the MyExperiment site (http://www.experiment.org).

It is beyond the scope of this paper to examine the pros and cons of each workflow in order to arrive at a best choice. While they differ in their capabilities (for example KEPLER has an extensive library for data connections and data manipulations, while TRIDENT features tight integration with other MicroSoft software suites such as OFFICE, AZURE, or SILVERLIGHT) and also in the way the workbench is organized, they more or less provide a similar environment. In our case we decided to work with the TIRDENT workflow largely because we had access to it as a beta site tester (this ensured on-the-spot help and support) and also because much of our previous application development had been carried out using MS software stacks. The TRIDENT workflow engine supports: (1) high-performance computing, parallel or concurrent execution, and distributed computations in the Grid environment (Pautasso & Alonso, 2006), (2) capturing provenance information where provenance is comprised of workflow provenance and data provenance: the former is metadata associated with a workflow specifying who, how, what and which resources were used in the workflow, while the latter is complementary metadata describing the derivation flow of data products(Simmhan et al., 2006; Rajbhandari et al., 2005), (3) sharing workflows through publication mechanisms or repositories, for example, publishing workflow on the myExperiment social web site, (4) composing workflow via the drag-and-drop manner on a GUI, and (5) automatic and holistic execution without any external intervenes. In addition, TRIDENT features an environment for creating interactive workflow components and also provides a fine framework for monitoring the execution of the workflow at runtime with extensive error messaging which is quite helpful when in the workflow sequence design phase.

3.2.2 TRIDENT Workflow Engine

While the paper's focus is on the activity libraries for the hydrologic modeling, it is helpful to give a short overview of TRIDENT so as to better understand what the underlying technologies are and based on that what the potential is for future expansion. The TRIDENT workbench is developed on top of the Microsoft Windows Workflow Foundation (WF), a technology for defining, executing and managing workflows. While WF requires .Net programming skill for writing and implementing workflows, TRIDENT provides a simpler and more flexible way to create and run WF data analysis workflows (Microsoft Research Group, 2009). It supports composition, execution, archiving and publishing workflows as well as capturing provenance for each experiment. The TRIDENT package includes two core applications, i.e. TRIDENT Workflow Composer and TRIDENT Management Studio. The former provides a work-board for importing activities (execution blocks within TRIDENT), composing workflows by selecting activities and configuring their relationships, monitoring the progress and performance of workflow execution, and recording workflows for repetitive executions. Activities must be pre-complied into the Dynamic Link Libraries (DLLs). The latter provides a powerful environment for managing workflows, activities, assemblies, users and workflow provenance etc., and moreover, enables users to schedule workflow execution, edit workflows by invoking TRIDENT Workflow Composer. A Trident titled SQL database is created automatically during the installation of TRIDENT. It records all types of data associated with those two applications, for instance, the activities, the workflows and the bonded relationships between a workflow and its activities. Any manipulation on the front will be synchronized to the underneath database. This elaborate design allows TRIDENT applications to connect to a remote server and load its activities and existing workflows.

In addition, TRIDENT involves a light-weight application tool merely used for invoking the execution of a workflow on its connected server, along with a Microsoft Word add-in that allows embedding and executing workflows in a Word document. The TRIDENT architecture is shown in Figure 3.1. Three workflow archiving formats are supported in TRIDENT, i.e. the XML-based file (.xoml), the workflow link (.wfl), and the workflow package following the Open Package Convention (.twp). It supports running workflows on remote computers, as well as running multiple workflows concurrently on different nodes of a Windows HPC server 2008 cluster.

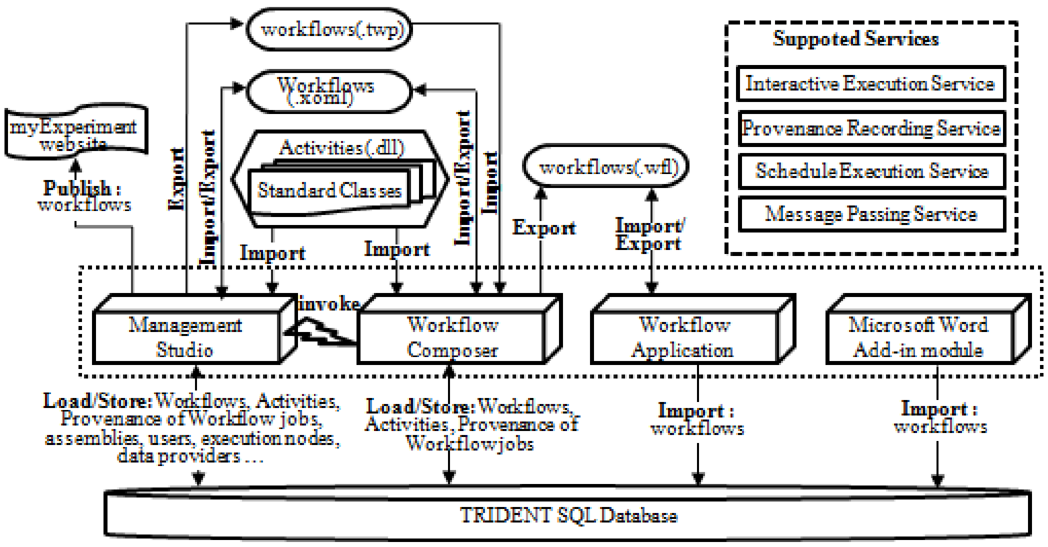


Figure 3.1 The architecture of the TRIDENT workflow

3.3 Architecture of the Modeling Environment

In a simplistic way using a workflow engine for modeling purposes comes down to create a set of general purpose and specific activities, i.e. those units that are responsible for actually doing the work in a workflow. Consequently, some thought needs to be spent on developing an architecture and building activity libraries that are complementary and support each other, especially when trying to sequence them into a workflow. What we mean by this is that it would not be very helpful if activities would be created that have non-matching syntactic and semantic connectors, just like trying to use an US electrical plug in a European socket: neither the plug works (likened to a syntactic problem) nor does the same current flows through the appliances (likened to a semantic issue).

The hydrologic community modeling system encompasses libraries of remote/local data retrieval, data processing, hydrologic computations, and model post-analysis. A variety of workflows conducting hydrologic simulations can be configured in this environment via linking up activities from corresponding libraries. These activities are the execution modules of the workflow environment and typically feature data I/O "plugs" that need to be connected to the next activity in line. This paper presents our current efforts on developing these libraries and how they interact with the workflow engine as depicted Figure 3.2. In addition to having standard activities such as I/O for files and databases, the data retrieval library features activities that can access important hydrologic data sources such as from the Consortium of Universities for the Advancement of Hydrologic Sciences, Inc., CUAHSI (http://www.cuahsi.org), Hydrologic Information System (HIS) water information server, meteorological data from North American Land Data Assimilation System (NLDAS-2, http://ldas.gsfc.nasa.gov), and multi-sensor precipitation estimates (MPE) data from the National Weather Service (NWS, http://water.weather.gov/precip/). The data processing library also takes on the task of syntactically and semantically mediating the heterogeneities between the retrieved data and the data sets controlling the forcing of the model computations. It also contains activities for manipulating geospatial and temporal data respectively, for instance, watershed delineation, unit conversion, and temporal and spatial data interpolations. The hydrologic computation library is the modeling kernel, which provides a batch of activities for simulating single hydrologic processes along with a migrated semi-distributed hydrologic model such as TOPMODEL. In constructing the hydrologic model library the question arises whether to recode legacy code (for example written in FORTRAN) into C language or in any of the .NET languages such as Visual Basic or C#, or to keep the code as is and to embed it into a wrapper that mimics connectivity to the workflow engine programming environment which is Visual Studio (more on this in section 3.4.3).

Finally the post-analysis library is designed for tasks such as uncertainty analysis and model performance evaluations but could also contains unit conversions for desired outputs as well as formatting tasks so results can be stored in custom formats. The library, while still under construction, contains some activities that address performance evaluations such as comparison of computed and observed discharge or water level sequences in terms of hydrograph shape, peak time and amount, and time lag or shifts. We also have included performance objective functions such as the sum of squared errors, the sum of absolute errors, and the Nash and Sutcliffe factor to name just a few. For a complete listing of the developed activities please refer to Appendix A.

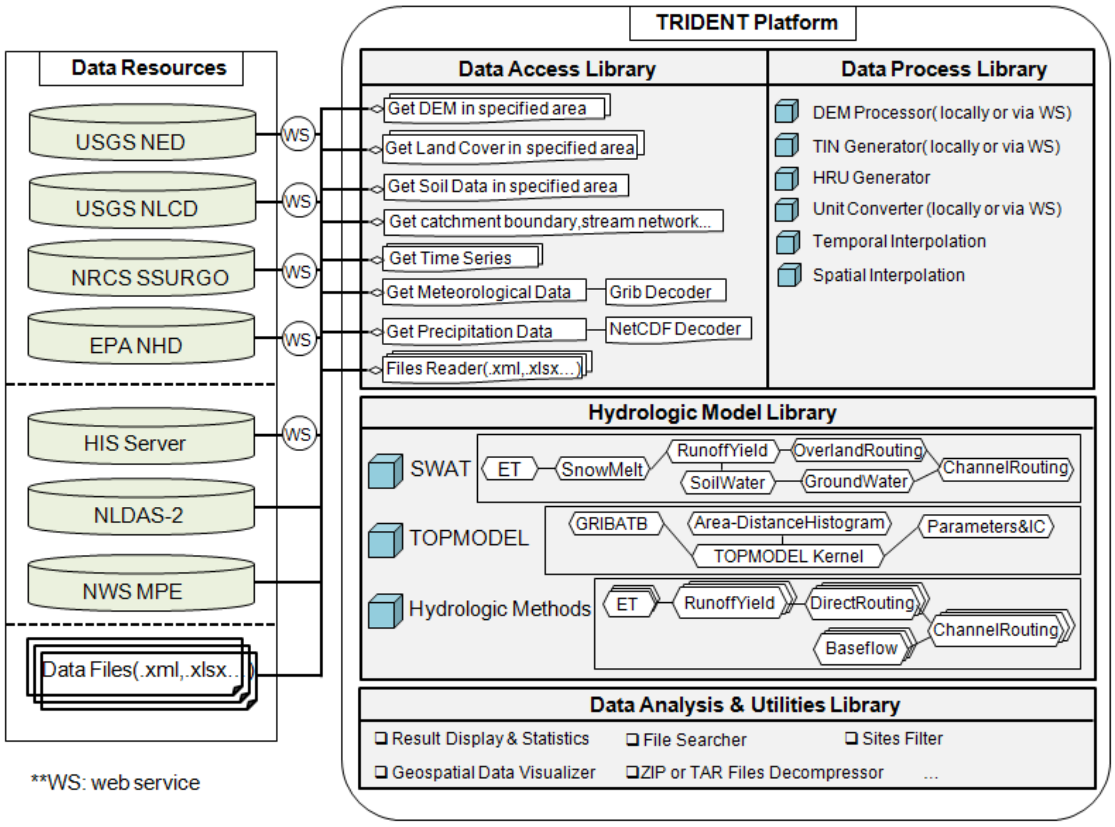


Figure 3.2 Architecture of the Hydrologic Community Modeling System

3.4 Building Modeling Libraries

3.4.1 Data Access Library

We will first focus on the details of our data access library which is basically a library of custom designed activities (in addition to a few standard ones already available as part of the TRIDENT installation) that target specific data sources with a bandwidth of relevant data needed for hydrologic modeling. Because the objective is to have a modeling environment that is applicable anywhere in the US, we focus on data sources that provide nationwide coverage.

3.4.1.1 WaterOneFlow web services

WaterOneFlow stands for a family of web services developed by CUAHSI (for a definition of WOF see http://his.cuahsi.org) using the Simple Object Access Protocol, SOAP. It facilitates retrieving hydrologic and meteorological observation time series data from a central metadata catalogue (HISCentral located at the San Diego Supercomputer Center) which currently (April 2011) holds metadata information for about 5.7 billion data points making it the largest in the world for water data. Currently about 70 data services have been registered in HISCentral reaching from small local data collection efforts, to regional information systems, to large experimental sites operated by governmental institutions, to large nationwide mission agencies such the US Geological Survey (USGS) National Water Information System (NWIS), the Environmental Protection Agency (EPA) STORET data store, and some of the National Climatic Data Center holdings. It also includes smaller services that are nationwide such as DayMET, in addition to NASA remote sensing data derived from NASA'S Moderate Resolution Imaging Spectroradiometer (MODIS) remote sensing data sets, output from the National Centers for Environmental Prediction North American Mesoscale model and the Daymet meteorological model, along with data sets provided by a number of academic entities covering specific areas (Whiteaker and To, 2008).

WaterOneFlow acts as a bridge that mediates communications between data servers and data consumers. Given the URI of the web services either on local servers or those of the HISCentral (for the central metadata catalogue) , a spatial bounding box as well as a temporal bracketthe client can interrogate the underlying database to access metadata of the site and the variable by sending request of "*GetSiteInfo*" and "*GetVariableInfo*" respectively. Once desirable data sets have been identified a time series of a specific variable can be retrieved by sending a "*GetValues*" request when giving a variable code, a site code for a given temporal bracket. These web services then return a data package consisting of metadata and the data itself which is described using the Water Markup Language (WaterML) which is a described XML schema.

TRIDENT offers up an empty shell activity that can connect to web services, which is a valuable feature. However, just connecting to a web service is not enough as WOF require bounding boxes and also some other information without which they will not work. Hence, the workflow sequence designed for use with WOF needs to have some mechanisms that feed the basic web service activity with the necessary attributes required for the call. We subdivide the whole procedure into several sub-tasks, including 1) querying the URLs or IDs of web services capable of providing required data (for a given bounding box this can be many), 2) searching and acquiring the information of sites and variables such as site codes and variable codes for the next-step data request, 3) retrieving and parsing time series. These sub-tasks are carried out by three activities. Herein, the HIS Central metadata web services (HISCentral) designed for accessing metadata of WaterOneFlow web services, sites and variables, are utilized in the development of the first two activities. Providing a latitude/longitude bounding box or a geospatial data file (containing a polygon, for example watershed boundaries), the first activity extracts the maximum and minimum latitude/longitude, then sends a request to the HISCentral, and subsequently collects information from all those web services that have stations within the specified area. The second activity with given information about the URLs or IDs of web services, a geospatial area and a variable name, then parses the variable name, retrieves information of associated variables, requests for making selections among those variables, and finally retrieves information of available sites that have the required data holdings.

One of the key issues in this activity concerns the semantic mapping of variable names to unique parameter codes that are unique to the workflow sequence. In other words, it is necessary to map and associate arbitrary variable names (and codes) to a master set of variable names (actually codes) that will remain unique throughout the workflow process. In this case we use the vocabulary of HISCentral as our “master” accepting the fact that this requires user interaction with the activity. More specifically, we require the user to map the variable name from the data source requested to the underlying master (which can be retrieved from HIS Central via a separate web service; for example a variable “temp” would need to be associated with either “air_temperature” or sol_temperature” or water_temperature”). With the semantics cleaned up, a set of available sites, and the desired temporal bracket, the third activity invokes the WOF “GetVAlues” service. The returned WaterML file is then parsed for the needed information, which is then re-formatted into a format that is understood by the next activity in the workflow sequence. Figure 3.3 shows the sample workflow sequence configured on the canvas provided by TRIDENT.

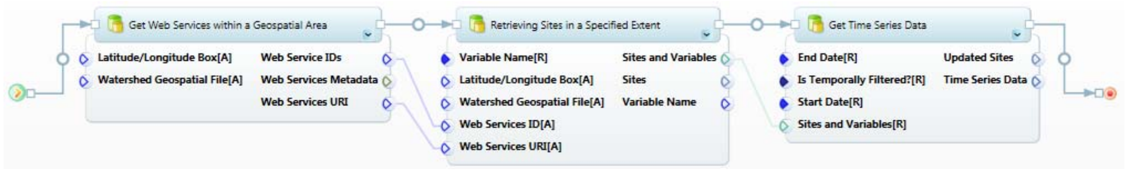


Figure 3.3 A workflow of retrieving data via WaterOneFlow web services

3.4.1.2 North American Land Data Assimilation System (NLDAS-2)

NLDAS-2 provides quality-controlled, spatially and temporally consistent, near real-time land-surface model (LSM) datasets (http://ldas.gsfc.nasa.gov/nldas/NLDAS2forcing.php) for the entire US. The dataset contains eleven fields that are quite useful for land/hydrologic modeling and include precipitation, air temperature, potential evaporation, wind velocity among others. Those forcing fields except precipitation are derived from that of the National Center for Environmental Prediction, NCEP, North American Regional Reanalysis (NARR) data, using a set of temporal disaggregation, spatial decomposition and vertical adjustment algorithms (Cosgrove et al., 2003). The precipitation field is derived from the gauge-based daily precipitation data product at the Climate Prediction Center (CPC) (Daly et al., 1994). The data are 1/8th-degree spatial resolution over the continental U.S. domain, hourly frequency, and range from Jan 1st 1979 to present (Mitchell et al., 2004). Gridded data of all fields are stored in a single GRIdded Binary (GRIB) formatted file for each hour. Those GRIB files are archived and can be accessed via FTP protocol.

In order to retrieve data of certain fields within a given temporal and spatial bracket, three activities are created that function as the downloading sequence for hourly GRIB files from the FTP site, as shown in Figure 3.4. The first activity, having specified a start time and an end time, generates file names according to the required naming convention for the GRIB file download site, and then proceeds to download. The second activity contains a modified "degribber" program, WGRIB, developed for manipulating, inventorying and decoding GRIB files (http://www.cpc.ncep.noaa.gov/products/wesley/wgrib.html). A table of the eleven variables is then presented to the user for selecting the desired parameter fields. Gridded data of selected fields can then be extracted from each file by invoking the WGRIB program with scripted commands. The third and final activity then clips the requested data sets from the national file using the spatial extent definition provided earlier, and then exports them as standardized time series.

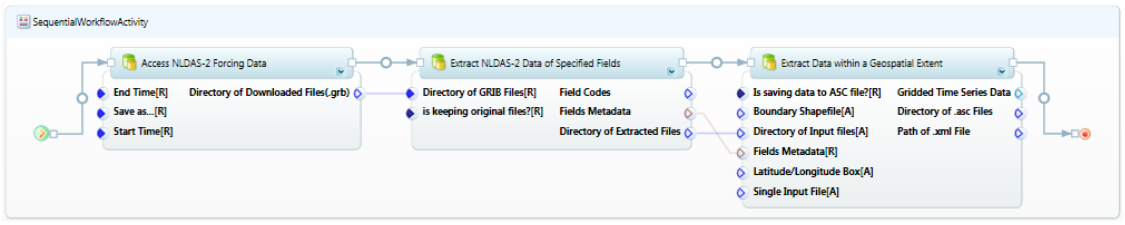

SequentialWorkflowActivity
Access NLDAS-2 Forcing Data
End Time[R]
Save as...[R]
Start Time[R]
Directory of Downloaded Files(.grb)
Extract NLDAS-2 Data of Specified Fields
Directory of GRIB Files[R]
is keeping original files?[R]
Field Codes
Fields Metadata
Directory of Extracted Files
Extract Data within a Geospatial Extent
Is saving data to ASC file?[R]
Boundary Shapefile[A]
Directory of Input files[A]
Fields Metadata[R]
Latitude/Longitude Box[A]
Single Input File[A]
Gridded Time Series Data
Directory of .asc Files
Path of .xml File

Figure 3.4 A workflow of retrieving NLDAS-2 data

3.4.1.3 Multi-sensor Precipitation Estimates (MPE)

Multi-sensor precipitation estimates are developed by the National Weather Service (NWS) operations at the 12 CONUS River Forest Centers (RFCs). For most continental regions, the precipitation datasets are produced using radar estimates from the WSR-88D NEXRAD system or from satellite precipitation estimates with gauge-based observational data for correction and ground truthing. For mountainous areas, MPE are produced by combining long term climatologic precipitation with gauge data. Multi-sensor data have proven to be more accurate than those produced from a single source (Seo,1999; Seo and Breidenbach, 2002). MPE data is delivered using daily averages (there is a new hourly product that will eventually be available across the entire US) ranging from Jan 1st 2005 to present with a spatial resolution of roughly 4km*4km (Hydrologic Rainfall Analysis Project or HRAP grid) and can be accessed via FTP. The files are recorded in NetCDF formatted files, and compressed in tar archives.

We have packaged MPE data access in three activities similarly to those presented before and as shown in Figure 3.5. The first activity is designed to download data files within the specified temporal scale from the NWS MPE download site. An additional decompressing procedure is embedded in this activity for releasing archived NetCDF files. The second activity parses the NetCDF files, and exports precipitation datasets in the format of standard arrays. The last activity extracts data within a latitude/longitude box or a watershed boundary from the data arrays, and organizes them into standardized time series. To this end we have taken the netCDF libraries and embedded them into our workflows, so any netCDF file can be accessed.

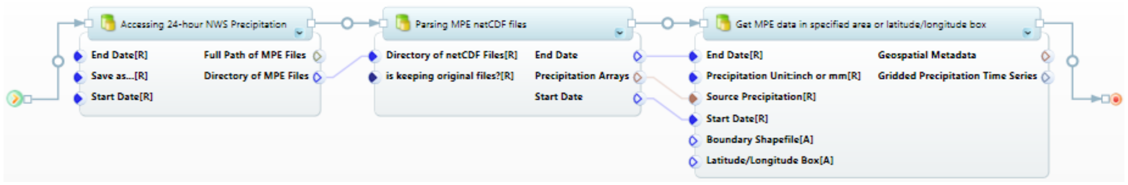

Accessing 24-hour NWS Precipitation
End Date[R]
Save as...[R]
Start Date[R]
Full Path of MPE Files
Directory of MPE Files
Parsing MPE netCDF files
Directory of netCDF Files[R]
is keeping original files?[R]
End Date
Precipitation Arrays
Start Date
Get MPE data in specified area or latitude/longitude box
End Date[R]
Precipitation Unit:inch or mm[R]
Source Precipitation[R]
Start Date[R]
Boundary Shapefile[A]
Latitude/Longitude Box[A]
Geospatial Metadata
Gridded Precipitation Time Series

Figure 3.5 A workflow of retrieving MPE data

3.4.1.4 USGS National Elevation Dataset (NED)

The National Elevation Dataset is the latest elevation data product assembled and distributed by the USGS. It is designed to provide seamless raster elevation data of the conterminous United States, Alaska, Hawaii and the island territories using a consistent datum, coordinate system, projection and elevation unit (Gesch et al., 2002). It provides national coverage at a grid spacing of 1 arc-second (approximately 30 meters) and a post spacing of 1/3 arc-second (approximately 10 meters), with the exception of Alaska that only has a resolution of two arc-second. In some areas, elevation data of a higher resolution of 1/9 arc-second are accessible. For all three NED layers, the horizontal datum is the North American Datum of 1983 (NAD83), the vertical datum is the North

American Vertical Datum of 1988 (NAVD 88) and the projection is a geographic coordinate system (decimal degrees of latitude and longitude). The elevation units are standardized to decimal meters.

The USGS Earth Resources Observation and Science (EROS) center has provided a set of web services that can be incorporated into custom applications for exploring NED data and relevant metadata. The core web services encompass the Inventory Service that can retrieve dataset information for a desired area of interest, the Request Validation Service that verifies and validates the returned information from the Inventory Service and composes URL(s) for downloading, and the Download Service that initiates download jobs and returns data to users (USGS EROS, 2010). We have developed an activity that supports downloading NED data of a specified resolution within a specified area making use of the USGS EROS web services . It allows users to select an archiving type either zip or tar-zip, a file format, i.e. ArcGrid, or GeoTIFF, or GridFloat or BIL, along with a resolution either 1, or 1/3 or 1/9 arc-second, and stores the downloaded data in a default folder. This activity can then be connected with the decompress activity that first searches the archived files for the ones needed and then decompresses them file by file. The workflow sequence is shown in Figure 3.6.

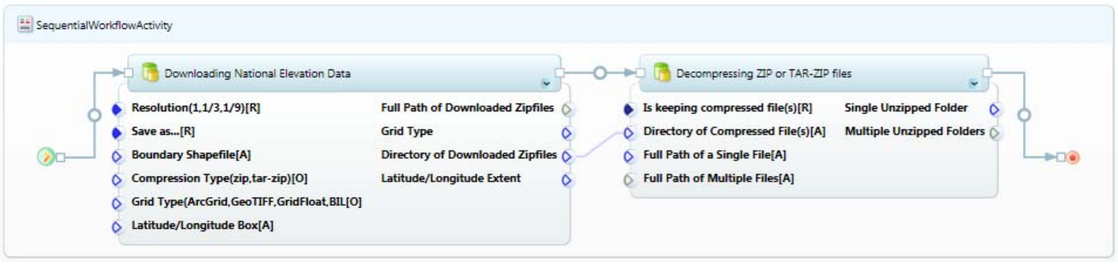

SequentialWorkflowActivity
Downloading National Elevation Data
Resolution(1,1/3,1/9)[R]
Save as...[R]
Boundary Shapefile[A]
Compression Type(zip,tar-zip)[O]
Grid Type(ArcGrid,GeoTIFF,GridFloat,BIL[O]
Latitude/Longitude Box[A]
Full Path of Downloaded Zipfiles
Grid Type
Directory of Downloaded Zipfiles
Latitude/Longitude Extent
Decompressing ZIP or TAR-ZIP files
Is keeping compressed file(s)[R]
Directory of Compressed File(s)[A]
Full Path of a Single File[A]
Full Path of Multiple Files[A]
Single Unzipped Folder
Multiple Unzipped Folders

Figure 3.6 A workflow of retrieving DEM data

3.4.1.5 National Land Cover Data (NLCD)

The National Land Cover Data is provided by the Multi-Resolution Land Characteristics Consortium (MRLC). The first generation of land cover database (National Land Cover Dataset 1992) was derived from the Landsat Thematic Mapper™ (Vogelmann et al., 2001). It only covers the conterminous United States, while the next-generation National Land Cover Database 2001 derived from Landsat imagery and ancillary data aims to cover Puerto Rico, Hawaii, Alaska as well (Homer et al., 2004). NLCD 2001 defines 29 classes of land cover in total, in which nine classes are only for coastal zones, four classes are unique for Alaska, and the rests are for the main continent. Two data products recording the per-pixel estimates of the percentage of imperviousness and tree canopy are generated additionally. It has a spatial resolution of 30 meters, and uses the same horizontal datum as the NED, but a different projection, i.e. Albers Equal Area Conic projection.

The process of retrieving NLCD data resembles that of retrieving NED data. Two activities are involved to address this task. The first one is responsible for retrieving land cover data in a specified area, i.e. a watershed boundary or a latitude/longitude bounding box, and the other decompresses the downloaded archives (which are encoded in GeoTIFF). Given a desired geospatial extent the activity first collects available land cover sources by tapping into the Inventory Service. Users can select single or multiple sources, which normally include NLCD 2001, NLCD 2001-Imperviousness, NLCD 2001-Canopy and NLCD 1992. For each selected source, a request containing its attached URL can be composed and then submitted to the Validation Service. When accessing the URL, a download ID will be assigned with which specified land cover data can be downloaded via the Download Service. The downloaded archives can then be decompressed and placed into designated folders, just as for the NED activity.

3.4.1.6 Soil Survey Geographic Database (SSURGO)

The U.S. Department of Agriculture's (USDA) Natural Resources Conservation Service NRCS) publishes three soil databases intended to support resource planning and management at different scales: the Soil Survey Geographic database (SSURGO), the State Soil Geographic database (STATSGO), and the National Soil Geographic database(NATSCO). The SSURGO database provides the most comprehensive soil information, while substantial spatial differences have been reported between STATSGO and SSURGO data especially for small areas and river watersheds (Juracek and Wolock, 2002; Lathrop et al., 1995), prompting the need for careful evaluation which database to use for what purpose. SSURGO is comprised of a tabular component containing information of soil attributes and a spatial component that can be visualized and analyzed via a Geographic Information System (GIS). The latter represents soil patterns in a landscape, and normally involves six spatial entities delineating the survey area boundary polygons, the line features, the point spot features, the map unit boundary polygons, the line map units, the point map units as well as a spot feature description entity. The map unit boundary polygon features along with the associated tabular soil attributes are of more importance in the parameterization of soil component for hydrologic modeling.

The NRCS provides a couple of tools for accessing soil data, of which the Soil Data Mart (SDM) tool is of the most commonly used one. However, the SDM only supports delivery of spatially distributed soil data in tabular form for specified counties or states, which is not user friendly and cumbersome to interpret. The information is distributed in ASCII delimited files, and required to be imported into a database just so it can be visualized let alone being parsed. To compensate for this, the Soil Data Access (SDA) system offers up a suite of web services that enable the request of geospatial data with tabular attributes for any area. Even though not all soil attributes can be accessed via submitting a single request, the embedded soil attributes in geospatial soil files are sufficient for common usages. Otherwise more abundant soil information can be obtained by sending SQL queries via the tabular web service. Those web services are based on the OGC WMS and WFS standards, and could be customized into TRIDENT activities as well.

Accessing soil data activity requires provision of a latitude/longitude geospatial extent and the coordinate system. Acceptable coordinate systems include the WGS84, the NAD83 and NAD83 UTM. If selecting the NAD83 UTM coordinate system, providing the UTM zone number is optional since the activity can parse the zone number based on the given longitude. The SDA tool provides various layers but only covers map unit point, line and polygon features together with a handful of soil attributes as well as the survey area polygon features. Returned soil spatial data is encoded in Geographic Markup Language, GML, and the activity parses the data and exports them into GIS shapefile format. However, the current SDA only supports requesting data for a relatively small area, thus attempting to access large amounts of soil data would probably fail. This issue is expected to be addressed in the future. Figure 3.7 depicts the procedure of accessing SSURGO spatial data and adapts them to a specified geospatial extent.

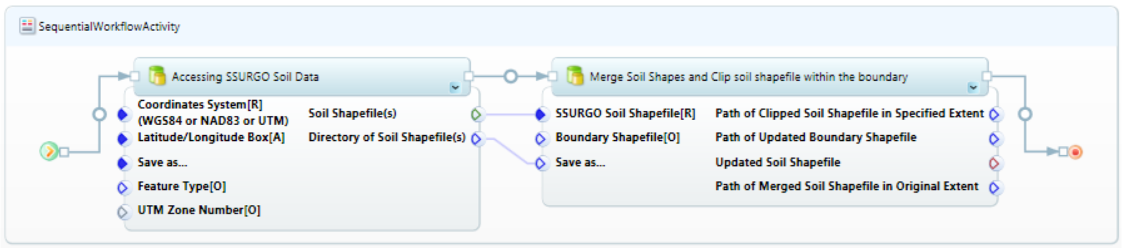

SequentialWorkflowActivity
Accessing SSURGO Soil Data
Coordinates System[R] (WGS84 or NAD83 or UTM)
Latitude/Longitude Box[A]
Save as...
Feature Type[O]
UTM Zone Number[O]
Soil Shapefile(s)
Directory of Soil Shapefile(s)
Merge Soil Shapes and Clip soil shapefile within the boundary
SSURGO Soil Shapefile[R]
Boundary Shapefile[O]
Save as...
Path of Clipped Soil Shapefile in Specified Extent
Path of Updated Boundary Shapefile
Updated Soil Shapefile
Path of Merged Soil Shapefile in Original Extent

Figure 3.7 A workflow of accessing soil data

3.4.2 Data Process Library

While data processing is one of the most time-consuming steps in a modeling effort and often the processing steps are repeatable tasks that lend themselves to be incorporated into a workflow sequence. Hence, in this subsection we introduce a number of workflow activities that open and establish connections to data centers where data sets pertaining to hydrologic modeling can be obtained. The examples given are by no means all encompassing, there are many more sources, but they serve as good examples what it takes to establish these data processing activities.

3.4.2.1 Geospatial Data Processing

Watershed delineation or decomposition is normally considered a preliminary task for hydrologic modeling. Most geographic information system(GIS) software tools provide functionalities for geospatial data processing, including capturing watershed characteristics, tracing flow directions, systematizing channel networks and delineating watershed boundaries (Beven and Morre, 1992; Wilson and Gallant, 2000; Maidment, 2002). Accordingly, we have developed a suite of activities carrying out procedures of DEM processing for watershed delineation that are based on procedures deployed in the Terrain Analysis Using Digital Elevation Models, TauDEM (Tarboton et al., 1991), and PIHMgis (Bhatt et al., 2008).

The step-by-step procedure of DEM processing is illustrated in Figure 3.8. We have dedicated one activity for each of the steps, with one “master” activity that implements the whole processing procedure. Aside from the DEM, the external inputs in this workflow include a threshold value of drainage area and an ESRI shapefile locating an outlet. For each grid, the number of upstream grids that drain to it is calculated. If the accumulated value is greater than the specified threshold value, the grid is defined as stream. The smaller the threshold is, the more complicated the river system will be, and the more sub-basins will be generated. While the outlet is normally located at a stream gauge, in some circumstances, the chosen stream gauge is not located on the defined stream network. We have thus developed an activity to address this issue. With given flow contribution data obtained from the third step of the DEM processing and the sites shapefile that could be obtained via calling WOF web services, the activity searches for the stream grid closest to the site, and then automatically creates a new shapefile containing that point. If multiple sites are present in the WOF downloaded shapefile, the one furthest downstream will be used as the target site. The outlet shapefile is not a mandatory input however, and without it the delineated exterior watershed boundary is identical to the periphery of DEM.

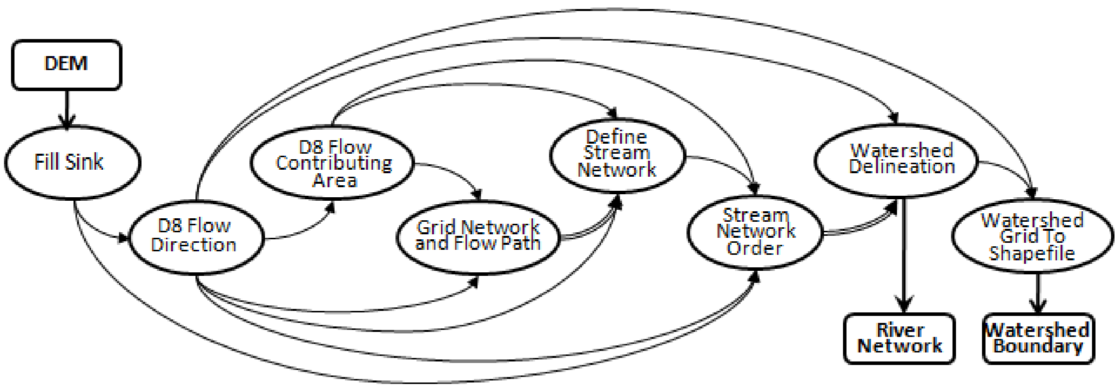


Figure 3.8 The step-by-step procedure of DEM processing

The process of triangulation aims at generating a triangulated irregular network (TIN) to describe the surface morphology (Tsai, 1993). The mesh generation is conducted based on Delaunay algorithms (Delaunay, 1934; Tucker, 2001), with watershed boundary, river network and other physiographic coverage as restrictive layers. A series of spatial analyses programs for converting, smoothing, splitting, merging watershed polygons and stream polylines have been selected and wrapped into activities. Those activities assist in preparing a single GIS layer that incorporates all geometric and topographic information of constraining layers, thus enhancing the quality and improving the efficiency for domain decomposition. Aside from constraining layers, a smoothing tolerance is required, which is responsible for simplifying the polylines by eliminating their fluctuations or extraneous bends while still preserving essential features (Bhatt et al., 2008).

We have also exposed some of the processing functions as web services which can be accessed via the internet by anyone seeking to process DEM information and needing to generate a TIN. These web services implement the OpenGIS® Web Processing Service (WPS) standard specifications, which exposes three standard methods, i.e. "*GetCapabilities*", "*DescribProcesses*" and "*Execute*". The first two methods are responsible for returning service-level and method-level metadata respectively while the last method launches the actual geospatial data calculation (Schut, 2007). We have developed two activities that can call and consume the responses of these web services, one that calls and invokes DEM processing and one that generates a TIN as shown in Figure 3.9. Note that even though these two activities basically do the same as those described earlier, the code assembly is quite different with the latter open for public access and also potentially being faster depending on the server they reside on and the available internet transmission speeds.

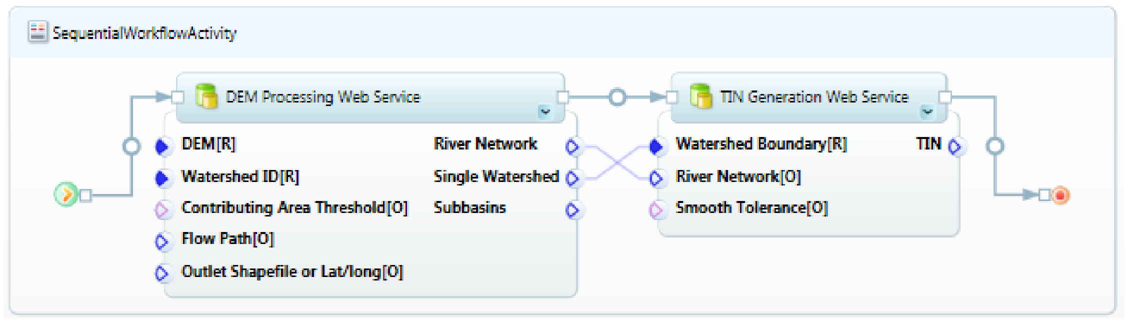

SequentialWorkflowActivity
DEM Processing Web Service
DEM[R]
Watershed ID[R]
Contributing Area Threshold[O]
Flow Path[O]
Outlet Shapefile or Lat/long[O]
River Network
Single Watershed
Subbasins
TIN Generation Web Service
Watershed Boundary[R]
River Network[O]
Smooth Tolerance[O]
TIN

Figure 3.9 The workflow accessing web services for DEM processing and triangulation

3.4.2.2 Generating Hydrologic Response Unit (HRU)

The concept of creating HRUs is based on the idea implemented in SWAT (see also http://swatmodel.tamu.edu/media/19754/swat-io-2009.pdf). The idea is to partition a watershed into a number of sub-basins based on flow attributes and then to sub-divide each sub-basin into HRUs as a function of soil properties and land use patterns. The HRUs are assumed to have homogeneous hydrologic responses, thus permitting a higher degree of efficacy and efficiency by limiting the computational units of the study area rather than carrying out computations over every available land use or soil type. Constructing HRUs for a watershed requires soil type and land cover data, which can be obtained from the National Resources Conservation Services, NRCS, and USGS respectively. Consequently, we have developed two activities that access USGS and NRCS, for extracting original classification information and performing reclassifications if necessary.

The first activity, processing land cover data, requires a number of preliminary steps. Because USGS uses the Albers Equal Area Conic projection, it needs to re-projected onto a geographic projection using latitude/longitude referencing as this is the base for the entire modeling effort. It also needs to clip the grid file to cover the desired extent, in case a watershed polygon is provided. The final reclassifying step is an optional one. Aside from classes for coastal zones and Alaska, there are still fifteen to be chose from. Because the HRU computation can be time consuming, we decided to introduce the possibility of defining a simpler classification system which narrows the original selection down to four, i.e. water, medium residential, residential, and agricultural. These new classification are described in an XML file that is embedded into the activity and can be updated easily. The activity will create a new grid file and replaces the fifteen original classes with the new ones which can then be used directly for building HRUs. However, since this step is not mandatory, the user can also use the original fifteen classifications for constructing the HRUs .

When accessing SSURGO spatial data one essential attribute is the feature symbol which logically identifies the corresponding survey area or map unit. A symbol can be considered as a specific soil combo type with different slopes. Since there might be hundreds of different symbols for a fairly small area, it would be quite difficult if one were to consider all of them in a modeling effort. It is easier to use NRCS classifications of those soil types into four major hydrologic groups, denoted as A,B,C,D together with the combinations of two of the hydrologic groups such as B/D, B/C and so on. The categorization of multiple soil types into specific hydrologic groups has the benefit of capturing characteristics of different soils on some level, and on the other hand can reduce the degree of complexity of using soil data.

Two key operations are conducted over the SSURGO spatial data file before creating HRUs. Since the original extent of a spatial data file is normally specified as a latitude/longitude box, extracting spatial information within a given watershed boundary is the first task. Moreover, a soil spatial file, for example, a map unit polygon file, contains a large number of small shapes identified as map unit symbols. Hence, we first introduce a reduction step in which we merge shapes with same identifiers for each hydrologic group, i.e. all those that belong to the same hydrologic group. The above computations are implemented in a dedicated activity, which takes the boundary shapefile and SSURGO spatial soil file as inputs, and exports the updated soil spatial file with the soil shapes re-delineated. If the given boundary shapefile contains more than one polygon or polyline feature, a single shape covering the entire area is created and exported as an updated boundary shapefile.

Constructing HRUs is embedded in another dedicated activity, which apart from the land cover and soil data, needs the sub-basin grid data as a key input. The sub-basin grid is composed of a number of cells, each of which is denoted by either a sub-basin identification number or the no-data identification number. The activity analyzes each cell of the sub-basin grid, and requests the land cover and soil type that belong to it. Cells with the same land cover and soil type will be accumulated into larger regions. Therefore, for each sub-basin, a variety of land cover and soil combinations can be generated. It is possible that some combinations only contain a few cells, in which case it is not economical to treat them separately as a distinct HRU. Hence, a threshold value described as the percentage of total area can be set in advance which will only allow those HRUs to exist that are either equal or larger than this threshold value. The accumulated areas of the discarded combinations will be evenly distributed onto the

HRUs. If none of the combinations meets the threshold requirement, the one taking up the largest area will be set as the only HRU for that sub-basin.

3.4.2.3 Time Series Data Process

Time series data accessed from diverse repositories are different in terms of unit, data type, temporal and spatial resolution. For example, HIS servers mainly return gauge-based time series with varied data types and units, while NLDAS-2 and MPE provide gridded data of different spatial resolutions. In most circumstances, the data formats do not meet model input requirements, thus customized activities have been developed for each individual data source. Data differs in its syntactic and semantic description in addition to not fitting the geospatial locations required, for example on grid points (or cell centers wherever the variable is described in the model) in addition to having different time stamps that need to be adjusted. We have therefore developed a number of activities that perform temporal and spatial interpolations (from “where it is” to “where it needs to be” in the spatio-temporal domain) in addition to auxiliary functions such as unit conversions.

Unit conversion has been implemented as a web service (this way it is accessible on the internet), which uses unit definitions and conversion factors described in an XML file. We have defined a default unit for each base unit such as length, mass, and time together with conversion factors for any other derived unit. For example, the default unit of length is set as “meter” and when trying to convert “foot” to “inch”, “foot” is first converted to “meter” by carrying out the calculation specified in the “Conversion” attribute of the “foot” field, i.e. multiplying by 0.3048, and then the converted value implements the opposite calculation algorithm specified in the “Conversion” attribute of “inch”, i.e. dividing by 0.0254. Since the units are recognized via their names, a set of possible names or abbreviations for each unit is specified in its “Alias” attribute. We have also implemented the feature that the user can specify the source unit, desired target unit, the variable name and a data array, and after invoking the web service to getting back a converted data array.

As an alternative option we have created a workflow activity (as opposed to a web service that is called) that also includes an automatic search procedure in which desired variables are matched to possible units and where the user can pick a preferred target unit for the output variable in question. For temporal interpolation of data points we have adopted linear and polynomial algorithms while the inverse distance weighted method is utilized for distributing or “moving” data from available gauges to sub-basins. Figure 3.10 shows a partial workflow of processing data accessed via WaterOneFlow web service.

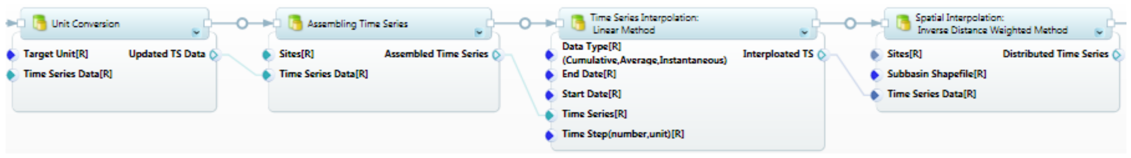


Figure 3.10 A partial workflow for time series processing

Commonly needed data operations for NLDAS and MPE gridded data involve temporal and spatial aggregations for which we have created a dedicated activity (note that this activity is customized for NLDAS and MPE data, which removes a little bit from the objective of a general purpose activity, but -it can be further expanded to handle additional data sources). The eleven fields in a NLDAS data set reflect variables that need either temporal, or spatial, or modifications in both spaces, for example “temperature” needs to be averaged while “precipitation” should be accumulated over a given temporal scale. Also, spatial operations often necessitate the aggregation of gridded data over sub-basins. Our activity groups the grids belonging to each sub-basin, and then averages grid data of each group at each time step .MPE is similar but does not involve separation of variables because it only has one data product, i.e. rainfall.

Because NLDAS requires downloading GRIB files containing hourly data, thus demanding a considerable amount of disk storage to be available when long simulations are required, we created an all-in-one activity that deletes the GRIB files after they have been processed for data extraction. This activity implements the sequence of accessing a GRIB file, parsing gridded data, deleting the file, and computing sub-basin-based data at each time step. While it is somewhat slower than the step by step workflow sequence of individual activities we had good experiences reducing the storage requirement whenever a long simulation time was required.

3.4.3 Hydrologic Modeling Library

There are many ways and options to execute a hydrologic model task featuring a wide range of simple peak runoff calculations, to parameterized models, to fully distributed models of many shades and flavors. While it is possible to include and add any number of modeling approaches (this could be a back bone for a community modeling system), it would be impossible to include all of them into this modeling framework at the initial state of development. Hence, we have targeted a small subset of modeling approaches to demonstrate how each type or approach can be implemented.

3.4.3.1 TOPography based hydrologic MODEL, TOPMODEL

TOPMODEL is a physically based watershed model that simulates hydrologic fluxes including infiltration-excess overland flow, saturation overland flow, infiltration/exfiltration, subsurface flow, evapotranspiration and channel routing through a watershed (Beven and Kirkby, 1979) and has been subject to a number of applications and improvement efforts, see for example Saulnier et al., 1998; Quinn,1998; Myrabø, 1997. A distinctive concept of TOPMODEL is to use a so-called topographic index to reflect variable characteristics of soil and topography over a watershed. A study area need be decomposed into a number of units with categorized topographic indices, which are analogous to the HRUs described earlier in that they represent an integrated area value, while in reality they may distribute across the sub-basins. Flow routing over a sub-basin/basin is computed using the Contributing Area method.

There are a number of TOPMODEL versions, from which we have selected the Visual Basic kernel for our development. This version is most suitable for humid or semi-humid watersheds with shallow soil coverage and moderate topography in addition to being written in language that is easily incorporated into the Visual Studio environment (which we use for our TRIDENT activity development). We have integrated TOPMODEL as a sequence of four activities. The first activity is a migration of the GRIDATB program which handles the computation of the topographic index and the classification of topographic units. Given a study area, it is optional to decompose the study area into sub-basins and then produce topographic units over each sub-basin or to perform the topographic computation over the whole basin directly. The second activity ingests geospatial data derived from the DEM processing steps mentioned earlier and produces the area-distance histogram for routing overland flow. The third activity is responsible for initializing parameters and initial states, which also provides an interactive interface for the purpose of parameter calibration. The last activity encapsulates the kernel of TOPMODEL that performs the actual hydrologic calculations.

3.4.3.2 Soil Water Assessment Tool, SWAT

SWAT is a physically-based, spatially semi-distributed model developed to simulate the effects of management decisions on water, sediment yields and pollutant loadings in watersheds related to soil, land use and management practices (Srinivasan and Arnold, 1994; Arnold et al., 1998). The hydrology component in SWAT is based on the water balance equation in the soil profile and simulates processes including canopy interception, infiltration, surface runoff, evapotranspiration, lateral flow and percolation. As mentioned before SWAT conducts a two-stage decomposition over a watershed, and the final products called HRUs are assumed to have homogeneous characteristics and thus expose conformable hydrologic responses. The responses of each HRU are determined individually and accumulated at the sub-basin level, and then routed through its associated main channel to the sub-basin outlet, which join the channel network for the final routing to the watershed outlet (Bouraoui et al., 2004). SWAT has been hugely popular due to its capability to model complex and coupled watershed processes quite successfully see for example Rosenthal et al., 1995; Peterson and Hamlett, 1998; Muttiah and Wurbs, 2002.

SWAT provides alternative methods for runoff simulation volume such as SCS Curve Number method and Green & Ampt infiltration method, alternative methods for estimation of evapotranspiration such as the Penman-Monteith, Priestley-Taylor along with the Hargreaves method, and Variable Storage and Muskingum methods for channel routing. We have disassembled the SWAT2005 version and embedded the hydrological computing segments in a set of activities. The collection of SWAT-based activities currently involves one activity for each process except evapotranspiration. The estimation of total flow volume and the fractional volume arriving at the main reach is carried out by a single activity regardless of the flow location. For example, the SCS Curve Number method and lag time method are bonded into a single activity to handle surface runoff process. Estimation of potential evapotranspiration is a relatively independent procedure thus each of the three methods is wrapped into an independent activity and also capable of contributing to other model set ups within TRIDENT. Since there are quite a number of parameters required in SWAT, each activity offers an interactive interface for inputting and adjusting these parameters from their default values.

3.4.3.3 General Hydrologic Processes

Aside from taking advantage of existing hydrologic models, a user can also create activity sequences for simulating hydrologic processes using basic methods and link them together as loosely-coupled models. While some of these activities use simplified hydrologic theory they permit a higher degree of flexibility when configuring hydrologic model approaches. Dominating flows in a watershed are direct runoff, base flow and channel flow where slow soil water movement is neglected, and direct runoff is perceived as the combination of flows on the surface and in the unsaturated soil zone. If modeling is carried out at the sub-basin-level, direct runoff is computed on each sub-basin individually and then routed to the outlet of the sub-basin rather than the main channel of the sub-basin via different unit hydrograph methods. In other words, the first-order channels are not taken into account in the channel routing.

In developing these activities we weighed the efforts required of using legacy codes versus coding new model kernels from the ground up. Consequently, some activities are created using legacy codes, while others are programmed directly mostly due to the underlying simplicity of the algorithm used. The currently encoded hydrologic methods are listed in Table 3.1.

Table 3.1 Methods for simulating hydrologic processes

Hydrologic processes	**Methods**
Evapotranspiration	Penman-Monteith method
	Thornthwaite method
	Priestly-Taylor method
	Hargreaves method
Runoff Yield	SCS Curve Number
	Green&Ampt method
Direct Runoff Routing	SCS Unit Hydrograph
	Synder Unit Hydrograph
	Clark Unit Hydrograph
Base Flow	Linear Reservoir
	Recession baseflow
Channel flow routing	Muskingum-Counge method
	Modified Wave method

3.4.3.4 Code Implementation

Our development is focusing on the creation of activities, which are encoded in C# language and compiled into Dynamic Link Libraries (DLLs). Each activity is required to define its input/output variables explicitly via a metadata-tagging approach, along with an “Execute” function that controls its main computations. A workflow host application, e.g. the TRIDENT workflow composer, is thus able to expose the input/output variables of an activity via parsing its properties when it is loaded. The host invokes the execution of an activity via its “Execute” function, and pipes the data flow from the output of one activity to the input of another activity at run time, commonly called “introspection”. A variable can be set as input, output or input/output, and assigned as mandatory or optional. The activity cannot execute if its required inputs were not given.

Some activities have been scripted from the ground up, while others have been created porting legacy codes directly which often poses some challenges because of language heterogeneities. For porting TOPMODEL (written in Visual Basic) we opted for the re-coding approach, because VB is quite similar to C# the preferred language in which the activities are encoded. While this is not without the need for some investment of time, it is relatively straight forward. The migration of the SWAT (written in C) is a different case however. Our objective was to migrate the code stack responsible for each hydrologic process (e.g. evapotransiration, runoff yield, overland routing) into an activity, so that the activities could gain more operational independence and possibly be swapped with other alternatives as is done in SWAT.

SWAT also poses a challenge because it performs the simulation of the entire hydrologic cycle at each time step whereby some of the implemented methods require the interaction of different processes at each time step. For example, in SWAT the Curve Number (CN) method is used for computing surface runoff yield and infiltration. The controlling parameter CN is updated at the end of each time step based on either the evapotranspiration amount or the soil water. If using soil water, the computation sequence would be to compute the infiltration first, then deriving the soil water from the infiltration, and finally updating CN with the soil water for the computation of next time step. However, this type of loop computation is not supported by TRIDENT, as it only allows one-directional data transport between two activities. The solution to this problem is to create an activity that executes the entire temporal loop for each hydrologic process without requiring/implementing interactive methods. Thus, in our example we update CN by using the evapotranspiration amount instead which is decoupled from the data products of curve number method. For inseparable processes such as the simulation of the soil water amount and the partial amount routed to the channel, they are combined into a single activity.

The other approach is to compile the legacy codes into Dynamic Link Libraries (DLLs). An internal interface is embedded in a DLL which mediates the communications between the DLL kernel and those code segments who call it. If the DLL is encoded by a low-level language (e.g. C or Fortran), an additional interface is required at the side of its consumer in order to define entry points of the DLL. Figure 3.11 shows the composition of an activity and its interaction with an external module; our netCDF activity has been created using this approach.

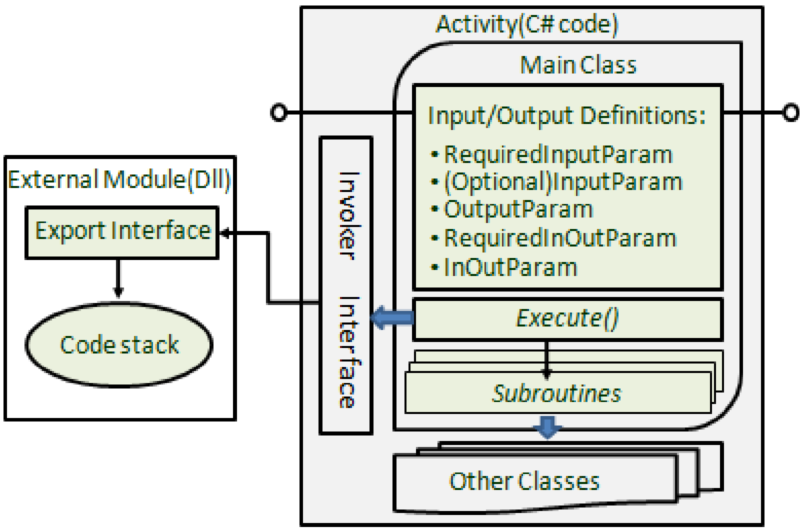


Figure 3.11 Composition of an activity and its interaction with a module

3.4.3.5 Data Model for HCMS

Within a hydrologic modeling workflow, there are two types of data that are communicated from one activity to the next. One comprises representatives of physical computation elements, i.e. sub-basin polygons, river reaches and hydrologic response units, and the other is (for most part) time series data, either a set of gridded data or a data array. In order for two activities to communicate with each other they need to have syntactic and semantic compatible I/O type definitions (e.g. string, double, ect). Since the basic types given by the .NET IDE are not suitable for our study, we standardize the definitions of data types as shown in Figure 3.12.

A watershed is represented by the BASIN class which comprises collections of SUBBASINs and RIVERs along with their number of appearances. Since the current activities require either the sub-basin collection or the river collection, the BASIN class is rarely utilized. The basic elements are SUBBASIN and RIVER, while the SUBBASIN can be further decomposed into HRUNIT. Aside from basic properties, the SUBBASIN specifies the outlet, centroid, and the farthest point to the outlet as a POINT type. The river embedded in a sub-basin is identified by the RiverID. A river can have two upstream river reaches and one downstream river reach. The magnitude and order properties defined in the RIVER type are important in the computation of channel routing. A river having no upstream is defined as the first-level river both in with respect to order and magnitude, while a river having two upstream rivers is defined as the maximum order and the accumulated magnitude of the two. The cross-sectional area of a river is defined by the SHAPE type. The HRUNIT defines the area with a specific combination of soil type and land cover type, which is assumed to have a homogeneous hydrologic response. The identification of soil or land cover types could be a number or a character(s), thus they are defined as object type.

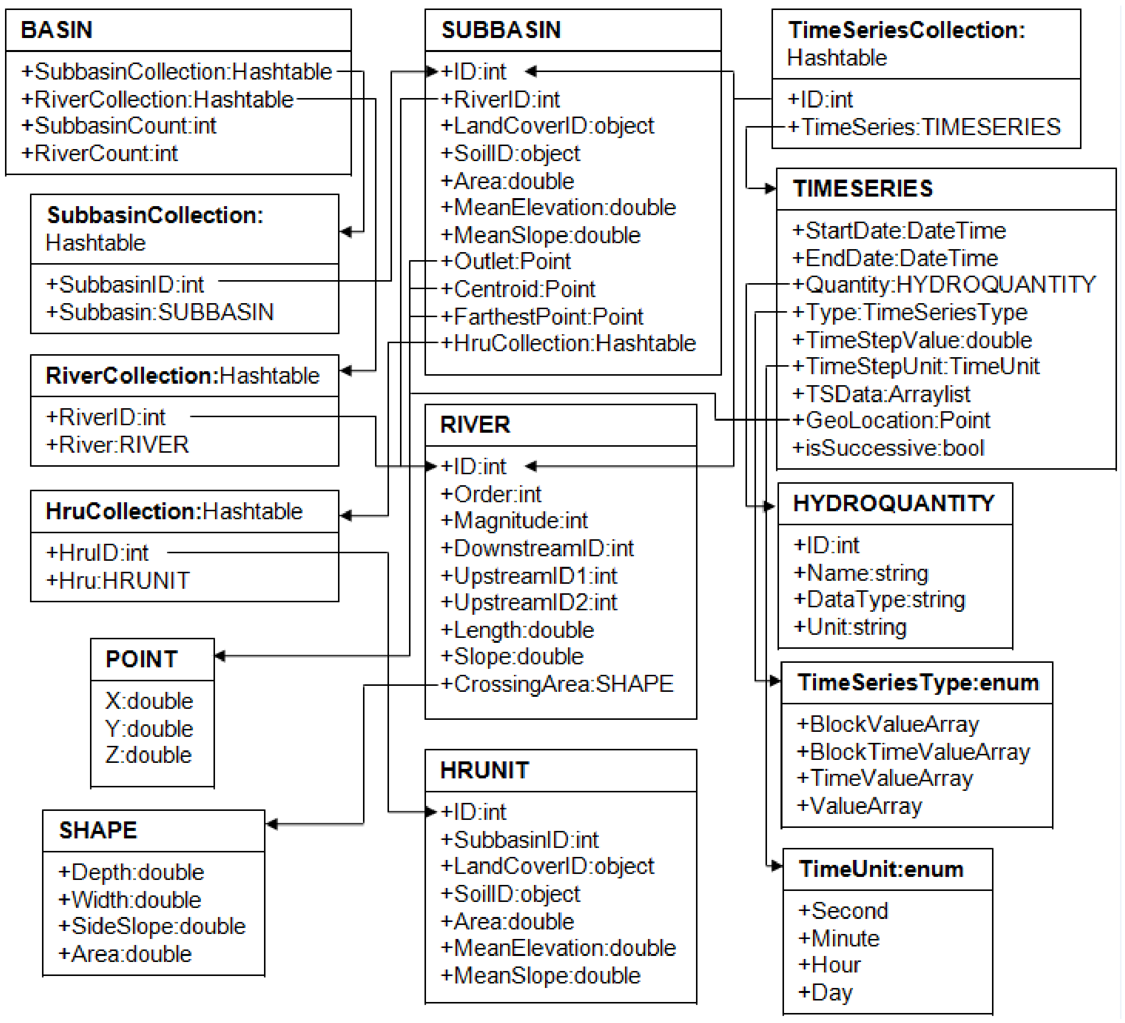


Figure 3.12 The data structure of Hydrologic Community Modeling System

Within the definition of TIMESERIES type, TimeSeriesType is utilized to differentiate the gridded data series (BlockValueArray, BlockTimeValueArray) and the common time series (TimeValueArray, ValueArray). For time series accessed from a site, the location of the site can be specified as POINT. A time series is always linked to a specific quantity (e.g. precipitation, temperature etc.), which is defined as HYDROQUANTITY type. The time series data are usually transferred as a collection of TIMESERIES objects.

3.4.4 Analysis Library

Performance of a hydrologic model can be evaluated using several measures, for example by analysing the “closeness” of the simulated variables (typically streamflow or stage) to observations made within a watershed (Krause et al., 2005). The “closeness” can be qualitatively assessed via visually inspecting the simulated and observed hydrographs, particularly focusing on hydrograph shape, peak time and amount, and time lag or shifts. Often emphasis is placed on a quantitative measure defined as mathematical measures of how well a model simulation matches the observations (Beven, 2001). Hence, we have developed an activity that both displays the simulated and observed hydrographs, and also performs calculations for seven performance measures including the Nash-Sutcliffe coefficient, the coefficient of determination, the index of agreement, the relative efficiency criterion, the root mean square error, the relative mean error and the percentage of peak difference. Table 3.2 lists the equations of the efficiency criteria. The activities that perform calculations of the seven performance measures separately have also been created, which can be linked to the analysis activity directly. In addition, an activity that analyses water balance within a hydrologic simulation is also placed in this library.

This library also includes activities that carry out common tasks. For example, an activity embedding the MapWinGIS ActiveX Control is created to display geospatial data. The MapWinGIS is the core component of the MapWindow application a GIS software (http://www.mapwindow.org), which can be used to visualize ESRI shapefiles, and image and grid data. Auxiliary activities such as searching for files in a specified directory with given extension, writing data to Excel, NetCDF files and SQL databases are also placed in this library.

Table 3.2 Summary of efficiency criteria in usages

Name	Equation	Fittness
Nash-Sutcliffe efficiency	$E = 1 - \frac{\sum_{i=1}^{n}(obs_i - sim_i)^2}{\sum_{i=1}^{n}(obs_i - \overline{obs})^2}$	Range: -∞~1.0 1.0-best fit
coefficient of determination	$r^2 = (\frac{\sum_{i=1}^{n}(obs_i - \overline{obs})(sim_i - \overline{sim})}{\sqrt{\sum_{i=1}^{n}(sim_i - \overline{sim})^2}\sqrt{\sum_{i=1}^{n}(obs_i - \overline{obs})^2}})^2$	Range:0~1 0-no correlation
index of agreement	$d = 1 - \frac{\sum_{i=1}^{n}(obs_i - sim_i)^2}{\sum_{i=1}^{n}(\lvert obs_i - \overline{obs}\rvert + \lvert sim_i - \overline{obs}\rvert)^2}$	Range:0~1 1.0-best fit
relative efficiency criterion	$E = 1 - \frac{\sum_{i=1}^{n}(\frac{obs_i - sim_i}{obs_i})^2}{\sum_{i=1}^{n}(\frac{obs_i - \overline{obs}}{\overline{obs}})^2}$	Range: -∞~1.0 1.0-best fit
Root Mean Square Error	$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(sim_i - obs_i)^2}{n}}$	Range:0~∞ 0-best match
Relative Mean Error	$RME = \frac{1}{n}\sum_{1}^{n}\frac{sim_i - obs_i}{obs_i}$	Range: -∞~∞ 0-best match
percentage of peak difference	$dPeak = \left\lvert\frac{Peak_{obs} - Peak_{sim}}{Peak_{obs}}\right\rvert * 100$	0~100 0-best match
** obs_i: observed quantity; sim_i: simulated quantity; $\overline{obs}$, $\overline{sim}$: average value; $Peak_{obs}$, $Peak_{sim}$: peak value of observed and simulated sequence.		

3.5 A Small Watershed Application

In order to demonstrate the functionality of our workflow environment for hydrologic simulations we have selected Furnace Creek a small sub-watershed of the Schuylkill basin (HUC 02040203) in Berks County in south-east Pennsylvania with a drainage area of a little over 4 square miles, as shown in Figure 3.13. Simulation time is just 2 days

(April 21 to 23, 1993), however it contains a severe rainstorm event, with which we can test the ability to reproduce an outflow hydrograph.

By specifying a latitude/longitude bounding box (-76.198°, 40.304°, -76.135°, 40.340°) as shown in Figure 3.13, the 1-arc second DEM data file in GeoTIFF format is retrieved from the NED server and in parallel, hourly precipitation data ranging from 21st to 23rd April, 1993 is downloaded from the NLDAS-2 server. For generating a digital watershed representation the Furnace Creek USGS gauge station (USGS 01470853, latitude: 40° 20'24", longitude: 76° 08'37") is set as the watershed outlet. For comparison purposes the discharge information available at this site (Instantaneous Data Archive with 15-minute intervals) is downloaded and then converted into 1-hour interval data which we do using our temporal interpolation activity.

Working through the tauDEM procedural steps the watershed is then partitioned into nine sub-basins including the river network as shown in the lower right panel in Figure 3.13; note that only four main stems participate in the channel routing calculations. The sub-workflow of processing precipitation data reads the sub-basin information and distributes precipitation data over each sub-basin. However, for a watershed this small (taking up only about one grid cell of NLDAS-2 type data), our sub-basins utilize a uniform precipitation distribution.

For this example we use a loosely coupled hydrologic model that adopts the SCS Curve Number for runoff yield, the SCS Unit Hydrograph method for direct runoff routing, the Recession method for baseflow, and the Muskingum method for channel routing. For short-term rainfall-flood events such as the one chosen, we neglect computation of evaporation. The model first simulates the amount of surface runoff for each sub-basin, and then routs the runoff yield to the sub-basin outlet, where with the baseflow amount is added. The model then proceeds to implement channel routing through the main channels and subsequently produces a discharge hydrograph at the watershed outlet (and also one at the exit of each main channel). For executing the hydrologic model, parameters and initial conditions can either be assigned during the workflow execution or be specified in an XML file, as mentioned before. The entire modeling workflow package consists of six sub-workflow segments as depicted in Figure 3.14. During execution, the first two sub-workflows execute in parallel, and then the other four sub-workflows proceed in sequence.

Figure 3.15 shows the resulting hydrograph plotted next to the one obtained from downloading the flow data from the USGS site. The result window also displays the seven performance measures we have included in our library as a set of post analyses activities. In this case the hydrographs show a fairly good match which is also reflected in the performance measures, however, it should be stressed that this application was not meant to show the accuracy of the methods used, rather to show the utility of the outlines workflow sequences, in fact a demonstration that hydrologic modeling can be done in an environment such as a workflow engine.

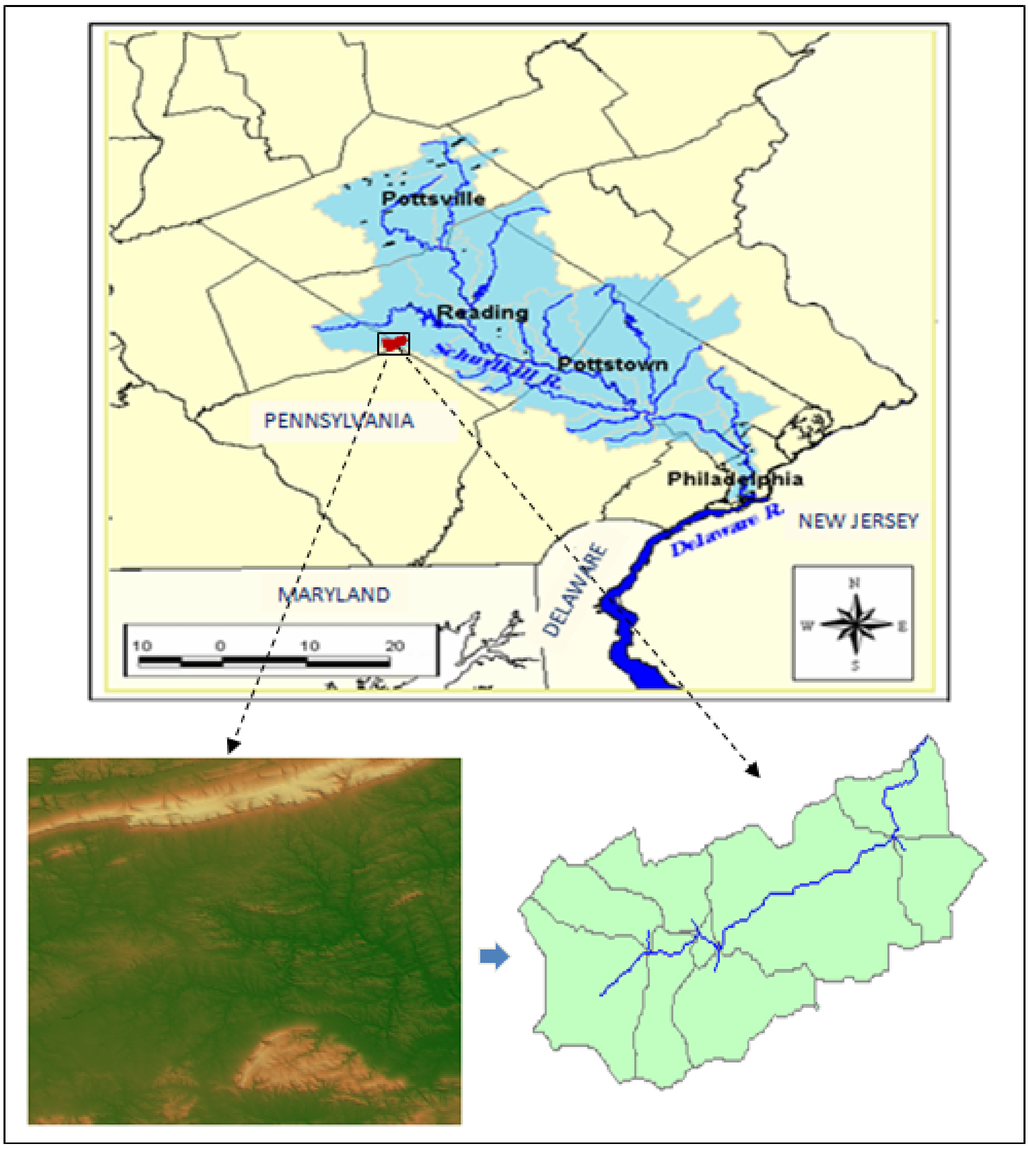


Figure 3.13 The location of the Furnace Creek, the DEM and generated sub-basins and rivers

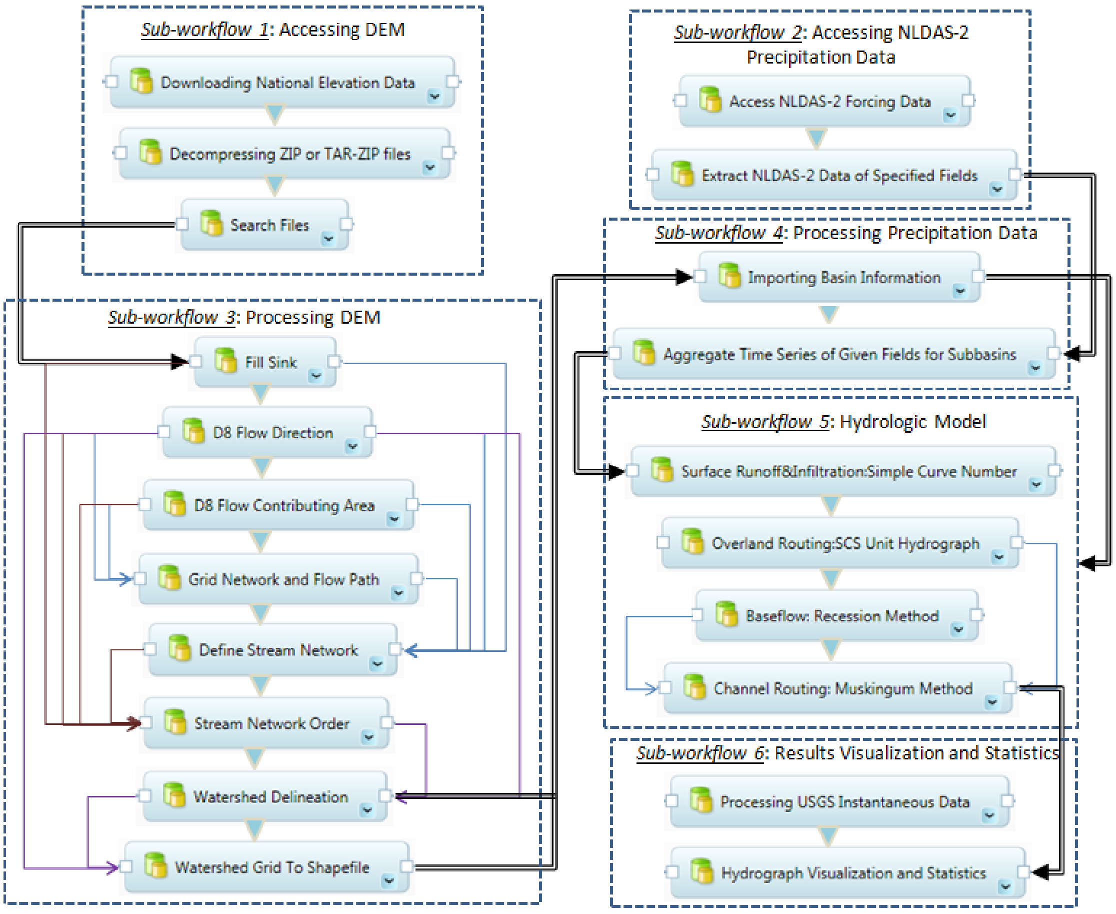
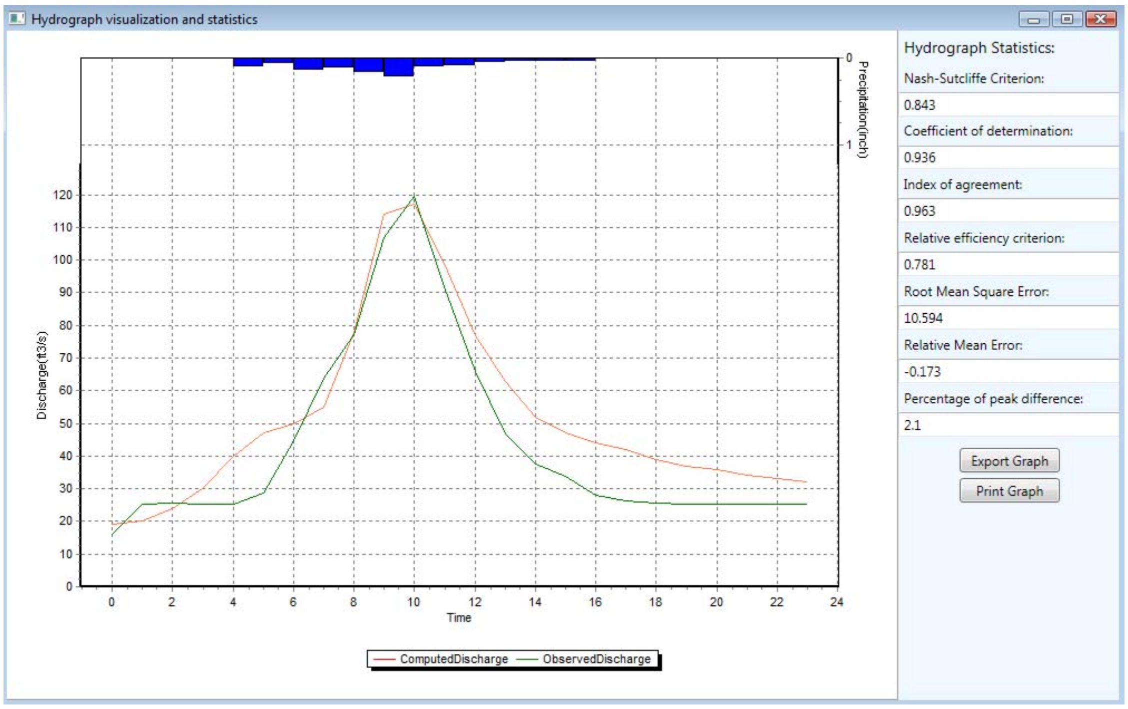

Sub-workflow 1: Accessing DEM
Downloading National Elevation Data
Decompressing ZIP or TAR-ZIP files
Search Files
Sub-workflow 2: Accessing NLDAS-2 Precipitation Data
Access NLDAS-2 Forcing Data
Extract NLDAS-2 Data of Specified Fields
Sub-workflow 3: Processing DEM
Fill Sink
D8 Flow Direction
D8 Flow Contributing Area
Grid Network and Flow Path
Define Stream Network
Stream Network Order
Watershed Delineation
Watershed Grid To Shapefile
Sub-workflow 4: Processing Precipitation Data
Importing Basin Information
Aggregate Time Series of Given Fields for Subbasins
Sub-workflow 5: Hydrologic Model
Surface Runoff&Infiltration:Simple Curve Number
Overland Routing:SCS Unit Hydrograph
Baseflow: Recession Method
Channel Routing: Muskingum Method
Sub-workflow 6: Results Visualization and Statistics
Processing USGS Instantaneous Data
Hydrograph Visualization and Statistics

Figure 3.14 A sample hydrologic modeling workflow sequence



Hydrograph Statistics:

Nash-Sutcliffe Criterion: 0.843

Coefficient of determination: 0.936

Index of agreement: 0.963

Relative efficiency criterion: 0.781

Root Mean Square Error: 10.594

Relative Mean Error: -0.173

Percentage of peak difference: 2.1

Figure 3.15 Comparison of simulated and observed hydrographs

3.6 Summary and Future Work

In this paper, we introduce the development of a hydrologic community modeling system based on the TRIDENT workflow system. Our aim is to utilize the characteristics of a workflow engine, such as provenance capture, repeatability of modeling executions, data handling activities at the pre-, execution-, and post-processing steps, and the ability construct entire modeling tasks within a single environment. The TRIDENT workflow system facilitates our development by providing standard interfaces for migrating legacy models or scripting new ones along with a workbench for manipulating model configurations and executions.

As key components of the environment, we introduce four libraries that are used for pre-processing steps such as establishing connections to data sources via web services and retrieving required data sets; data preparation which concerns steps such as decompressing, extracting of data sub sets, and reformatting data; execution of models for which we have selected three types, i.e. a simple parameterized (TOPMODEL), a distributed approach (SWAT), and a loosely coupled one (individual hydrologic processes); and a post-processing library in which we integrated some data analyses activities such as visualization of hydrographs, performance measures such as the Nash Sutcliff coefficient to several auxiliary activities that perform steps such as unit conversion, data decompression, and I/O tasks.

We have outlined some of the challenges that arise when trying to embed legacy codes into a workflow environment because some of the limitations of the workflow sequencing such as the inability for activities to communicate with each other in time loops, i.e. communications at each time-step level across activity sequences. Another challenge arises from the need to wrap legacy codes written in incompatible languages, such as C or FORTRAN, so they can interact with the activities whose language is found in the .NET environment such as Visual Basic or C#.

We have also demonstrated the utility of the workflow system by applying it to a small watershed in SE Pennsylvania where we executed several workflow sequences to model a two-day storm event. Our objective was to produce an outflow hydrograph at the outlet of the watershed based on the modeling effort and to compare this hydrograph with one derived from a USGS measurement station at the outlet. We showed how web services can be used to find, access, and retrieve data from national catalogues and data sources with for modeling purposes and how these steps are embedded in workflow components called activities. We constructed a seamless workflow from the selection of the modeling area via a bounding box, to the visualization and reporting of performance measures, all fully automated and repeatable in stored workflow sequences. Our future work will focus on the extension of the four libraries we started to build. All of them need expansion to allow for an increased number of data sources, to permit an increased number of modeling alternatives, and to expand post-processing capabilities, for example multiple plotting of time series in one graph, or statistical computations and assessments. One additional important objective is to conduct some comparative tests to find out how the numerical execution times are impacted when embedding larger numerical kernels into an activity, versus embedding it into a WPS web service, versus executing the program as a standalone piece of software. We have yet to quantify impacts on performance as well as the ability to push computations out into the cloud (in this case using MicroSoft's AZURE application). Finally, we would like to conduct some comparisons between smaller and larger watersheds using simpler and more complex models, to obtain a better understanding where trade-offs are and when using a workflow engine may become too cumbersome or even infeasible.

CHATPER 4: APPLICATION AND PERFORMANCE ASSESSMENT USING A WORKFLOW ENGINE BASED HYDROLOGIC COMMUNITY MODELING SYSTEM

Abstract

The TRIDENT-shelled Hydrologic Community Modeling System (HCMS) is used to design workflow-based hydrologic model systems that incorporate components of data access, data preparation, model, and post analysis. The key strength of the HCMS lies in its ability of composing seamlessly integrated models with swappable modules, thus offering users alternative choices of hydrologic model assemblies. The objective of our study is to assess the applicability, ease-of-use, and performance of the TRIDENT based HCMS. Comparisons are made on the performance of accessing and processing precipitation data, discretizing the watershed using different delineation schemes, estimating alternative potential evapotranspiration (PET) schemes, and simulating hydrologic responses via the migrated Soil and Water Assessment Tool (SWAT) and TOPography based hydrologic MODEL (TOPMODEL) along with a loosely coupled hydrologic model. Generally the workload of conducting hydrologic simulations can be reduced significantly when using the HCMS. Performance can be improved when using parallel or distributed computations. Further, decline in performance can be compensated by the enhanced flexibility of model composition, the capability of capturing model provenance, and the possibility of expanding and growing the HCMS. In this paper, we use the Schuylkill watershed located in the Southeastern Pennsylvania as testbed in which we conduct our performance studies.

4.1 Introduction

The watershed physical processes can be simulated or predicted using numerous and hydrologic models at varying degrees of complexity. These models reach from simple parameterized one-equation models that compute the peak runoff, such as the 'Rational Method', to complex deterministic formulations of linked partial and/or ordinary differential equations describing the governing processes (or at least all those that dominate) that are integrated over two- or even three-dimensional gridded representations of the geo-volume they are supposed to represent. Models that fall into this category include the Soil Water Assessment Tool, SWAT, (Arnold et al. 1993), the Penn State Integrated Hydrologic Model, PIHM, (Qu and Duffy, 2007), or the well known MODFLOW suite (McDonald and Harbaugh, 1988). Of course, the more attention is spent on the detailed representation of the governing processes the higher the work load required to get just one model run off the ground. Typically more complex models are also more input data hungry and consequently much more time is needed to prepare all data sets that are required (boundary conditions to force the model and initial conditions to start the numerical integration) for the chosen spatial and temporal scales especially if one is to track time variant events throughout the duration of the simulation. The same can be said for the generated output data that can be quite voluminous requiring advanced tools to cut, slice and dice through the contents for analysis or even just plain visualization.

It is easy to imagine that if one were to use several models at the same time for example, because one wants to investigate the impact of differences between theoretical formulations or shortcomings of chosen approaches, that the work effort required for preparation and execution becomes even higher. Hence, it is no surprise that communities have sought solutions to ease the work burden and, for example, agreed to establish testbeds for model skill assessment where researchers can compare their model side-by-side to other models, such as the Delaware Bay Modelling Evaluation Environment, or MEE, that was started by researchers at NOAA (Patchen, 2008; Stammermann and Piasecki, 2009). This approach assumes that many different modellers contribute their models and runs to a library of sorts where researchers can then examine how models compare to each other for a given data set for a specific time period. However, while quite helpful and potentially revealing the idea suffers from the need of substantial manpower (in from of many contributing to the effort) because a system of this type offers nothing to reduce the workload on data pre- post-processing as well as the model execution needs at run-time. While the aforementioned system is just one approach that has been used to create “modeling systems” and that is mentioned here as a motivation for the work presented, there are in fact quite a number of what are called Community Modeling Systems (CMS) approaches that are currently being suggested and pursued with various degrees of organization, different strategies and principles (for a review on those see Bo and Piasecki, 2011a).

An alternative approach is to provide an environment in which a single researcher can perform comparison tasks of the type just described without spending the same amount of time otherwise required for the execution of many different models. In this environment a) much of the data preparation burden is automated, b) model selection is easy because several models can be selected, c) post processing can be automated for all model results regardless their structure, and d) any of these runs can be repeated because provenance information is recorded as well. The authors have developed a prototypical Hydrological Community Modeling System (HCMS) based on a workflow engine environment (Bo and Piasecki, 2011b). This system takes advantage of a heavyweight backbone structure in which scientific workflows can be generated and executed; modeling workflows being one of the prominent applications that can be embedded into such an environment. For a more detailed introduction into the development of the environment please refer to Bo and Piasecki, 2011b.

In this paper we seek to apply our prototype HCMS to a testbed in order to test its utility for serving as a Hydrologic Community Modeling System, including the ease-of-use, performance assessments when executing programs that are either embedded or are accessed via web service calls, the generality of basic library modules, and/or potential shortcomings and difficulties in using this system.

4.2 Tools, Testbed, and Tasks

4.2.1 Overview of the Hydrologic Community Modeling System (HCMS)

Since it is not the scope of this paper to present the details of the HCMS (the reader is referred to Bo and Piasecki 2011b), a short overview may be helpful to set the context of the main thrust of this paper.

Our HCMS has been developed based on Microsoft's TRIDENT workflow engine (Microsoft Research Group, 2009), and provides a platform for the composition and execution of various hydrologic-modeling-oriented workflows. Within this system a hydrologic model kernel can reside as a single node or a sub-workflow in the parent workflow performing the comprehensive simulations. The HCMS is comprised of four libraries: one for accessing data from either internet-based data servers or local storages such as files or databases; one for processing geospatial data or time series to meet the requirements of the model inputs; one for performing hydrologic simulations, and finally one for analyzing and post-processing model results using for example statistics calculations, or performance measure such the Nash-Sutcliff coefficient, or storage activities that would write out result in netCDF (among others).

One of the main features we have explored is the ability of the TRIDENT system to access web services, which provides a convenient way to access national data stores such as US Geological Survey National Water Information System, or the USGS MapServer for digital elevation data to name just a few of several for which we have developed customized access points. These access points are organized in the access library. The hydrologic modeling library features the TOPography based hydrologic MODEL, TOPMODEL, and the hydrologic components from the Soil and Water Assessment Tool, SWAT, which we have been migrated into the HCMS. The purpose of this specific selection was to demonstrate the possibility of migrating two models of intermediate and high complexity, respectively, into the HCMS. The HCMS also features a number of activities (or compute kernels) encapsulating basic algorithms for simulating specific hydrologic processes (e.g. overland runoff yield and routing, channel routing etc.) which have been incorporated and organized in a hydrologic component library. While this library is not all encompassing yet, it does offer users a high degree of flexibility to customize models being able to swap modules in and out (or run them in parallel) when designing the workflow sequences.

Last but not least the post-processing library contains a number of modules that help with the visualization and analysis of the results. There are a number of performance measures for assessing time series data produced by a model and compared to in-sit measurements, such as discharge, that are quite popular in the hydrologic community. We have added indexes such as the Nash Sutcliff and RSME to this library in addition to having added some plotting capabilities for time series data. There are also output activities that allow writing the data out in any desired custom format or commonly used formats such as netCDF or GRIB, or even data streams that deposit results in a database.

4.2.2 Testbed: Schuylkill River Catchment

It was decided to use a sizable catchment in the 5,000 square kilometer range to better understand how the HCMS would respond or handle computational demands of a catchment this big, without needing to contrast it to larger watersheds with even higher computational demands prompting longer wait times. There was also the need to select a catchment that has a good collection of measurement stations either inside or in the immediate vicinity of the catchment for testing the interpolation routines for re-assigning data at certain geospatial locations to locations in the catchment where data was needed for simulation. Lastly, the research team also wanted a catchment that brings some familiarity, preferably one that is as close as to Philadelphia. The Schuylkill catchment provided an ideal candidate; in fact the outlet point of the catchment (which is a dam that also serves as head-of-tide for the tidal portion of the Schuylkill River) is just a stone throw away from Drexel University's premises.

The Schuylkill catchment (Hydrologic Unit Code, HUC, 02040203) is located in Southeastern Pennsylvania as shown in Figure 4.1. It is about 132 km in length, 41 km in width, and covers an area of approximate 5229 square kilometers. The Schuylkill River is the largest tributary of the Delaware River system and accounts for about one third of its total flow in the lower reaches. The river itself is about 214 km long starting from its headwaters at Tuscarora Springs in Schuylkill County, and ending at its mouth at the Delaware River in Philadelphia. Major tributaries to the Schuylkill River include Little Schuylkill River, Maiden Creek, Tulpehocken Creek, Manatawny Creek, French Creek, Perkiomen Creek, and Wissahickon Creek, all of which run through mixed land use zones that are dominated by sub urban sprawl developments interspersed with some agricultural and forested land areas, with the latter dominating the uplands and the former being prevalent in the lower reaches.

The Schuylkill experiences a modified continental climatic; warm and humid in summers, moderately cold in winters, and has abundant rainfall distributed throughout the year. The mean annual temperature is about 11 °C with summer and winter averages of 22 °C and 0 °C respectively. The mean annual precipitation is around 45-50 inch/yr in the mountainous headwaters region, and decreases to 43 inch/yr eastward to the Coastal Plain. Precipitation is distributed fairly uniformly throughout the year, and is generally sufficient for crops and vegetables. In Pennsylvania, approximately 50% of annual precipitation is evaporated or transpired by plants back to the atmosphere, 20% runs off into rivers and stream during rainfall and snowmelt events and the left 30% infiltrates the ground surface to recharge groundwater aquifers (Fleeger, 1999; Biesecker et al., 1968). The stream flow rate tends to be highest in late winter and early spring generally due to snowmelt and low evaporation and transpiration, while be lowest in late summer and early fall primarily due to high rates of evaporation and transpiration.

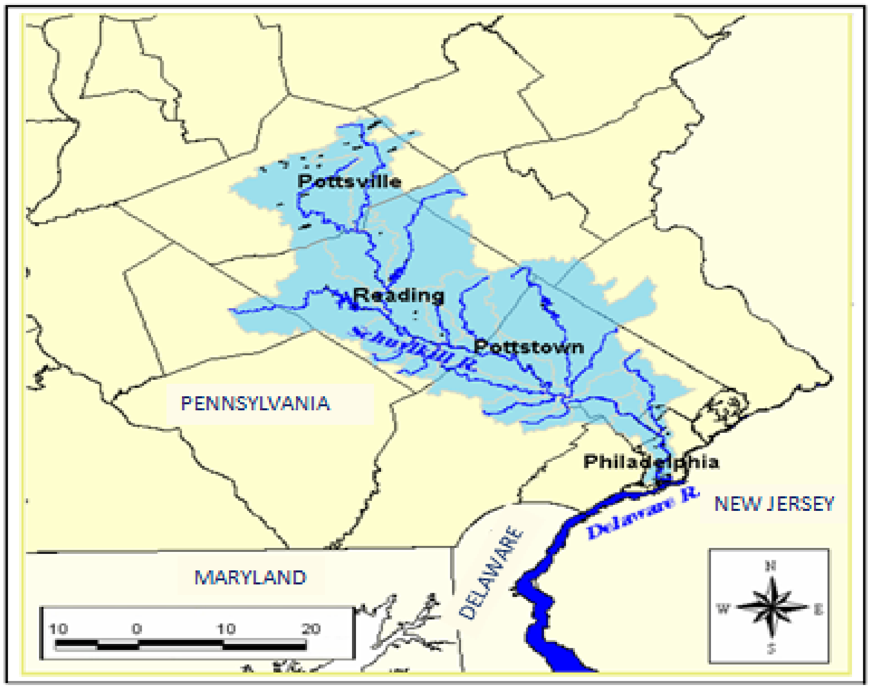


Figure 4.1 Location of the Schuylkill river watershed

4.2.3 Modeling Tasks

The decision was made to use a sizable catchment in the 5,000 square kilometer range in order to demonstrate the utility of the HCSM we execute a number of tasks that are typical to a hydrologic modeling effort. First, we apply the SWAT module to simulate daily runoff hydrographs based on data retrieved over a 4 year period ranging from 2005 to 2008. This task is intended to test the ability to extract large data sets and to subset them for preparation of a SWAT modeling exercise, and at the same time measure how much CPU clock is used (or needed) to execute a simulation this long using the daily time step SWAT typically utilizes for advancing the model's state. Secondly, we use TOPMODEL along with a loosely coupled hydrologic model for the simulation of a flood event, i.e. a significant rainstorm moving the catchment (note that SWAT is not well suited for this type of simulation because of its daily time step which is too coarse for simulating rainstorm events).

For both cases, first steps include collection and processing of data by using different data resources and subsequent activities, where we conduct a correlation analysis on the precipitation accessed from different data sources. We then use workflow sequences to assess the speed by which DEM processing is executed and contrast an approach in which we have each step embedded into a separate activity with one in which we have embedded all steps into a single module that is accessed via web service calls. This test will aid in understanding to what extent the workflow engine will degrade execution or run time performance when carrying out model runs through this additional layer of software and thus give a ballpark estimate what size watershed will consume what wall clock or computational resources.

Another important task is to evaluate how complex configuring the various models is, and how many interactions need to be implemented versus how general a preparation activity can be when setting up model simulations. The workflow for the SWAT-simulation involves six activities; one each for modeling potential evapotranspiration (PET), snow melt, overland flow, soil water, groundwater flow and channel flow. While there are choices for any of these modules, we decided to isolate PET for a comparison using all four alternative PET activities (in parallel) each of which adopts a different calculation approach. The workflows for TOPMODEL and the loosely-coupled model are comparatively simple both of which only require four activities for simulating a three-day flood event (note that there are additional activities for data preparation and post processing). The final step is to import the modeled and the observed streamflow data into the activity responsible for result analysis, which presents the comparisons in terms of plotting the hydrographs and computing the statistic criterion coefficients and water budgets.

4.3 Data Preparation and Manipulation

4.3.1 Data Retrieval

Depending on the complexity of model used hydrologic simulations can be data intensive efforts. A complex model such as SWAT (recall that we use only the hydrologic components here) requires a complementary set of complex input data which covers Digital Elevation Model (DEM) data, land cover and soil data, meteorological data (e.g. precipitation, solar radiation) and hydrologic data (e.g. water level, discharge). HCMS supports direct data transfer from a list of online databases based on SOAP or FTP protocols, of which the U.S. Geological Survey (USGS) National Elevation Dataset (NED) and the National Land Cover Data (NLCD) provide DEM and land cover data, the National Resources Conservation Service (NRCS) Soil Survey Geographic Database (SSURGO) distributes soil survey spatial and tabular data, the Environmental Protection Agency (EPA) provides accesses to hydrographic data, while the North American Land Data Assimilation System (NLDAS), the Hydrologic Information System (HIS) and the National Water Information System (NWIS) maintains accesses to a large amount of meteorological and hydrologic data. Table 4.1 summarizes the available data resources and data utilized in this study.

As a first step HCMS downloads DEM data with a spatial resolution of 1 arc second along with land cover data for the rough extend of the Schuylkill River catchment area. HCMS then has the ability to access NRCS' Soil Data Access web service in order to retrieve SSURGO data via a custom made activity. However, these web services currently only allow transferring data from within a restricted area that is significantly smaller than the Schuylkill catchment which is due to bandwidth restrictions at the host site. Efforts are under way to remedy this situation and to permit larger area downloads. While we could have addresses this problem through creating "stitching" services, we decided to obtain soil data through the Soil Data Mart (http://soildatamart.nrcs.usda.gov/) instead, which allows downloading soil survey spatial and tabular data at the spatial scale of a county (which is larger than Soil Data Access area/download). For the Schuylkill catchment this required the downloading of eleven soil data sets and while this still required some "stitching" this was easier to carry out than to work with the service described above. In a third step gridded hourly data of precipitation, temperature, air pressure, long wave/short wave solar radiation and vertical/horizontal wind speed ranging from 2005 to 2008 are retrieved from the NLDAS-2 data server. Data sets are provided on a nationwide grid on an hourly basis and are stored in GRIB file format (which is a binary format). This poses several challenges. For one, HCMS needs an activity to download the GRIB files for a specified time period. This process repeats itself for each hour, i.e. 8760 times for each year simulated. It then needs another activity to invoke the GRIB library in order to de-GRIB the files and extract data of the required meteorological quantities as well as do some directory clean up by deleting GRIB files that have been processed. For the envisioned 4 year simulation, it takes nearly 6 hours to retrieve all GRIB files and 120 hours to decode them. This may seem excessive and in a way could be construed as prohibitive but one only need to compare this to a manual process (or perhaps semi manual process using batch files, WINDOWS, or scripts, UNIX) to realize what time savings this generates considering that this is a 'from-scratch' modeling exercise.

Table 4.1 Information of data resources and usages

Acronym	Data Availability&Access	Data Scale	Data In Use
USGS NED	National Elevation Data, accessed via "Application Services"	1,1/3,1/9 arc second	1 arc second (39.86,-76.4,40.9,-75.1)
USGS NLCD	National Elevation Data, accessed via "Application Services"	30m*30m	(39.86,-76.4,40.9,-75.1)
NRCS SSURGO	Soil survey spatial and tabular data, accessed via "Soil Data Access" web services	Currently only for restricted areas	Accessed via Soil Data Mart, covering 11 counties.
HIS	Hydrological and Meteorological data, accessed via "WaterOneFlow" web services	Varied temporal scales	Streamflow of Schuylkill River(Station: USGS 01474500)
NLDAS-2	Meteorological data(temperature, precipitation, radiation etc),accessed via FTP	1/8 degree, 1979.1-present, 1-hour	2005.1.1-2008.12.31
NWS MPE	Multi-sensor Precipitation Estimates, accessed via FTP	4km*4km, 2005.1.1-present, 24-hour	2005.1.1-2008.12.31
EPA NHD	National Hydrography Dataset (watershed and stream shapefiles), accessed via EPA Geospatial services	Medium and High resolutions	N/A

Daily precipitation data (Multi-sensor Precipitation Estimates, MPE) for the same temporal range is retrieved from NOAA's National Weather Service, NWS, River Forecast Centers. MPE precipitation data, just as the NLDAS data, is provided on a grid (in this case the HRAP grid) with nationwide coverage, one file for each day. These data are also encoded in a binary file format, in this case NetCDF which need to be decoded using the netCDF libraries (these together with the GRIB library have been ported into the HCMS and made available via activities). These downloads are much less time consuming and are handled within minutes. Finally, daily value discharge data from Schuylkill station (USGS 01474500) are obtained from the Hydrologic Information System server (HISCentral, which provides all USGS station data as a proxy) via activities invoking HISCentral's WaterOneFlow web services. Instantaneous discharge data (15-minutes intervals) data for the targeted flood event are downloaded from the USGS NWIS website directly.

4.3.2 Watershed: GeoSpatial Computations and Activities

4.3.2.1 DEM Processing

DEM processing seeks to partition a watershed into smaller units representing homogeneous topographic characteristics within the watershed and basically follows the modules and approaches as implemented in tauDEM (Tarboton et al., 1991). The HCMS offers three different pathways of working through the DEM processing steps, i.e. catchment delineation, subdivision of the catchment into sub-catchments, and identification of the stream network. The first alternative is to use a web service based activity in which the activity engages a dedicated server for the compute intensive DEM processing and which then sends features of the watershed and stream network back to the activity. Herein the DEM processing web service is developed based on the OpenGIS Web Processing Service (WPS) interface standard and intended a) to allow anybody in the community to access this service, i.e. outside the HCMS, and b) to off-load computationally burdensome processes to a powerful server thus reducing the load on the client machine. The second workflow alternative is an activity that resembles the first one in its definitions of inputs and outputs, but instead of calling a web service, it performs the whole procedure of DEM processing locally, i.e. inside an activity. The third alternative is the step-by-step DEM processing that uses separate activities handling different steps thus generating a workflow sequence. This alternative was introduced to allow for swapping in and out activities in case updates and improvement became available without having to change a monolithic code or simply to use an alternative process. The corresponding workflows are shown in Figure 4.2.

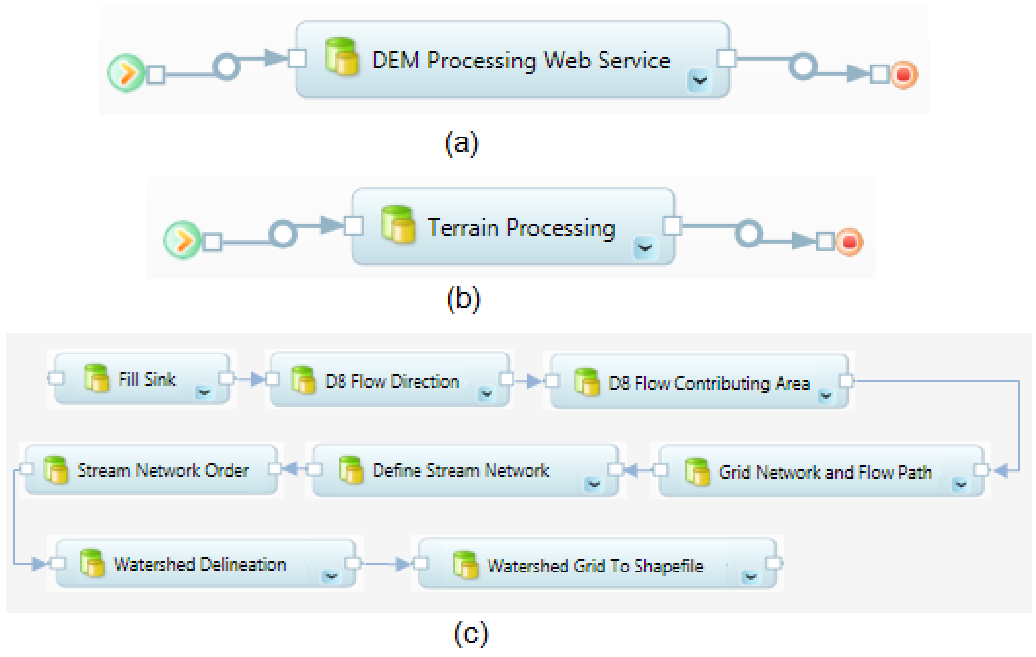


Figure 4.2 The workflows of DEM processing: (a) using the activity invoking the WPS-based web service (b) using the activity conducting the whole processing (c) using activities responsible for each step.

For performance comparisons all three workflows are applied to process DEM data for the Schuylkill River catchment. We have created a supplemental workflow that identifies the outlet location of a catchment right at the USGS gauge stations, so we can better compare the computed and observed discharge (Figure 4.3). For the first workflow it is unrealistic to wrap a large DEM data set into a SOAP message. Instead, the activity proceeds to upload the data from the external FTP site and deposits it on the same server that hosts the DEM processing web service. In other words, the activity not only calls the WPS service for execution but also provides for the necessary data by using FTP protocol to get the data from the source, thus effectively involving three servers. This is much faster because downloading data to the WPS service server host is much less work intensive than having the workflow engine host server having to upload the very same data to the WPS server. However, when compared to the standalone version of tauDEM there is a certain level of performance loss using any of these workflows. The first workflow may take a few extra minutes to upload the DEM data depending on the current network speed. In addition, the TRIDENT engine requires a few seconds (up to one minute) to launch the execution of the workflow depending up how much CPU has been taken up by other processes on the computer. The third workflow comprised of several individual activities actually needs extra time, usually several seconds, invoking each activity. However, it should be noted that even while the standalone version does not suffer from the overheads imposed by the workflow environment, processing of a catchment this large took more than fourteen hours to complete (on a ACPI Multiprocessor X64-based PC with 15.9 GB RAM and two 2.29GHz Quad-Core AMD CPUs). In light of this overall computational burden those few extra minutes should hardly matter though, even if on uses a powerful server that could reduce the processing time to just a few hours. Figure 4.4 shows the 1 arc second DEM (4681*3744 cells) and the delineated watershed with 7 sub-basins as well as the generated stream network.

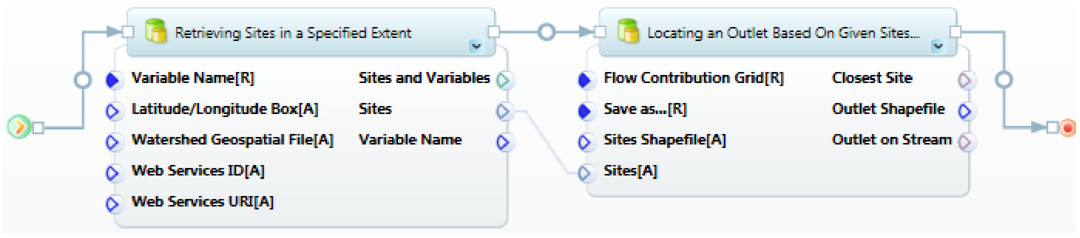


Figure 4.3 The workflow for locating the outlet of the watershed

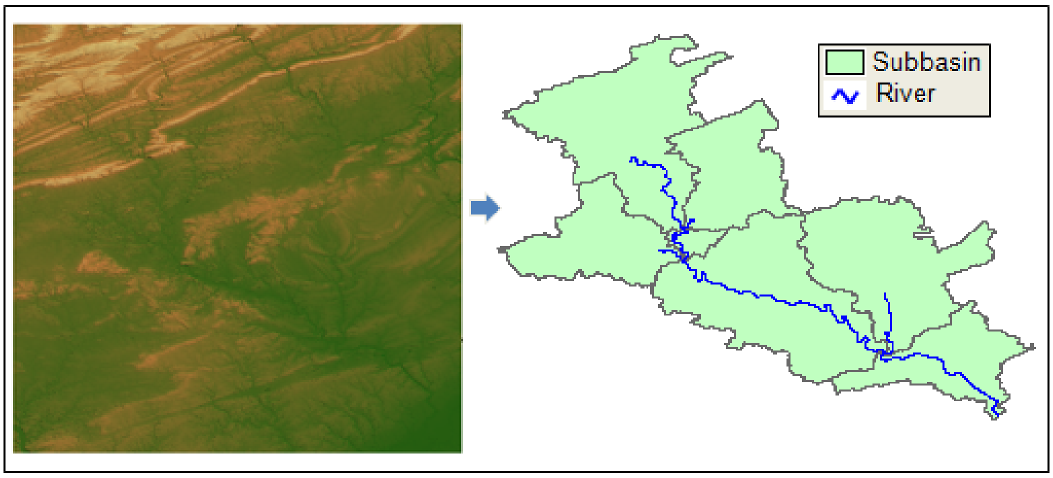


Figure 4.4 The DEM, Delineated Sub-basins, River System of Schuylkill Watershed

The right degree of watershed sub-division is often not known a prior and probably need to be found through trial and error by using ever finer sub divisions until little change can be observed. The literature too is not equivocal about it; for example, Mamillapalli et al. (1996) showed that finer subdivisions of a watershed can lead to more accurate predictions of streamflow whereas FitzHugh and MacKay (2000) as well as Jha et al. (2004) stated that the variation in total number of sub-basins had very little effect on streamflow. Hence, in our study, we utilized two types of subdivisions and attempted to highlight their influences on the streamflow simulation.

The watershed subdivision is controlled through a critical source area (CSA) threshold which is the minimum upstream drainage area for a channel to originate, and specified as the number of cells or the percentage of total watershed area (Di Luzio and Arnold, 2004). Lower CSA numbers results in a higher degree of sub-division with a denser river system and more sub-basins. The opportunity to use various degrees of sub divisions offers up another test scenario, in which we can use the HCMS to execute parallel computations with the intent of learning more about a 'good' sub division. The threshold number set for the first division was 500,000 resulting into 7 sub-basins. A finer subdivision was obtained by using the threshold number of 100,000 which resulted in the generation of 33 sub-basins, as shown in Figure 4.5.

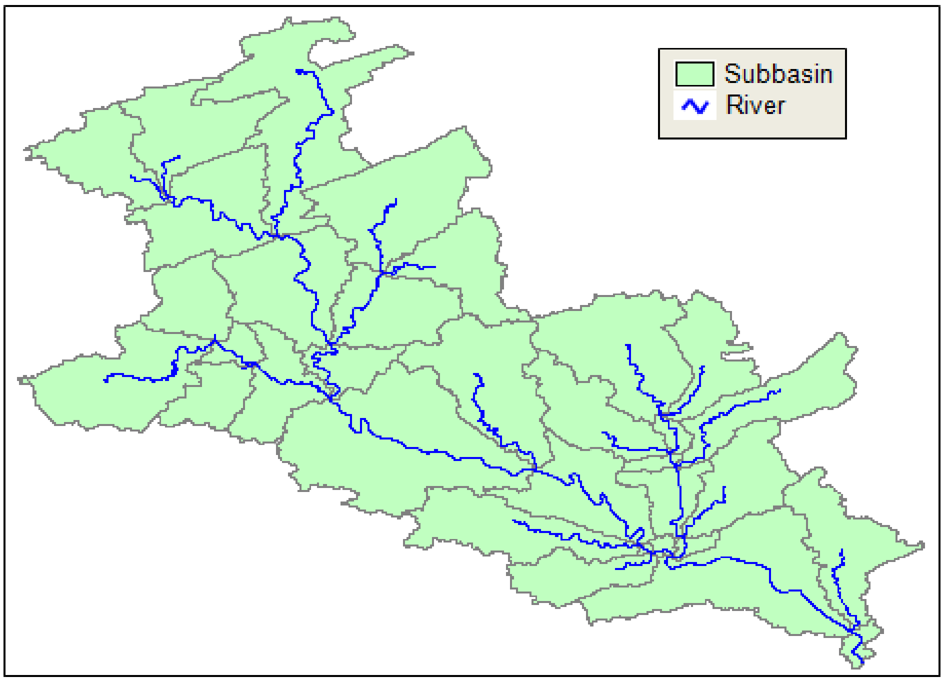


Figure 4.5 The watershed generated by a finer subdivision

We used the third workflow alternative to compare results from these two sub divisions, which is a good example how to effectively use the HCMS. Since the only difference between these two catchments lies in their stream network, we executed a general DEM workflow to the stage where the stream network is determined, after which we split the workflow, one continue using the 7 sub-basins and the other using the 33 sub-basins set up which then run in parallel. It avoids implementing the same data operations repetitively, especially when considering the time consuming process of ‘fill sink’ which alone accounts for over thirteen hours of processing time.

4.3.2.2 Construction of Hydrologic Response Units (HRUs)

In SWAT, sub-basins of a watershed are further subdivided into HRUs that consist of homogeneous land cover and soil characteristics. The HRUs are commonly identified as a combination of a land cover type and a soil type, and are accounted for by area percentages of the total sub-basin area. A HRU is not necessarily a homogenous and connected unit but is comprised of a number of cells that are scattered throughout the sub-basin but are not identified and referenced using geospatial coordinates. Before constructing the set of HRUs, several data operations are required over the land cover and soil data of the basin.

Firstly, the land cover grid is projected to match the coordinate system of the DEM after which it is then clipped along the Schuylkill River catchment boundary eliminating the cells outside of the catchment. The National Land Cover Dataset has a classification system consisting of 15 land cover classes for the mainland US, of which the Schuylkill River catchment contains 13 types except the Shrub/scrub and the Grassland/herbaceous classes. We assumed an additional meta-class system with the intent to reduce the HRU workload resulting into 4 meta-classes as shown in Table 4.2. This reclassification is carried by a dedicated activity and consumes about 70 minutes. The final land cover grid is shown in Figure 4.6.

Table 4.2 Current and Original Land Cover Classes

Current Land Cover Classification		**Original Land Cover Classification**	
Assigned ID	Type	ID	Type
1	Water	11	Open Water
		90	Woody wetlands
		95	Emergent herbaceous wetlands
2	Medium Residency	21	Developed, open space
		22	Developed, low intensity
		23	Developed, medium intensity
		24	Developed, high intensity
3	Forest	41	Deciduous forest
		42	Evergreen forest
		43	Mixed forest
4	Agriculture	31	Barren land
		52	Shrub/scub
		71	Grassland/herbaceous
		81	Pasture/hay
		82	Cultivated crops

The soil geospatial data are originally stored in 11 separate GIS shapefile (one each for every county), each of which contains a large number of shape attributes that are embedded as map units in the SSURGO set. First, for each shapefile the map units are merged based on the hydroloigc groups they belong to. Then, the 11 shapefiles are then merged into one, which is then subsequently clipped along the catchment boundary. This procedure has also been coded into a single activity, which takes a substantial time to execute. The final soil grid shows that the Group B and Group C take dominant roles in the watershed (Figure 4.7).

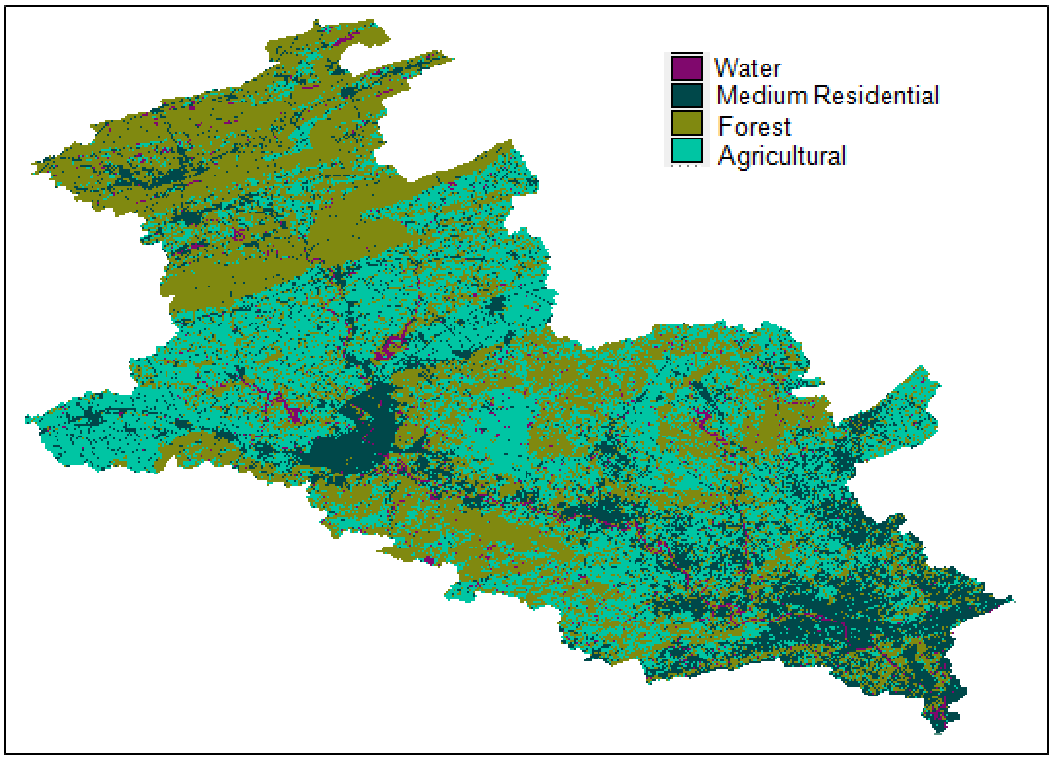


Figure 4.6 Reclassified Land Cover Grid

With the resulting products from the DEM processing, along with the pre-processed land cover and soil data, the HRUs can now be created automatically by yet another specific activity. For the coarsely divided watershed (7 sub-basins) 23 HRUs are created, while the finer gridded one (33 sub-basins) yields 114 HRUs. In order to make the HRU determination a one-time processing effort, metadata pertaining to the HRU classifications are recorded in XML files for future reference. The workflow sequence of building HRUs is shown in Figure 4.8, where the activities for processing land cover and soil data are executed in parallel and then linked to the activity of HRU constructor.

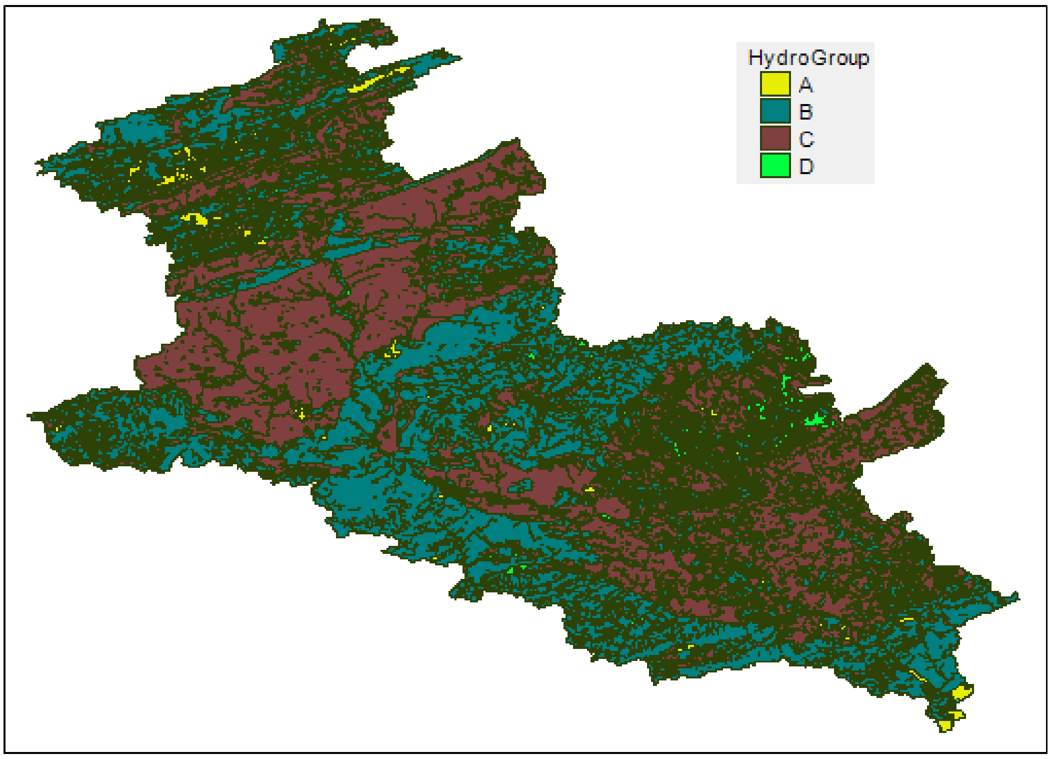


Figure 4.7 Distribution of Soil Groups

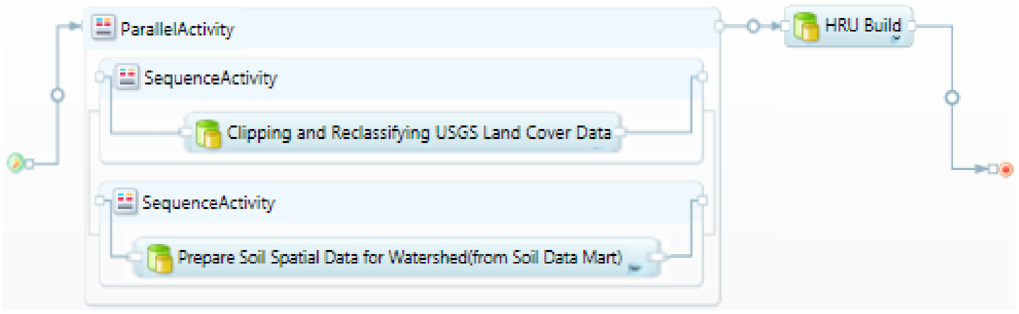

ParallelActivity
SequenceActivity
Clipping and Reclassifying USGS Land Cover Data
SequenceActivity
Prepare Soil Spatial Data for Watershed(from Soil Data Mart)
HRU Build

Figure 4.8 The workflow of constructing HRUs

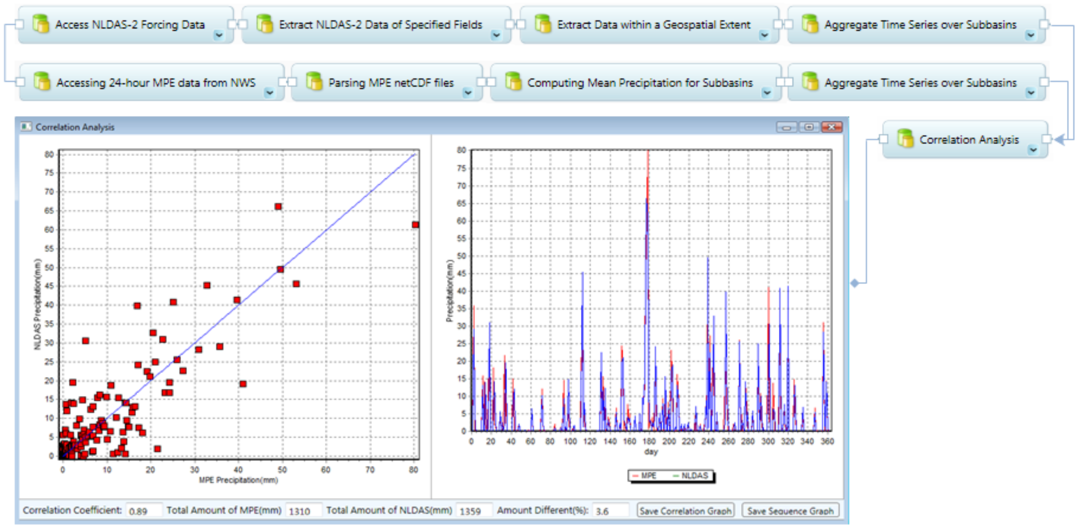
4.3.3 Watershed: Temporal Data

4.3.3.1 Precipitation

Precipitation data is probably the single most important driving force of hydrologic models; hence accurate determination of the spatial and temporal extent and variation is of crucial importance. The HCMS can access precipitation data from two national data centers; NLDAS and NOAA/NWS MPE. MPE data can be obtained in finer spatial (HRAP grid with approximately 4x4 km spacing) but coarser temporal resolution (daily, even though an alternative system providing hourly data now exists for most of the continental US) than that of NLDAS, which delivered on a 224x464 grid (13.8x13.8km spacing) with hourly intervals. One additional test for which the HCMS is ideally suited is the comparison of these two rainfall products for the Schuylkill catchment for which we target the mean daily precipitation. For the NLDAS set the gridded hourly precipitation data is first extracted from the NLDAS GRIB files, is then aggregated over the sub-basins they located in, is subsequently accumulated for each day, and is finally averaged over the entire catchment based on the area-weighted-factors.

MPE daily data are directly extracted from the NetCDF files followed by an aggregation over the sub-basins and subsequently over the entire watershed to produce a comparable number. Both time series are then imported into the correlation analysis activity, which displays a scatter plot of NLDAS precipitation against MPE precipitation, a linear plot of each precipitation sequence against time and the statistics of correlative coefficients, total precipitation amounts and the percent differences. We have developed a number of post processing activities for these analyses steps and subsequent displays. A sample workflow depicting this process along with the result window is shown in Figure 4.9, where the result window features exporting the graphs in different formats.

The workflow sequence is implemented separately for each of the years from 2005 to 2008 based on using the 7-sub-basin catchment. The scatter plots of NLDAS precipitation against MPE precipitation are shown in Figure 4.10, while the plots of precipitation against time are not listed here for space consideration. The correlation coefficient is around 0.9 with the exception of year 2007, while the percentage of amount difference stays between 1.4% and 5.3%. Since the NLDAS and NWIS precipitation data exhibit a good correlation and the daily NWIS MPE data are not suitable for the short-term flood simulation performed by the workflows of TOPMODEL and the loosely-coupled model, we use NLDAS precipitation data through the following simulations.


Access NLDAS-2 Forcing Data
Extract NLDAS-2 Data of Specified Fields
Extract Data within a Geospatial Extent
Aggregate Time Series over Subbasins
Accessing 24-hour MPE data from NWS
Parsing MPE netCDF files
Computing Mean Precipitation for Subbasins
Aggregate Time Series over Subbasins
Correlation Analysis
Correlation Analysis
NLDAS Precipitation(mm)
MPE Precipitation(mm)
Precipitation(mm)
day
MPE
NLDAS
Save Correlation Graph
Save Sequence Graph

Correlation Coefficient: 0.89 Total Amount of MPE(mm) 1310 Total Amount of NLDAS(mm) 1359 Amount Different(%): 3.6

Figure 4.9 The workflow sequence for retrieving and processing MPE and NLDAS precipitation data, and making correlation analysis.

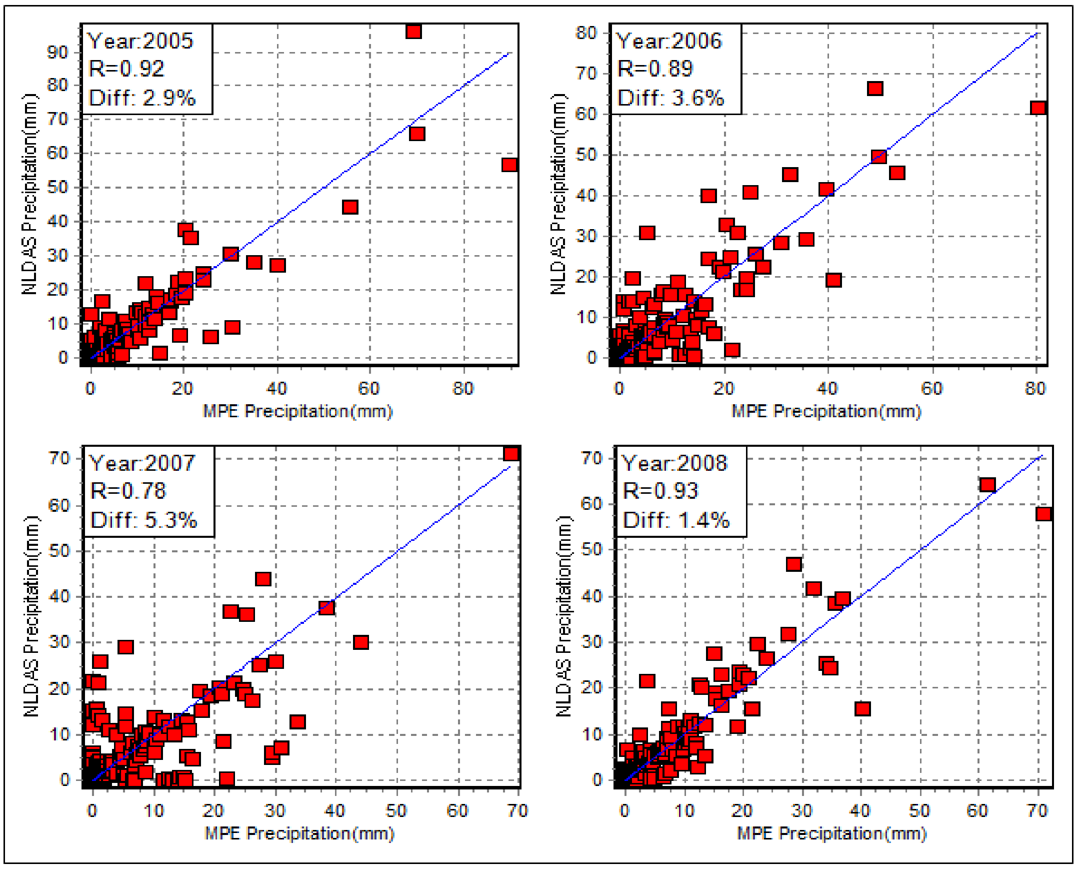


Figure 4.10 Scatter plots of daily precipitation of MPE and NLDAS for the years 2005 to 2008. The correlation coefficient(R) and percentage of amount difference (Diff) are shown on each graph

4.3.3.2 Other driving forces and observations

Apart from precipitation other meteorological forcing data within HCMS can (currently) only be obtained from the NLDAS database; their retrieval operations are identical to those of retrieving of precipitation data sets. Any of these variables are assumed to distribute evenly over each sub-basin as well. When converting the original hourly data into daily data, the averaging and aggregating operations are conducted on different meteorological quantities. For example, daily temperature is averaged from hourly temperature readings, and daily solar radiation is accumulated from hourly radiation measurements. In practice, the sub-workflow addressing NLDAS data processing, shown in Figure 4.9, is utilized to perform data operations for all meteorological quantities even though only a sub set may be used. The computational cost of this potential ‘over-processing’ is small compared to alternative of developing an interactive activity in which the user selects what he wants. Note that the only difference between the 7 and 33 sub-basin cases is the execution of the aggregation activity for each sub-basin. Consequently, we have isolated this activity from the general workflow chain and placed it in a parallel smaller workflow (which branches off at first before rejoining the longer sequence again later) that executes the aggregation steps for the 33-sub-basin case. This is easily done in the workflow composer and avoids having rerun the entire workflow, in fact it allow for any number of parallel executions for whatever degree of sub division desired.

As mentioned before daily observations of streamflow at the Schuylkill station are retrieved via WaterOneFlow web service calls, which require no additional workflow operation except a small validation process that addresses data consistency and completeness. Since the WaterOneFlow can only access instantaneous data going back one month, the required streamflow data are downloaded from the USGS instantaneous archives. An activity is utilized to extract the streamflow data array from the downloaded file and formatted into the HCMS defined time series (see Lu and Piasecki, 2011b for more details). It enters into the interpolation activity to produce the hourly observed discharges since the original temporal interval is half hour.

4.4 Hydrologic Modeling

4.4.1 Estimate of Potential EvapoTranspiration (PET)

Potential evapotranspiration can be described as the maximum amount of water that can be evaporated within a time period from a vegetated region (Thornthwaite, 1948; Penman,1948). PET is one of the major components in the hydrologic budget calculations and while its accurate determination is obviously of great importance it also presents a great deal of difficulty when trying to quantify it. Numerous methods have been developed to estimate PET for various scenarios resulting in the applicability of one method in one setting while producing less accurate results in another. Thus the HCMS activity library is host to several approaches, namely the Penman-Monteith method (Monteith,1973), the Hargreaves method (Hargreaves, 1975; Hargreaves and Smani, 1985), the Thornthwaite method (Thornwaite,1948) and the Priestley-Taylor method (Priestley and Taylor, 1972). The Penman-Monteith method has shown to yield excellent PET estimates in a number of studies (Beyazgul et al., 2000; Hussein, 1999; Tyagy et al., 2003). However, it is a method that requires several data feeds including temperature, atmospheric pressure, relative humidity, solar radiation and wind speed and in cases where it is difficult or impossible to find all of these variables the method may not be applicable or loses some of its power in producing good PET estimates. Priestley and Taylor (1972) simplified the Penman-Monteith equation by removing its aerodynamic component and introducing an energy compensation factor, in which the wind speed is no longer required. The Hargreaves and Thornthwaite methods take the simplification a step further and estimate PET merely on the basis of temperature data. Within the HCMS it is quite simple to switch methods of PET estimation or to run comparison between methods. The implemented PET activities produce both daily and monthly PET in which we have added a correction factor (based on the daily photoperiod) suggested by Pereira and Pruitt (2004) to modify the Thornthwaite method to yield daily values as it was originally designed to only yield monthly estimates. Monthly PETs are obtained by accumulating the daily PET at the month scale for the other three methods.

In this particular application, we engage the four PET activities to compute monthly PET for the Schuylkill watershed (7 sub-basins version). The computation is carried out over each sub-basin at first and then proceeds to the spatial aggregation over the entire watershed. Since there was no observation gauge within this watershed, we set the Penman-Monteith estimate as the benchmark value. The monthly PETs from January 2005 to December 2008 are plotted as shown in Figure 4.11 (a). The PETs estimated by the other three methods are graphed against the Penman-Monteith PETs in (b) to (d). Results show that the Thornthwaite method underestimates the PET significantly with values less than 50% of that of the Penman-Monteith PET estimate, while the other two are much closer. On the other hand, all of them show good correlation to the Penman-Monteith PETs, with the correlation coefficient ranging from 0.93 to 0.99. The estimates of Penman-Monteith method are adopted in the following application of SWAT and TOPMODEL.

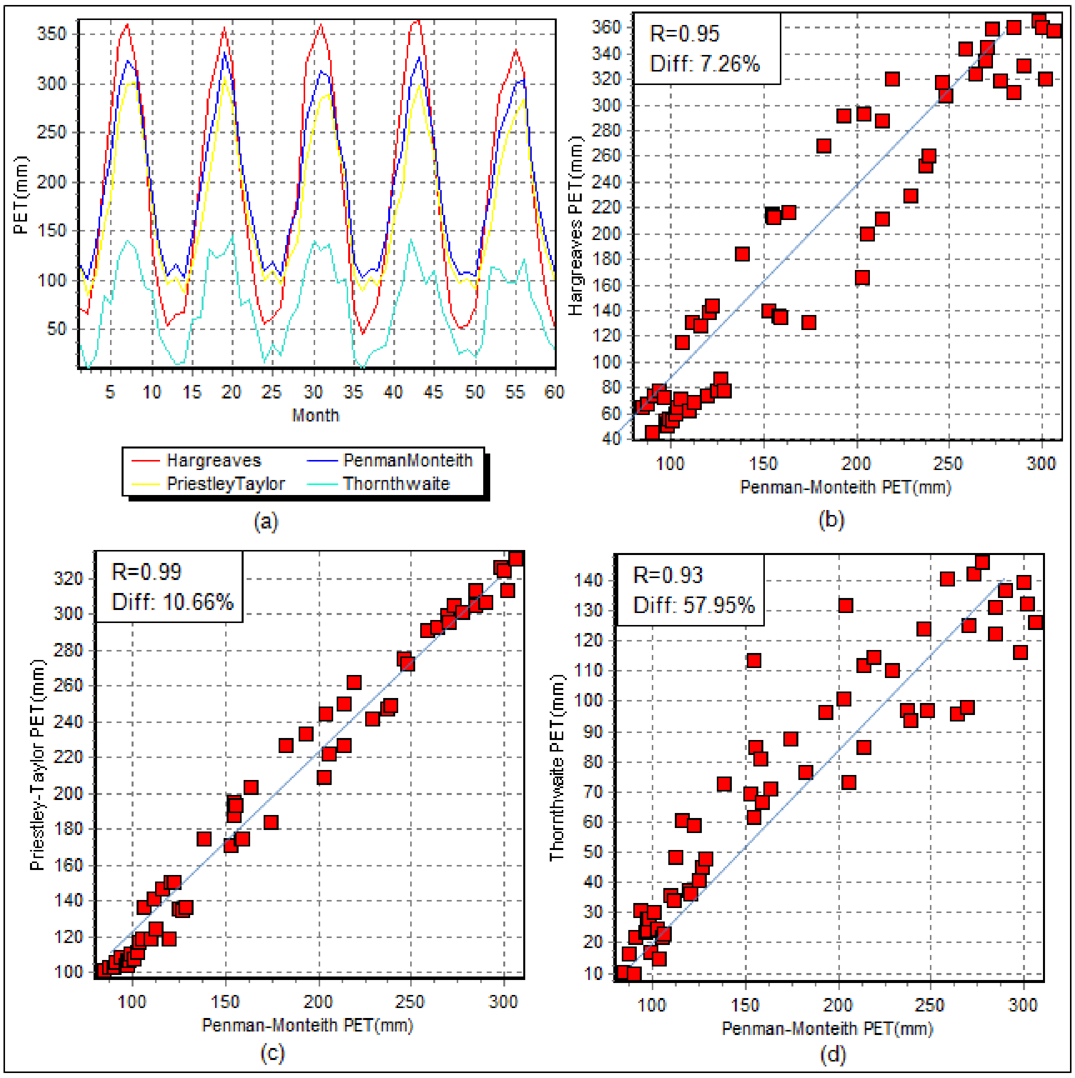


Figure 4.11 Results of monthly PET analysis: (a) Monthly PET sequences against time (b-d) Scatter plots of Hargreaves PET,Priestley-Taylor PET and Thornthwaite PET against Penman-Monteith PET respectively.

4.4.2 SWAT Application

4.4.2.1 Workflow Setup

The SWAT model is a physically based, semi-distributed watershed model that can be applied to predict the impact of land management practices on water, sediment, and agricultural chemical transport (Gassman et al., 2007). As mentioned before the HCMS only incorporates its hydrology components for which we have developed activities that mimic: snow melt, surface runoff, soil water and lateral flow within the unsaturated zone, groundwater flow in the saturated zone, and channel flow at a daily interval. Surface runoff is computed using the modified SCS curve number method, and the variable-storage-routing method is used to compute water transport through stream networks (Neitsch et al., 2005). The lateral flow and groundwater flow are routed to the channel using the lag-time method.

The workflow is set up by daisy-chaining these activities and piping the outputs of one activity to the inputs of the next activity as shown in Figure 4.12. Note that only one-directional data transport takes place between two activities, i.e. there is no feedback along the temporal axis. The external data inputs of this workflow include PET computed by the Penman-Monteith method, precipitation as downloaded from NLDAS, feature data of sub-basins, streams, and then the HRUs. We have incorporated an interactive interface to each activity which allows users to modify parameters and reset initial conditions. These are recorded in an XML file that is automatically created in the activity, which can then be reloaded on the next run-time.

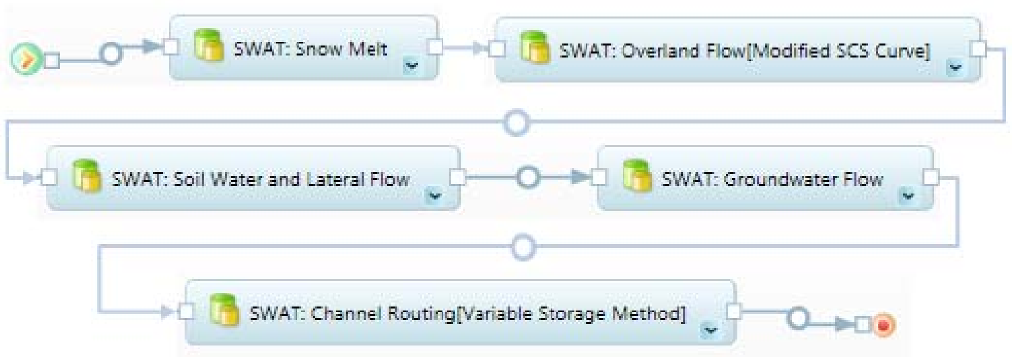
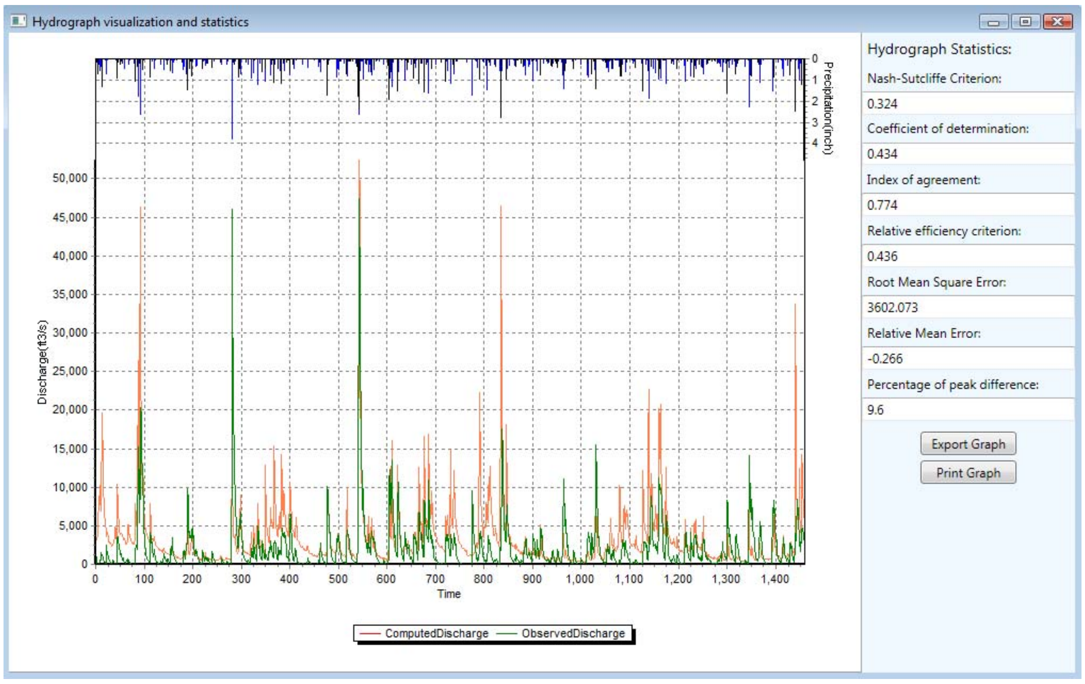

SWAT: Snow Melt
SWAT: Overland Flow[Modified SCS Curve]
SWAT: Soil Water and Lateral Flow
SWAT: Groundwater Flow
SWAT: Channel Routing[Variable Storage Method]

Figure 4.12 The workflow of hydrologic simulation with SWAT activities

4.4.2.2 Results and post-analysis

The workflow performs 2 simulations over a period of 4 years (2005-2008) for the Schuylkill River catchment with a 7 and 33 sub-basin division, respectively. The simulated and observed daily stream flows at the Schuylkill station (USGS 01474500) are entered into the result analysis activity at the end of the workflow sequences. It displays hydrographs and computes seven performance coefficients including such popular coefficients such the Nash-Sutcliffe efficiency, the root mean square error, the relative efficiency criterion. The daily mean precipitation for the catchment is computed by means of weighted averaging individual sub-basin precipitation values which are then imported to the results activity and displayed on the graphing panel as well.

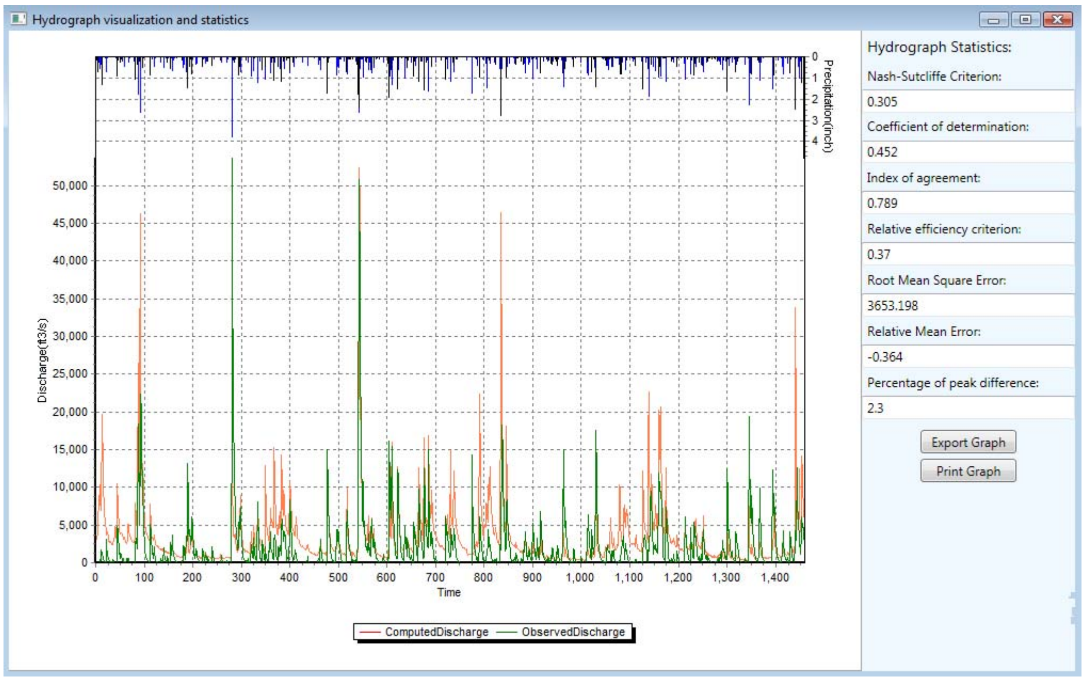
While it is not really crucial for the intent of this paper for completeness sake, Figures 4.13 and 4.14 show the hydrographs of the 7 and 33 sub-basins simulations, respectively. As is quite obvious the simulated hydrographs in both cases do not fit well and also there are no significant differences between the 7- and 33-sub-basin simulations as is evidenced by the performance coefficients listed. While the finer subdivision does affect the run-time efficiency it does not provide more accurate estimates of the hydrograph.



Hydrograph Statistics:

Nash-Sutcliffe Criterion:	0.324
Coefficient of determination:	0.434
Index of agreement:	0.774
Relative efficiency criterion:	0.436
Root Mean Square Error:	3602.073
Relative Mean Error:	-0.266
Percentage of peak difference:	9.6

Figure 4.13 Results of 4-year simulation by using 7-sub-basin watershed



Hydrograph Statistics:

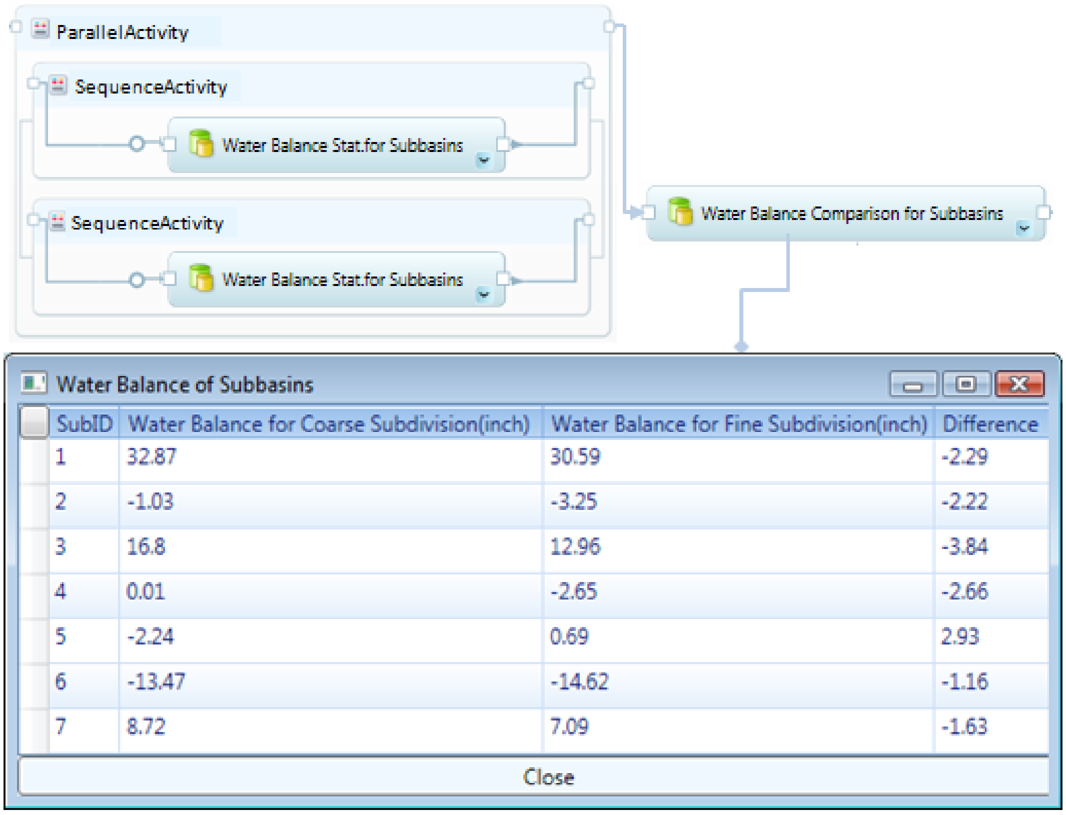
Nash-Sutcliffe Criterion:	0.305
Coefficient of determination:	0.452
Index of agreement:	0.789
Relative efficiency criterion:	0.37
Root Mean Square Error:	3653.198
Relative Mean Error:	-0.364
Percentage of peak difference:	2.3

Figure 4.14 Results of 4-year simulation by using 33-sub-basin watershed

An important hydrologic objective of any simulation is the desire to close the water balance, i.e. to account for all quantities of water that move either vertically or laterally in and out of a pre-specified control volume. This seemingly trivial objective of mass conservation is not easily answered however in many instances, because the accurate estimation of fluxes and quantities in addition to identifying the proper pathways are both quite complex and numerous. Closing the water balance of the watershed involves estimates of precipitation, flow at the outlet, actual evapotranspiration, and change in soil water content for which we have created a water-balance activity. Results of the water balance calculations for the 7- and 33-sub-basin watershed cases are given in Table 4.3, where the 'Difference' is computed from the precipitation minus the sum of the other quantities. For both cases, the input precipitation and the total sum of hydrologic components actually come out to be a fairly good match. We have also created an activity that performs water balance calculations for each of the sub-basins within the catchment that requires river discharge as an additional input and which can be run in parallel. Since a sub-basin within a catchment is always the aggregation of even smaller sub-basins within the larger sub-basin, comparisons of water balance calculations can be made between the 7- and 33- sub-basin approaches as shown in the workflow sequence in Figure 4.15. The water balance accumulated from that of finer sub-basins is close to the one of corresponding coarse sub-basin, with a difference ranging from 1.16 to 3.84 inch.

Table 4.3 Water balance of Schuylkill watershed for 4 years (2005-2008)

	Precipitation	Evaporation	Runoff	ΔSoil Water	Difference
7-Sub-basin(in)	193.7	186.2	5.05	-0.05	2.5
33-Sub-basin(in)	193.7	189.08	4.88	-0.05	-0.21


ParallelActivity
SequenceActivity
Water Balance Stat.for Subbasins
SequenceActivity
Water Balance Stat.for Subbasins
Water Balance Comparison for Subbasins
Water Balance of Subbasins
Close

SubID	Water Balance for Coarse Subdivision(inch)	Water Balance for Fine Subdivision(inch)	Difference
1	32.87	30.59	-2.29
2	-1.03	-3.25	-2.22
3	16.8	12.96	-3.84
4	0.01	-2.65	-2.66
5	-2.24	0.69	2.93
6	-13.47	-14.62	-1.16
7	8.72	7.09	-1.63

Figure 4.15 Comparisons of water balance for sub-basins between coarse and fine subdivisions

4.4.3 Application of TOPMODEL and a loosely coupled model

The previous SWAT workflow application was employed for a long-term daily time-step hydrologic simulation. In contrast, for this section we will simulate a short term event, i.e. an hourly rainfall-runoff simulation, for which we deploy the TOPMODEL and a loosely-coupled hydrologic model workflows using the 7-sub-basin Schuylkill River catchment. TOPMODEL utilizes a topographic index that depends on features such as local slope, upslope contributing area and downslope contour length to highlight the hydrologically significant areas within a catchment (Beven and Kirby,1979). Consequently, the TOPMODEL workflow starts from computing the topographic index using DEM data (pit-filled and sub-basin information) and then categorizing them into a specified number of groups. In parallel, the workflow creates the area-distance histograms for sub-basins based on flow direction (also gleaned from the DEM data), which are used to rout flow through channels. Required parameter inputs are displayed at the beginning of the workflow, which allows users to either accept default or to input custom values. Outputs from these three activities are then piped into the TOPMODEL activity to perform the desired hydrologic simulation, the workflow sequence of which is shown in Figure 4.16. Other model inputs include precipitation and potential evapotranspiration, which are either directly obtained from NLDAS (precipitation) or computed indirectly using the Penman-Monteith method (PET), with data needs supplied by the other NLDAS fields. Both data sets are computed with hourly intervals and are inputted into the TOPMODEL activity as driving forces.

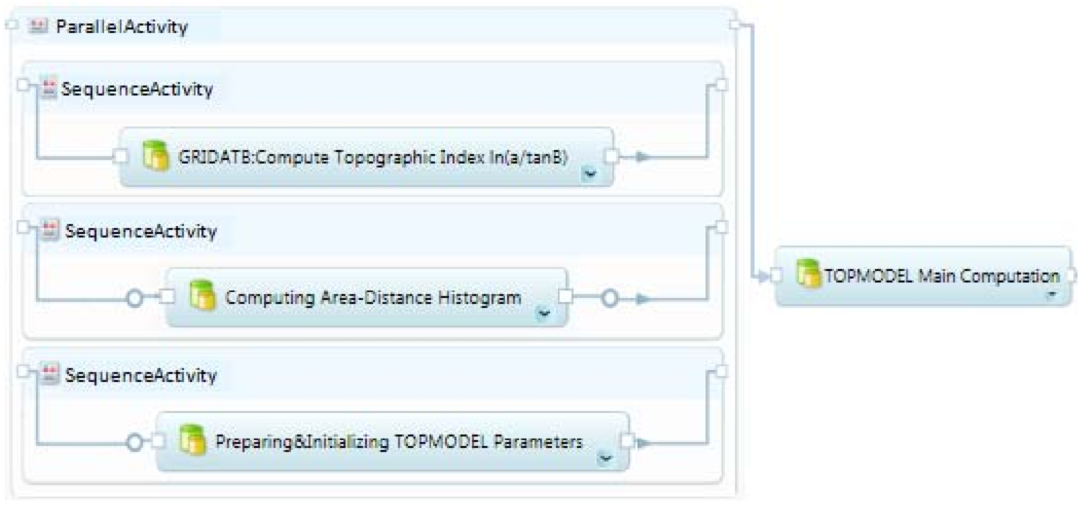

ParallelActivity
SequenceActivity
GRIDATB:Compute Topographic Index ln(a/tanB)
SequenceActivity
Computing Area-Distance Histogram
SequenceActivity
Preparing&Initializing TOPMODEL Parameters
TOPMODEL Main Computation

Figure 4.16 The workflow of hydrologic simulation with TOPMODEL activities

In addition to TOPMODEL we also use a loosely-coupled hydrologic model that adopts the SCS Curve Number for runoff yield, the SCS Unit Hydrograph method for direct runoff routing, the Recession method for baseflow, and the Muskingum method for channel routing all of which provide an alternative computational kernel for this short term event computation and the workflow of which is depicted in Figure 4.17. Evaporation is not considered in the simulation thus hourly precipitation is the only driving force. The sub-basin and river feature data obtained from the DEM process are other important model inputs. Parameters associated with each hydrologic method can be inputted or modified via an interactive window displayed whenever an activity is called in the workflow.

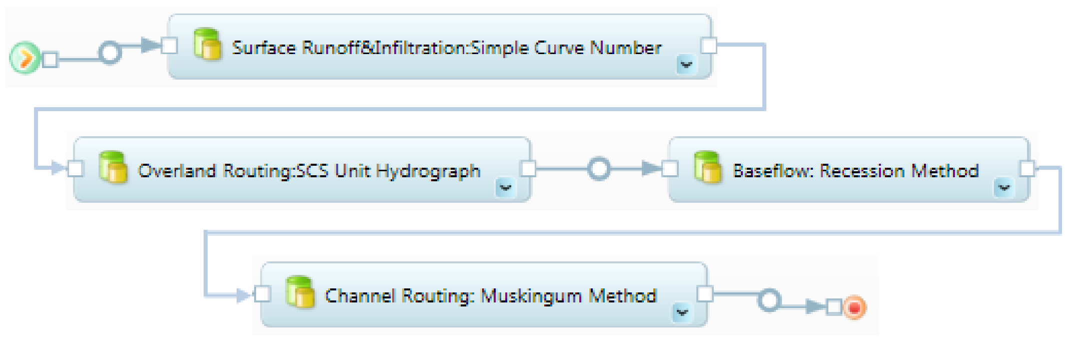

Surface Runoff&Infiltration:Simple Curve Number
Overland Routing:SCS Unit Hydrograph
Baseflow: Recession Method
Channel Routing: Muskingum Method

Figure 4.17 The workflow of a loosely coupled hydrologic model

Both workflows are applied to simulate a severe rainstorm (and subsequent flood) event that occurred from October 8th -11th, 2005. The observed hydrograph (USGS) alongside the simulated one from each of the two workflows are shown in Figure 4.18. Incidentally, the simulation of the loosely couple model is quite accurate and shows good agreement with the observed hydrograph; for this case the Nash-Sutcliffe coefficient is 0.95 and the Root Square Mean Errors (RSME) is about 4210 ft^3/s. However, the simulation of the TOPMODEL is far less accurate, with a Nash-Sutcliffe coefficient of 0.68 and a RSME of 10500 ft3/s. It indicates that the increased complexity of model structure and methodologies may not lead to the improved simulation results.

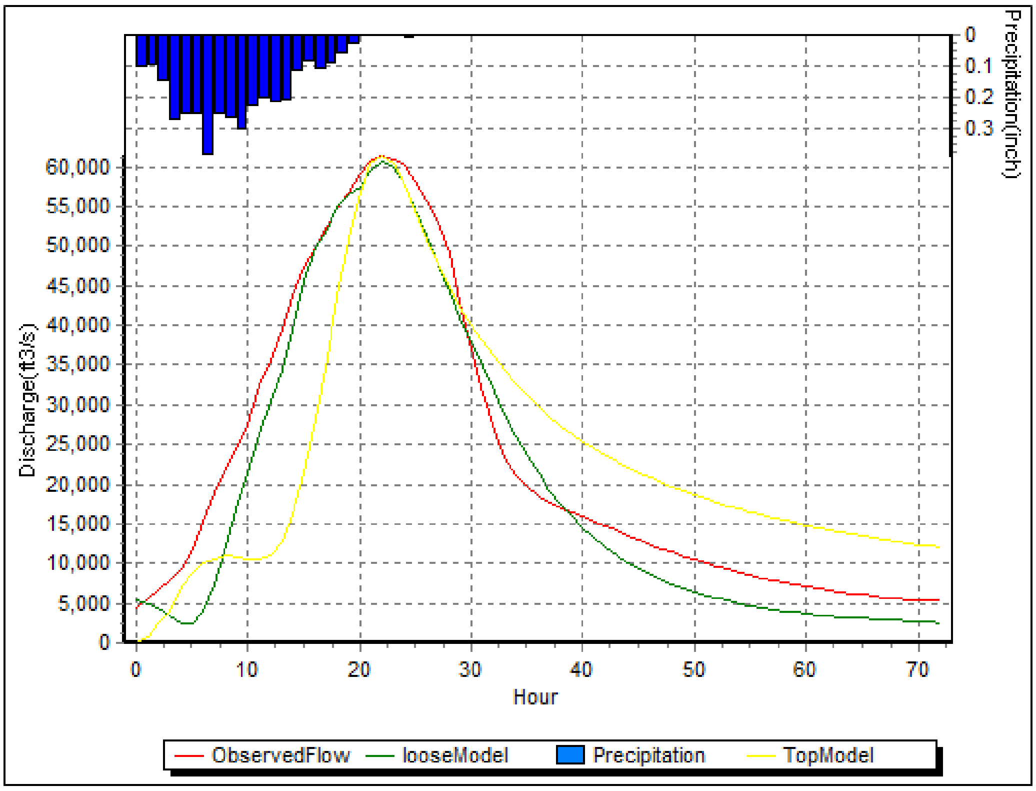


Figure 4.18 Comparisons of simulated hydrographs of TOPMODEL and the loosely coupled model with the observed hydrograph

4.5 Discussion

4.5.1 Applicability of HCMS

4.5.1.1 Data availability and workflow sequencing

The key objective of this study is to evaluate the utility of the workflow-based Hydrologic Community Modeling System for which we carried out some typical use cases in addition to recording several performance measures. Before detailing some of the pros and cons, in general the author’s experience has been that it is remarkably straightforward to build up workflows in the HCMS for hydrologic modeling purposes in addition to saving time and effort through the automated execution that the workflow sequences afford. Especially the preparation of model input data, which is often quite time consuming, can be made much easier when embedding the necessary steps into a workflow. Note that this is particularly important in the context of wanting a system that is applicable anywhere in the US and as such is in need of data with nationwide coverage. In this regard, the two precipitation data sources (NLDAS at 13.8x13.8 km spatial and hourly temporal resolutions) and the NOAA/NWS MPE data (4x4 km spatial and daily temporal resolutions) are excellent data sources, with NLDAS providing additional meteorological data relevant to hydrologic modeling tasks. There are more data nationwide data sources that could be tapped into however; for the example the National Climatic Data Center, NCDC, holds the world largest meteorological data set some of which is accessible via CUAHSI's HIS-Central WaterOneFlow web services. Other climate data includes MODIS, or DAYMET, in addition to data products published though NOAA's National Weather Service. An entire different suite of data is offered up by NASA's Distributed Active Archive Centers (DAACs) of which the NSDIC (cryosphere, snow), LP (land processes), and GHRC (global hydrology) centers are probably the most relevant to the HCMS. While this list is not at all comprehensive it serves to illustrate the degree of which data is available that could be used for consumption in the HCMS.

Another convenient feature of the HCMS is its capability of processing raw data, such as the DEM data which is one of the fundamental steps of model data pre-processing. The HCMS allows access to different compute environments, i.e. either via activities embedded on the server inside the workflow, or via calling up web services that execute the steps on a remote server, or by sourcing out computations to the AZURE cloud (note that we have not tested the latter). The workflows can be organized such that certain computations are only carried out once, the results of which can then be stored and recalled later (also via activities) so they can be further processed in any type of computation desired either in sequence or in many parallel strands. A good example for this type of pre-processing and using-it-later approach are the Hydrologic Response Unit, HRU, calculations (which are very time consuming) which can then be used by SWAT in many different run time scenarios. The data needs for these computations are extensive (soil and land cover) but can be accessed at a nationwide level as well; Soil Survey Geographic (SSURGO) data while not straightforward to handle is accessible from the US Department of Agriculture National Resources Conservation Service as is the National Land Cover Data (NLCD) from USGS. In short, the spatial applicability of HCMS is well supported because many data sources have a) nationwide coverage, and b) allow programmatic access to their data holdings using web technology.

There are however a number of shortcomings that the HCMS exhibits, some of which are due to the nature of a workflow engine environment while others are more of external origin. While some effort can be expended to acquire and build an extensive post-processing library of activities, it is clear that a workflow engine is not an alternative to a full-fledged GIS environment such as ESRI’s ARCInfo, GRASS, or MapWindows. This is largely due to the fact that workflows are designed to be hit-button-and-forget operations without much GUI interaction (even though this can be done in a limited fashion) while a GIS environment lives from its ability to respond to user interactions while conducting spatial data analyses.

Another downside consequence is that while the ability to access a wide variety of data stores, each activity that does so must be customized so it can handle the syntactic and semantic specificities of a data source. Some efforts are underway to reduce this rampant heterogeneity in the data world, however, for the foreseeable future this will remain the status quo. As a result, adding functionality to the HCMS does require some coding work and in addition to understanding the constraints of the TRIDENT (or any other) environment. In this vein, the data access points are not uniform in their set up (some use FTP protocols, others use web service calls, yet others provide links to EXCEL spreadsheets) in addition to having access restrictions based on user origin (authentication) or simply based on bandwidth limitations on the data server side. The latter has implications on how easy it is to download the data needed for a specific catchment versus having to divert to an alternative albeit more cumbersome access point to get the data needed. These too may be improved in the future, however, at this point in time the HCMS must take some detours losing a little of its straightforwardness when accessing data sources.

4.5.1.2 Hydrologic models and their portability into HCMS

A second assessing metric is the ability of a HMCS to host different numerical models and approaches to simulating hydrologic processes. The authors acknowledge the fact that there are many models of various complexities available that provide a large pool to chose from and also realize that not all of them could be brought into the HCMS fold. Hence, we selected three numerical modeling approaches to elucidate how difficult it is to port legacy codes into the HCMS, to access them remotely via web service calls, and what it would take to build a flexible swap-in-swap-out numerical kernel. Our experience shows that it is possible to incorporate different models of various complexities into HCMS even using different compute access points, i.e. as an activity or remotely. There are limits to this however; we attempted to dis-aggregate the PennState Integrated Hydrologic Model (PIHM) into the workflow which proved an unsuccessful attempt due to the complexity of the model composition, i.e. its internal data structure as well as the degree of coupled equations that need to be solved simultaneously rather than in sequence or parallel (without data communication needs in these parallel strands). It might have been possible to access PIHM remotely as a wrapped web service call but that would have left the need for pre- and post-processing of PIHM I/O data into digestible files that would need to be moved and copied into specific directories remotely on to some server, which seemed too onerous a task.

Developing independent hydrologic process modeling components on the other side is ideal for the HCMS. The workflow composer is a convenient tool to plug-and-play components and building model execution chains that can be sub chains, full data processing chains, modeling chains, post-processing chains that can be linked together or stay apart to whatever degree of steps are desired for any given modeling task at hand. For example, PET is a hydrologic quantity that can be estimated via numerous approaches, we have 4 different alternatives in the hydrologic library, be tested and compared, and swapped in/out whenever needed or not. We found the effort to be quite manageable when encoding new process alternatives especially when the accompanying theoretical and implementation demands are reasonably short. With the underlying data structure it also fairly easy to make sure that activities “understand’ each other, i.e. the syntactic and semantic requirements for successful communications are being met by each of the activities. We have also found that the best strategy when composing workflows is to use short workflow assemblies first, especially for those tasks that are time consuming, such as doing the sink-fill and flow direction computations when processing the DEM. Since these are computations that only need to be carried out once, they lend themselves to be parceled out and just using the resulting data sets repeatedly as input for the actual modeling runs.

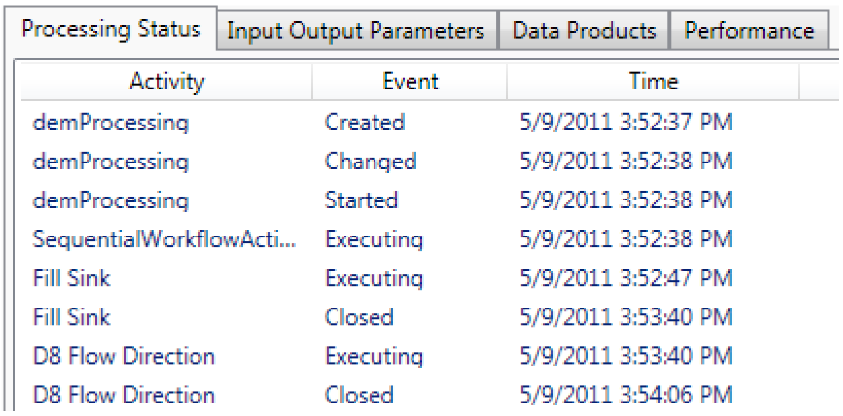
We also found it much easier to engage the more complex SWAT model within the HCMS as compared to the standalone version, because of the largely automated input data processing with some flexibility when assigning either default or custom values to parameters that need to be set at run-time. For example, the standalone SWAT version requires the preparation of a configuration file (with a strict format) that describes the configuration of model computations, the locations of model driving forces, parameter values and a schema for channel routing. Specifications such as the outflow of stream need to be added to the outflow of stream B the result of which then enters stream C as one of its inflows must be explicitly stated for each stream segment in the modeling domain. In the HCMS all of this can be avoided through an automation that is embedded in the SWAT data preparation activity.

One shortcoming of the HCMS is the lack of an automated parameter estimation segment, which would bring the modeling effort one step closer to a true hit-a-button-and-walk-away operation. Parameters within SWAT and TOPMODEL still need to be determined by a trial-and-error approach that is controlled manually. The SWAT standalone version offers this feature and we hope to be able to migrate an automated parameter estimation environment to HCMS in the future to fill this gap. Last but not least a user must realize the strength and also recognize the weaknesses of a workflow engine. In general one can state that a workflow engine shines when it comes to repetitive executions of the same procedure over and again. The utility of the system starts to gradually decline the more variations are added that seek to test out small changes here and there. As long as the computations still involve repetitive steps this is probably acceptable, but it will diminish the more the user seeks to interact with the workflows. Clearly, the ability to record provenance in these cases is a huge advantage especially when need arises to redo model runs, in addition to being able to share “successful” or interesting workflows with others using sites such as MyExperiment.org.

4.5.2 Performance of HCMS

The second important aspect of whether the HCMS is a viable platform for community modeling purposes concerns its run-time performance. The question is not so much if the HCMS is able to achieve significant speed improvement during run time, but rather what the computational costs and speed losses are when adding a middle-layer such as a workflow engine to the modeling effort. We should say that we have not tested all possibilities afforded, especially not the option of pushing compute intensive application out to the AZURE cloud, or utilizing any other parallel code implementation (tauDEM for example could be run as parallel code). Rather we have restricted ourselves to a relatively common set up in which we work with 2 or 3 ordinary DELL servers, a set up we assume to be more widespread among hydrologic modellers.

One aspect that stands out is the fact that the workflow engine requires some overhead time to get instantiated and ready to execute workflows. Instead of performing the computations embedded in the workflow immediately, we found that the engine needs about 15-40 seconds to create an execution scenario before proceeding to work through the workflow activities. One convenient feature that TRIDENT offers is that the processing status is monitored and that CPU performance can be traced during execution. Figure 4.19 shows an example of recorded processing status; notice that the initiation time consumed by creating the scenario is not recorded here. If one assumes that the typical patience threshold of an online user is about 6 seconds, then 15-40 seconds wait time must be considered a burden. However, what is considered a burden may be relative when compared to the overall run time. A program needing several hours to execute will not suffer from a few extra seconds for initiation, while a program needing 30 seconds doubles up in wait time; hence these extra seconds are either insignificant or significant from a percentage point of view, in reality should not play much of role however. For example, the processing status shown in Figure 4.19 is recorded for processing a small DEM (1176*883), which takes 313 seconds overall with 24 seconds for creating the scenario. The standalone program takes around 92% of this time; the difference is not significant, almost non-existent when considering the 14 hours it takes to process the DEM for the Schuylkill River catchment.



Activity	Event	Time
demProcessing	Created	5/9/2011 3:52:37 PM
demProcessing	Changed	5/9/2011 3:52:38 PM
demProcessing	Started	5/9/2011 3:52:38 PM
SequentialWorkflowActi...	Executing	5/9/2011 3:52:38 PM
Fill Sink	Executing	5/9/2011 3:52:47 PM
Fill Sink	Closed	5/9/2011 3:53:40 PM
D8 Flow Direction	Executing	5/9/2011 3:53:40 PM
D8 Flow Direction	Closed	5/9/2011 3:54:06 PM

Figure 4.19 A sample of recorded processing status

Another source of execution time loss is the activation of interactive components in an activity. Workflows containing interactive activities add some additional lag time as the workflow engine takes 5-15 seconds to start up an interactive window. This too is beyond the 6 second “patience” threshold and while not really substantial could be viewed to be somewhat annoying, especially in those workflows that require several of the interactive windows to appear. For example, the workflow for SWAT contains five activities which call up an interactive window for loading or assigning parameters; hence this adds about 25 – 75 seconds to the execution. Given an overall run time of about 15 seconds for the standalone SWAT program performing the 4-year daily runoff simulation for the 7-sub-basin catchment, this more than quadruples the execution time. In terms of percentage calculations this 400% increase seems unacceptable, even though this could be mitigated somewhat if SWAT where to be transferred into a single activity (no swapping modules possible in this case however).

While the previous paragraphs show that run-time efficiency of a model kernel is adversely affected when being migrated into the HCMS it is also important to realize that time saved preparing model inputs makes these additional seconds or minutes insignificant. While it is difficult to exactly quantify the time saved during the data preparation steps our impression that one saves hours on these tasks versus spending seconds (during execution) should be pretty accurate. One should also take notice that not all processes, activities, and steps take the same time, in fact they can vastly differ. In the examples presented processing DEM and NLDAS-2 meteorological data as well as constructing HRUs were by far the most time-consuming tasks. Also, using web service based versions, such as the DEM processing, can yield improvement as the computations are performed on a remote server in parallel with other sub-workflows executed at the local machine thus harnessing additional CPU times elsewhere. In general, the authors feel that the overall time savings using the HCMS are significant enough to put a check mark on this performance criterion; losses due to additional middleware are more than adequately compensated for when processing model data.

4.6 Summary and Future work

In this paper, we tried to assess the utility of a workflow (MicroSoft's TRIDENT) shelled Hydrologic Community Modeling System, HCMS, by carrying out several typical hydrologic modeling studies and subsequently recording our experiences in addition to computing some performance measures. Key aspects to consider for the degree of HCMS utility were the potential and ability of the system to be used anywhere in the nation which to a great extent translates into being able to access relevant data on a nationwide scale. A second key aspect was to measure the computational costs of using a middle ware layer such as a workflow engine in terms of performance and degradation in run- and turn-around time for a modeling task when compared to the usual approach, i.e. stand alone codes and manual preparation of I/O data. To this end we selected a demonstration watershed of considerable size, the Schuylkill River catchment in southeast Pennsylvania having a drainage area of about 5,000 square-kilometers, for both long-term (4 years) and short term (several days) rainfall-runoff simulations that were compared to USGS stream gauge station data. For these tests we used different sub-divisions of the watershed, compared similar data products from two different data sources, in addition to deploying different model formulations and also running several comparisons between different hydrologic process formulations, such as for potential evapotranspiration. This ensured that many of our developed workflow activities, organized in 4 libraries (data-access, data preparation, modeling, post-processing), were put through their paces while at the same time providing the base for our assessments.

We found that the HCMS can be used quite well as a modeling platform because data is available with good spatial and temporal coverage regardless the location in the US. This already pretty good situation is bound to improve even further as more mission agencies improve their access points to their data holdings, be it through web service connections, better data preparation, larger server bandwidth, or through harmonizing efforts that brings more interoperability to the data world. This is a key strength of the HCMS, that it is able to access data via a wide range of access trajectories be it remote (or local) files systems, database connections, or web services. Building activities that can accomplish these connections while not trivial is reasonably easy, hence the HCMS exhibits a good deal of growth potential as more data sources can be added to the data access library.

We also found that while not computational cost free, the middle ware layer that a workflow engine represents between modeling and the user, is computationally “affordable”. Added time demands are in the range of seconds or a few minutes, that, depending on the application, represent insignificant additions to otherwise lengthy (hours) computations, especially in the data preparation arena. During model execution, often just taking seconds to minutes, lag times due to workflow initiation and preparing interactive window launching adds to the time demands often doubling (or more) execution or turn-around times. These costs while unavoidable do not seem to be too detrimental though given the time savings one gets when using the automated data preparation features of HCMS. Also, there is the potential to utilize dedicated servers (more powerful than the ones we used) that could reduce the execution times particularly those requiring hours.

There are a number of tasks that could be carried out in the future that would help to improve and mature the HCMS. Computational efficiency could be improved greatly if some of the data preparation steps were available for a parallel compute environment, or if the computation of the time-demanding data sets cold be pushed into a high performance computing environment; the option of trying the AZURE cloud being the most obvious one. For example, there is a parallel version of tauDEM available now which could be utilized instead of the single processor version we use for HCMS.

While we have a web service based version for the DEM processor, we would also like to add web services versions of the HRU constructor and NLDAS-2 data processor those two (in addition to the DEM processor) being the most time demanding computations in the data preparation segment. We could also expand this idea and develop web-based versions of all hydrologic compute kernels in the HCMS thus allowing a powerful server to serve as the host instead of running these computations on the client machine.

Future work will also need to focus on the inclusion of some of the auxiliary functions (or activities) in the HCMS. An automated parameter identification module would greatly help the modeling sequences in the calibration steps. There is also the need to build out the post-processing library to include more visualization and statistical analyses options both of which are somewhat limited in the current implementation. In this vein we would also to like to increase the data access library as more data sources are already available that could be tapped into. Lastly, in this paper we only used one catchment for demonstration purposes, it would helpful to actually carry out a study that uses a slate of 4 or 5 different seized watersheds to see how well the HCMS “scales” to different watershed sizes.

CHAPTER 5: SUMMARY AND FUTURE WORK

In this thesis, we presented three papers regarding to the development of a hydrologic community modeling system. We first took a review of current efforts in building community modeling systems within the hydrologic related realms, and investigated the potentials of coupling frameworks to serve as the backbone of community modeling systems. The general advantages of using existing coupling frameworks lie in their capabilities of incorporating new modules or wrapping legacy codes with much ease, mediating the execution and communication between modules seamlessly, and supporting the construction of more complex integrated models with compliant modules etc. However, the obstacles also stayed in the way, for example, their practical usages might involve a steep learning curve, and migrating a piece of legacy codes with complex structure and data model while lacking of code documentations could be extremely difficult. Further there could be some efficiency loss for a legacy model when being replanted to new environments. Of concern these challenges, we embark on our studies presented in the second paper.

In the second paper, we made attempts to develop a hydrologic community modeling system based on the TRIDENT workflow system which is functionally analogical to coupling frameworks but also incorporates additional features such as recording provenance and distributing computations over network etc. Our objective was to provide such an environment supporting the construction of variable hydrologic modeling systems that channel model pre-processing, model and model post-processing through workflow sequences. As essential components of this environment, we presented our development of four libraries responsible for accessing data from online data repositories or local data files, preparing source data into model inputs, performing hydrologic computations, and analyzing modeling results respectively. Each library consists of a number of activities either migrated from legacy models/codes or scripted from the ground up. The environment was demonstrated by composing a seamless workflow for hydrologic simulation and applying it to a small watershed in Southeastern Pennsylvania. To better test this environment and evaluate its performance, we conducted more case studies in the third paper.

The HCMS was applied to the Schuylkill watershed located in the Southeastern Pennsylvania mainly for long-term or short term rainfall-runoff simulations in the third paper. We demonstrated its capabilities of accessing and comparing data from different data resources, performing DEM processing using different types of workflows and under different delineation schemes, estimating single hydrological quantity (potential evapotranspiration) with swappable activities encapsulating various mathematical methods, simulating entire hydrologic simulation with different models (the migrated SWAT, TOPMODEL and a loosely coupled hydrologic model), directly transporting results to the post-analysis activity for evaluation and comparison. These case studies confirm the feasibility of the environment, and detect limited run-time performance loss ranging from a few seconds to a minute more for workflows executed in local machine. This loss can hardly be noticed for workflows involving heavy computations, and also can be compensated by the time saved from the step of data preparations.

Our future work will focus on the extension of the current four libraries to power up the modeling environment. We will enrich the data access library with accesses to an increased number of data sources, provide alternative activities for data processing such as data interpolation or spatial data downscaling, migrate more existing hydrologic models into model library, and expand the post-processing capabilities. Typical activities such as the one for automatic parameterization as well as web service based activities for HRU construction and NLDAS-2 data processing will be created in the first place.

LIST OF REFERENCES

1. Abbott, M.B., Vojinovic, Z., 2009. Applications of Numerical Modelling in HydroInformatics. Journal of HydroInformatics, 11(2-4), pp. 308-319, IWA Publishing, doi:10.2166/hydro.2009.051.

2. Abel, D.J., Davis, P.J., and Davis, J.R., 1994. The systems integration problem. International Journal of Geographical Information Systems, 8(1), pp.1-12.

3. Argent, R.M., 2004. An overview of model integration for environmental applications-components, frameworks and semantics. Environmental Modeling & Software, 19, pp.219-234.

4. Argent, R.M. and Rizzoli, A.E., 2004. Development of multi-framework model components. In: Transactions of the Second Biennial Meeting of the International Environmental Modelling and Software Society, Osnabrück, Germany, 14-17th June, pp.365-370.

5. Arnold, J.G., Allen, P.M. and Bernhardt, G., 1993. A comprehensive surface-groundwater flow model. Journal of Hydrology, 142, pp.47-69.

6. Arnold, J.G., Srinivasan, R., Muttiah, R.S., Williams, J.R., 1998. Large area hydrologic modeling and assessment part I: model development. Journal of American Water Resources Association. 34(1), pp.73-89.

7. Bari, M.A., Shakya, D.M. and Owens, M., 2009. LUCICAT Live A modeling framework for predicting catchment management options. In: 18th World IMACS/MODSIM Congress, Cairns, Australia, 13-17th July, pp.3457-3463.

8. Bernholdt, D.E., Allan, B.A., Armstrong, R., Bertrand, F., Chiu, K., Dahlgren, T.L., Damevski, K., Elwasif, W.R., Epperly, T.G.W., Govindaraju, M., Katz, D.S., Kohl, J.A., Krishnan, M., Kumfert, G., Larson, J.W., Lefantzi, S., Lewis, M.J., Malony, A.D., McInnes, L.C., Nieplocha, J., Norris, B., Parker, S.G., Ray, J., Shende, S., Windus, T.L. and Zhou, S., 2006. A component architecture for high-performance scientific computing, International Journal of High Performance Computing Applications, ACTS Collection Special Issue, 20(2), pp.163-202.

9. Beven, K. J., 2001. Rainfall-Runoff Modelling – The Primer, John Wiley& Sons Ltd., Chichester.

10. Beven, K. J. and Moore, I. D.,1992. Terrain Analysis and Distributed Modeling in Hydrology.John Wiley and Sons, New York.

11. Beven, K. J. and Kirkby, M. J., 1979. A physically based variable contributing area model of basin hydrology. Hydrologic Sciences Bulletin, 24(1), pp.43-69.

12. Beyazgul, M., Kayam, Y., and Engelsman, F., 2000. Estimation methods for crop water requirements in the Gediz Basin of Western Turkey. Journal of Hydrology, 229 (1-2), pp.19-26.

13. Biesecker, J.E., Lescinsky, J.B. and Wood, C.R.,1968. Water Resources of the Schuylkill River Basin. Water Resources Bulletin No. 3. Pennsylvania Department of Forests and Waters, Harrisburg, PA.

14. Bhatt, G., Kumar, M. and Duffy, C. J., 2008. Bridging Gap Between Geohydrologic Data and Integrated Hydrologic Model: PIHMgis, In: Sanchez-Marre, M., Bejar, J., Comas, J., Rizzoli, A., Guariso G.(eds). International Environmental Modelling and Software Society (iEMSs), Barcelona, Catalonia, July 7-10, 2008, pp.743-750.

15. Bo, L., and Piasecki, M., 2011, A Review of Efforts to establish a Hydrologic Community Modelling System, submitted to Journal of HydroInformatics, April 2011. In review.

16. Bo, L., and Piasecki, M., 2011, Development of a Hydrologic Commuity Modeling System (HCMS) using a Workflow Engine, submitted to Journal of HydroInformatics, May 2011. In review.

17. Bongartz, K., Flugel, W.A., Krause, P. and Taddei, U., 2003. Geoinformatics toolset for integrated river basin management(IRBM) of the Saale river, Thuringia, Germany. In: International Congress on Modelling and Simulation(MODSIM), Townsville, Australia, 14-17th July, 6pp.

18. Bouraoui, F., Benabdallah, S., Jrad, A. and Bidoglio G., 2005. Application of the SWAT model on the Medjerda river basin (Tunisia). Physics and Chemistry of the Earth, 30, pp.497-507.

19. Brandmeyer, J.E. and Karimi, H.A., 2000. Coupling methodologies for environmental models. Environmental Modeling & Software, 15, pp.479-488.

20. Bulatewicz, T. F., 2006. Support for model coupling: An interface-based approach. Phd. Thesis. Ch2: Related Work. Department of Computer and Information Science, University of Oregon. pp. 7-19.

21. Campbell, A.P. and Hummel, J.R., 1998. The dynamic information architecture system: an advanced simulation framework for military and civilian applications. In: Advanced Simulation Technologies Conference,Boston,MA,US, 5-9th April, 6 pp.

22. Ciupke, O, Schmidt, R., 1996. Components as context-independent units of software. In: Special Issues in Object-Oriented Programming, Workshop Reader of the 10th European Conference on Object-Oriented Programming, Heidelberg, Germany, pp.139-143.

23. Collins, N., Theurich, G., DeLuca, C., Suarez, M., Trayanov, A., Balaji, V., Li, P., Yang, W., Hill, C., and da Silva, A., 2005. Design and Implementation of Components in the Earth System Modeling Framework. International Journal of HPC Applications, 19, pp.341-350.

24. Cosgrove, B., Lohmann, D., Mitchell, K., Houser, P., Wood, E., Schaake, J.,Robock, A., Marshall, C., Sheffield, J., Duan, Q., Luo, L., Higgins, R., Pinker, R., Tarpley, D. and Meng, J., 2003. Real-time and retrospective forcing in the North American Land Data Assimilation System (NLDAS) project. Journal of Geophysical Research, 108(D22), 8842, Doi:10.1029/2002JD003118, pp.13.

25. CRC, 2010. Chesapeake Bay Research Consortium, Community models and integrated observations for the Chesapeake Bay. Accessed October 2010 at http:// ches. communitymodeling.org /documentation /pdf/ ModelPreamble.PDF

26. CSIRO, 2010. Commonwealth Scientific and Industrial Research Organisation, Australia national science agency, accessed October 2010 at http://www.csiro.au/.

27. CUAHSI, 2010. Consortium of Universities for the Advancement of Hydrologic Science,Inc., accessed October 2010 at http://www.cuahsi.org.

28. Cubasch, U., Storch, H., Waszkewitz,J. and Zorita, E., 1996. Estimates of climate changes in southern Europe using different downscaling techniques. Climate Research, 7, pp.129-149.

29. Cuddy, S.M., Letcher, R.A. and Reed, M.B., 2002. Lean interfaces for integrated catchment management: rapid development using ICMS, In: Rizzoli, A. E. and Jakeman, A. J., (eds.), Integrated Assessment and Decision Support, Proceedings of the First Biennial Meeting of the International Environmental Modelling and Software Society, 3, pp.300-305.

30. Dahlgren, T., Epperly, T., Kumfert, G. and Leek, J., 2007. Babel User's Guide, March 23, 2007 edition, Center for Applied Scientific Computing, U.S. Dept. of Energy and University of California Lawrence Livermore National Laboratory, pp. 269.

31. Daly C., Neilson R. P. and Philips D. L., 1994. A statistical-topographic model for mapping climatological precipitation over mountainous terrain. Journal of Applied Meterology, 33, pp.140-158.

32. David, O., Markstrom, S.L., Rojas, K.W., Ahuja, L.R. and Schneider, I.W., 2002. The Object Modeling System. In: L.R.Ahuja, L. Ma, and T.A. Howell (Eds.),Agricultural System Models in Field Research and Technology Transfer, Lewis Publishers, New York, pp. 317-330.

33. David, O., Schneider, I.W. and Leavesley, G.H., 2004. Metadata and modeling frameworks: The Object Modeling System example. In: Proceedings of International Environmental Modelling and Software Society, Osnabrück, Germany, 14-17th June, pp.5.

34. Deelman, E., Singh, G., Su, M.H., Blythe, J., Gil,Y., Kesselman, C., Mehta, G., Vahi, K., Berriman, G., Good, J., Laity, A., Jacob, J.C. and Katz, D.S., 2005. Pegasus: A framework for mapping complex scientific workflows onto distributed systems, Scientific Programming Journal, 13 (3), pp.219-237.

35. Delanunay, B., 1934. Sur la sphere vide, Bull. Acad. Science USST VII: Class Sci. Mat. Nat., b, 793.

36. Dickinson, R.E., S.E. Zebiak, J.L. Anderson, M.L. Blackmon, C. DeLuca, T.F. Hogan, M. Iredell, M. Ji, R. Rood, M.J. Suarez and K.E. Taylor, 2002. How can we advance our weather and climate models as a community? Bulletin of the American Meteorological Society, 83(3), pp.431-434.

37. Di Luzio, M. and Arnold, J.G., 2004. Formulation of a hybrid calibration approach for a physically based distributed model with NEXRAD data input. Journal of Hydrology, 298, pp.136-154.

38. ESMF 2010, The Earth System Modeling Framework, Accessed October 2010 at http://www.earthsystemmodeling.org/.

39. Fekete, B.,M., Wollheim, W.M., Wisser, D. and Vorosmarty, C.J., 2009. Next generation framework for aquatic modeling of the Earth System, Geoscientific Model Development Discussions, 2, pp.279-307.

40. FitzHugh, T.W. and MacKay, D.S., 2000. Impacts of input parameter spatial aggregation on an agricultural nonpoint source pollution model. Journal of Hydrology, 236, pp.35-53.

41. Fleeger, G.M. 1999. The Geology of Pennsylvania's Groundwater. Third edition. Pennsylvania Geological Survey, 4th series, Educational Series 3, 34 pp.

42. Fortune, D., Gijsbers,P., Gregersen, J. and Moore, R., 2008. OpenMI-Real progress towards integrated modeling. In: Abrahart,R.J., See,L. M. , Solomatine, D. P. (eds.) Practical HydroInformatics: Computational Intelligence and Technological Developments in Water Applications, Springer Berlin Heidelberg, Ch32. pp.449-464.

43. Früh, B., Schipper, J.W., Pferiffer, A. and Wirth, V., 2006. A pragmatic approach for downscaling precipitation in alphine-scale complex terrain. Meteorologische Zeitschrift, 15(6), PP.631-646.

44. Fulp, T.J., Vickers, W. B., Williams, B. and King, D.L. 1995. Decision support for water resources management in the Colorado River regions. In: Ahuja L, Leppert J, Rojas K, Seely E (eds). Workshop on Computer Applications in Water Management, Colorado Water Resources Research Institute, Fort Collins, CO, Information Series No. 79, pp. 24-27.

45. Gamma, E., Helm, R., Johnson, R. and Vlissides, J., 1995. Design Patterns: elements of reusable object oriented software. Addison Wesley. Reading, Mass.

46. Gassman, P.W., Reyes, M.R., Green, C.H. and Arnold, J.G., 2007. The Soil and Water Assessment Tool: historical development, applications, and future research directions. Transactions of the ASABE, 50(4), pp.1211-1250.

47. Gesch, D., Oimoen, M., Greenlee, S., Nelson, C., Steuck, M. and Tyler, D., 2002. The National Elevation Dataset, Photogrammetric Engineering and Remote Sensing, 68, pp. 5-11.

48. Gil, Y., Deelman, E., Ellisman, M., Fahringer, T., Fox, G., Gannon, D., Goble, C., Livny, M., Moreau, L. and Myers, J.,2007. Examining the challenges of scientific workflows, IEEE Computer, 40(12), pp.24-32.

49. Goderis, Roure, D.D., Goble, C., Bhagat, J., Cruickshank, D., Fisher, P., Michaelides, D. and Tanoh, F., 2008. Discovering scientific workflows: the myExperiment benchmarks. IEEE Transaction and automation science and engineering (submitted), pp. 12.

50. Gregersen, J. B., Gijsbers, P. J. A., Westen, S. J. P. and Blind, M., 2005. OpenMI: the essential concepts and their implications for legacy. Software Advances in Geosciences, 4, pp.37-44.

51. Gregersen, J.B., Gijsbers, P.J.A., and Westen, S.J.P., 2007. OpenMI : Open Modelling Interface. Journal of Hydroinformatics , 9 (3), pp.175-191.

52. Gourbesville, P., 2009, Data and HydroInformatics: New Possibilities and Challenges, Journal of HydroInformatics, Vol. 11, No. 3-4, pp. 330-343, IWA Publishing, doi:10.2166/hydro.2009.143.

53. Hargreaves, G.H., 1975. Moisture availability and crop production. Trans. ASAE, 18, pp.980-984.

54. Hargreaves, G.H. and Samani, Z.A., 1985. Reference Crop Evapotranspiration From Temperature. Applied Engineering in Agriculture, 1(2), pp. 96-99.

55. HEC 2010, The Hydrologic Engineering Center of the US Army Corps of Engineers, Vicksburg, Mississippi, accessed October 2010 at http://www.hec.usace.army.mil/

56. Hewett, C.J.M., Doyle, A. and Quinn, P.F., 2010, Towards a hydroInformaics framework to aid decision-making for catchment management, Journal of HydroInformatics, 12(2), pp.119-139, IWA Publishing, doi:10.2166/ hydro.2009.022.

57. Hill, C., DeLuca, C., Balaji, V., Suarez, M. and Silva, A.D., 2004. The architecture of the Earth System Modeling Framework. Computing in Science and Engineering, 6, pp.18-28.

58. Hillyer, C., Bolte, J. and Lamaker, A., 2003.The ModCom modular simulation system. European Journal of Agronomy, 18, pp.333-343.

59. Homer, C., Huang, C. , Yang, L. , Wylie, B. and Coan,M., 2004. Development of a 2001 national land cover database for the United States. Photogrammetric Engineering and Remote Sensing, 70(7), pp.829-840.

60. Holzworth, D.P., Huth, N.I. and de Voil, P.G., 2010. Simplifying environmental model reuse. Environmental Modelling & Software, 25, pp.269-275.

61. Horak, J., Orlik, A. and Stromsky, J., 2008. Web services for distributed and interoperable hydro-information systems. Hydrology and Earth System Sciences, 12, pp.635-644.

62. Howie, C.T., Kunz, J.C. and Law, K.H., 1996. Software Interoperability, Stanford University, Stanford,CA. Prepared for Rome Laboratory, accessed October 2010 at http://www.dacs.dtic.mil/techs/interop/title.shtml.

63. Hummel, J. R. and Christiansen, J. H., 2002. The dynamic information architecture system: a simulation framework to provide interoperability for process models. In: Proceedings of Simulation Interoperability Workshop, Orlando, FL, 8-13th September.

64. Hussein, A. S. A.,1999. Grass ET estimates using Penman-type equations in central Sudan. Journal of Irrigation and Drainage Engineering, 125(6), pp. 324-329.

65. INSPIRE 2007, Infrastrucutre for Spatial Information in the European Community, European Commission, accessed October 2010 at http://inspire.jrc.ec.europa.eu/.

66. Jha, M., Gassman, P.W., Secchi,S., Gu,R. and Arnold, J., 2004. Effect of watershed subdivision on SWAT flow, sediment, and nutrient predictions. Journal of the American Water Resources Assoication, 40(3), pp.811-825.

67. Juracek, K.E. and Wolock, D.M., 2002. Spatial and statistical differences between 1:250,000- and 1:24,000-scale digital soil databases. Journal of Soil and Water Conservation, 57(2), pp.89-94.

68. Knapen, J.J.R., Verweij, P., Wien, J.E. and Hummer,S.,2009. OpenMI-The universal glue for integrated modeling? In: 18th World IMACS Congress and MODSIM09 International Congress in Modeling and Simulation, Cairns, Austraila, 13-17th July, pp.895-901.

69. Killeen, T., DeLuca, C., Toth, G., Gombosi, T., Stout, Q., Goodrich, C., Sussman, A. and Hesse, M., 2006, Integrated frameworks for earth and space weather simulation. In:American Meteorological Society Meeting, Atlanta, GA, 29th January -2rd February, pp.9.

70. Kralisch, S., Krause, P. and David, O., 2005. Using the object modeling system for hydrologic model development and application. Advances in Geosciences, 4, pp.75-81.

71. Krause, P., Boyle, D.P. and Base,F., 2005. Comparison of different efficiency criteria for hydrologic model assessment. Advances in Geosciences, 5, pp.89-97.

72. Kuo, Y.H., Klemp,J.B., and Michalakes, J., 2004. Mesoscale numerical weather prediction with the WRF Model. In: Symposium on the 50th Anniversary of Operational Numerical Weather Prediction, University of Maryland, pp.14-17.

73. Lakhankar, T., Fekete, B.M. and Vörösmarty, C. J., 2008. Semantic web infrasturece supporting NextFrAMES modeling platform. In: American Geophysical Union, Fall Meeting, Sanfrancisco,CA, 15-19th December, abstract #IN11B-1035.

74. Lathrop, R.G., Aber, J.D. and Bognar, J.A.,1995. Spatial variability of digital soil maps and its impact on regional ecosystem modeling. Ecological Modeling, 82(1), pp.1-10.

75. Leavesley, G.H., Markstrom, S.L., Brewer, M.S. and Viger, R.J., 1996a. The modular modeling system (MMS) --the physical process modeling component of a database-centred decision support system for water and power management. Water, Air and Soil Pollution, 90, pp.303-311.

76. Leavesley G.H., Restrepo P.J., Markstrom S.L., Dixon M. and Stannard L.G. 1996b. The Modular Modelling System (MMS): User's Manual. Report 96-151, U.S. Geological Survey, 175 pp.

77. Leavesley, G.H., Markstrom, S.L.,Viger, R.J.,and Hay, L.E., 2005. The Modular Modeling System(MMS):a toolbox for water- and environmental-sources management. US Geological Survey and International G-WADI Modelling Workshop, February-March 2005. National Institute of Hydrology, Roorkee, India.

78. LLNL 2010, Lawrence Livermore National Laboratory, The ParFlow Project: modeling surface and subsurface flow on high performance computers, accessed October 2010 at https://computation.llnl.gov/casc/parflow/overview.html.

79. Ludäscher, B., Altintas, I., Berkley, C., Higgins, D., Jaeger, E., Jones, M., Lee, E.A., Tao, J. and Zhao, Y., 2006. Scientific workflow management and the Kepler system, Concurrency and Computation: Practice & Experience, 18(10), pp.1039-1065.

80. Maidment, D. R., 2002. Arc Hydro GIS for Water Resources, ESRI Press.

81. Mamillapalli, S., Srinivasan, R., Arnold,J.G. and Engel, B.A., 1996. Effect of spatial variability on basin scale modeling. In: Proceedings of the Third International Converence/Workshop on Integrating GIS and Environmental Modeling. National Center for Geographic Information and Analysis,Santa Barbara,California. Available at http://www.ncgia.ucsb.edu/conf/SANTA_FE_CD-ROM/ sf_papers/ mamillapalli_ sudhakar/ my_paper.html. Accessed on 10 March 2011.

82. Marke, T., Mauser, W., Pfeiffer, A., and Zangl, G., 2011. A pragmatic approach for the downscaling and bias correction of regional climate simulations-evaluation in hydrological modeling. Geoscience Model Development Discussions, 4, pp.45-63.

83. Maxwell, T. and Costanza, R., 1994. Spatial ecosystem modeling in a distributed computational environment. towards sustainable development: concepts, methods, and policy. Island Press, Washington, D.C. pp. 111-138.

84. Maxwell.T. and Costanza, R., 1996. Facilitating high performance, collaborative spatial modeling.In: 3rd International Conference on Integrating Geographic Information Systems and Environmental Modelling. Santa Barbara, CA, 21-25th January, pp.8.

85. Maxwell, T., 1999. A paris-model approach to modular simulation. Environmental Modelling & Software,14 (6), pp.511-517.

86. Maxwell, T. and Costanza, R., 1995. Distributed modular spatial ecosystem modelling. International Journal of Computer Simulation, 5(3), pp.247-262.

87. McDonald, M.G. and Harbaugh, A.W., 1988. A modular three-dimensional finite-difference ground-water flow model. USGS Techniques of Water-Resources Investigations, Book 6, Chapter A1. USGS.

88. Meyer, B., 2001. .net is coming. Computer, 34(8), pp.92-97.

89. Michalakes, J., Chen, S., Dudhia, J., Hart, L., Klemp, J., Middlecoff, J. and Skamarock, W., 2001. Development of a Next Generation Regional Weather Research and Forecast Model. In: Proceedings of the 9th ECMWF Workshop on the Use of High Performance Computing in Meteorology, Singapore, 25-29th October, pp. 269-276.

90. Michalakes, J., Dudhia, J., Gill, D., Klemp, J. and Skamarock, W., 1998. Design of a next-generation regional weather research and forecast model. Towards Teracomputing.World Scientific, River Edge, New Jersey, pp. 117-124.

91. Microsoft Research Group, 2009. Project Trident: An Introduction. Microsoft project Trident: A scientific workflow workbench. User Manual. Version 1.0.

92. Mitchell E., Lohmann, D., Hourser, P.R., Wood, E.F., Schaake, J.C., Robock, A., Cosgrove, B., Sheffield, J., Duan, Q., Luo, L., Wayne, H., Pinker, R., Tarpley, J., Lettenmaier, D., Marshall, C., Entin, J., Pan, M., Shi, W., Koren, V., Meng, J., Ramsay, B. and Bailey, A., 2004. The multi-institution North American Land Data Assimilation System (NLDAS): Utilizing multiple GCIP products and partners in a continental distributed hydrological modeling system. Journal of Geophysical Research, 109, doi:10.1029/2003JD003823, pp.32.

93. MM5, 2003, The PennState/UCAR mesoscale model Community Model, accessed October 2010 at http://www.mmm.ucar.edu/mm5.

94. Monteith, J.L., 1973. Principles of Environmental Physics. Elsevier, New York, 241 pp.

95. Moore, R.V. and Tindall, C. I., 2005. An overview of the open modeling interface and environment (the OpenMI). Environmental Science & Policy, 8, pp.279-286.

96. Muttiah, R.S. and Wurbs,R.A.,2002. Scale-dependent soil and climate variability effects on watershed water balance of the SWAT model. Journal of Hydrology, 256, pp.264-285.

97. Myrabø, S., 1997. Temporal and spatial scale of response area and groundwater variation in till, Hydrological Processes, 11(14), pp.1861-1880.

98. Neitsch, S.L., Arnold, J.G., Kiniry, J.R. and Williams, J.R., 2005. Soil and Water Assessment Tool Theoretical Documentation, Version 2005. Grassland, Soil and Water Research Center, Texas Agricultural Experiment Station, Temple, Texas.

99. Oinn, T., Addis, M., Ferris, J., Marvin, D., Senger, M., Greenwood, M., Carver, T., Glover, K., Pocock, M.R., Wipat, A. and Li, P., 2004. Taverna: a tool for the composition and enactment of bioinformatics workflows. Bioinformatics, 20(1), pp.3045-3054.

100. OpenMI, 2010, The Open Modeling Interface, The OpenMI Association, accessed October 2010 at http://www.openmi.org/.

101. Padulo,L. and Arbib, M.A., 1974. Systems Theory: a Unified State-Space Approach to Continuous and Discrete Systems. W.B.Saunders, Philadelphia, PA.

102. Patchen, R., 2007. Establishment of a Delaware Bay model evaluation environment. In: Spaulding, M.L.(eds.) Proceedings of the Tenth International Conference on Estuarine and Coastal Modeling, Newport, Rhode Island, 5-7th November, pp.783-818.

103. Pautasso, C. and Alonso, G., 2006. Parallel computing patterns for grid workflows. Workshop on Workflows in Support of Large-Scale Science(WORKS), Paris, France, June 19- 23, pp.1-10.

104. Peckham, S., 2008. CSDMS handbook of concepts and protocols: A guide for code contributors. Last accessed on 3rd January 2010 at http://csdms.colorado.edu/wiki/Tools_CSDMS_Handbook.

105. Penman, H.L., 1948. Natural evaporation from open water, bare soil and grass. Proceedings of the Royal Society. London, A193, pp.120-146.

106. Pereira, A.R. and Pruitt, W.O., 2004. Adaptation of the thornthwaite scheme for estimating daily reference evapotranspiration. Agricultural Water Management, 660, pp.251-257.

107. Peterson, J.R. and Hamlett,J.M.,1998. Hydrologic calibration of the SWAT model in a watershed containing fragipan soils. Journal of the American Water Resources Association. 34(3), pp.531-544.

108. PIHM, 2010, Penn State Integrated Hydrologic Modelling System, accessed in October 2010 at http://pihm.psu.edu/index.html/

109. Pressman, R.S., 2001. Software Engineering: A Practitioner's Approach. McGraw Hill. New York.

110. Priestley, C.H.B. and Taylor, R.J.,1972. On the assessment of surface heat flux and evaporation using large-scale parameters. Monthly Weather Review,100, pp.81-92.

111. Qu, Y. and Duffy, C. J., 2007. A semidiscrete finite volume formulation for multiprocess watershed simulation. Water Resources Research, 43, W08419, doi:10.1029/2006WR005752.

112. Quinn, P.F., Ostendorf, B., Beven, K.J. and Tenhunen, J., 1998, Spatial and temporal predictions of soil moisture patterns and evaporative losses using TOPMODEL and the GAS-Flux model for an Alaskan catchment. Hydrology and Earth System Sciences, 2, pp. 51-64.

113. 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. In: Proceedings of International Congress on Modelling and Simulation (MODSIM03), Townsville, Australia, 14-17th July, pp. 1727-1732.

114. Rahman, J.M., Seaton, S.P. and Cuddy, S.M., 2004a. Making frameworks more useable: using model introspection and metadata to develop model processing tools. Environmental Modelling & Software, 19, pp.275-284.

115. Rahman, J.M., Cuddy S.M. and Watson, F.G.R., 2004b. Tarsier and ICMS: Two Approaches to Framework Development, Mathematics and Computers in Simulation, 64(3-4), February 2004, pp. 339-350.

116. Rajbhandari, S., Wootten, I. and Rana, O., 2006. Evaluating provenance-based trust for scientific workflows. In: Sixth IEEE International Symposium on Cluster Compution and the Grid(CCGrid06), Singapore, pp.365-372.

117. Rassam, D.W., Pickett,T. and Knight, J.H., 2009. Incorporating floodplain groundwater interactions in river modeling. In: 18th World IMACS/MODSIM Congress, Cairns, Australia, 13-17th July, pp.3116-3122.

118. Reed, M., Cuddy, S.M. and Rizzoli, A.E., 1999. A framework for modeling multiple resource management issues-an open modeling approach. Environmental Modelling and Software , 14, pp.503-509.

119. Rizzoli A.E., Davis, J.R. and Abel, D.J., 1998. Model and data integration and re-use in environmental decision support system. Decision Support Systems, 24(2), pp.127-144.

120. Rosenthal, W. D., Srinivasan, R. and Arnold, J. G., 1995. Alternative river management using a linked GIS-hydrology model. Transactions of the ASAE, 38(3), pp.783-790.

121. Rumbaugh, J., Blaha, M., Premerlani,W., Eddy, F. and Lorensen,W., 1991. Object-oriented Modeling and Design. Prentice-Hall.

122. Saulnier, G.M., Beven, K.J. and Obled, Ch., 1998, Including spatially variable soil depths in TOPMODEL. Journal of Hydrology, 202, pp.158-172.

123. Schut, P., 2007. OpenGIS® Web Processing Service. Open Geospatial Consortium Inc. OGC 05-007r7, June.

124. Seo, D. J., 1999. Real-time estimation of rainfall fields using radar rainfall and rain gauge data. Journal of Hydrology, 208, pp.37-52.

125. Seo, D. J. and Breidenbach,J., 2002. Real-time correction of spatially nonuniform bias in radar rainfall data using rain gauge measurements. Journal of Hydrometeorogy, 3, pp.93-111.

126. Simmhan, Y., Plale, B. and Gannon, D., 2005. A survey of data provenance in e-science,SIGMOD Record, 34 (3), pp.31-36.

127. Simmhan, Y.L., Plale, B. and Gannon,D., 2006. A framework for collecting provenance in data-centric scientific workflows. In: International Conference on Web Service (ICWS), Chicago, IL, 18-22th September,pp.427-436.

128. Singh, V. P. and Woolhiser, D. A., 2002, Mathematical modeling of watershed hydrology, Journal of Hydrologic Engineering, 7, pp.270- 292.

129. Srinivasan,R. and Arnold,J.G.,1994. Integration of a basin scale water quality model with GIS. Water Resources Bulletin, 30(3), pp.453-462.

130. Stammermann R. and Piasecki, M., 2009. Skill assessement of MARINA and UnTRIM hydroldynamic codes using NOAA's Delaware Bay estuary modeling evaluation environment. In: Spaulding, M.L.(eds.) Proceedings of the Eleventh International Conference Estuarine and Coastal Modeling, Seattle, Washington, 4-6th November, pp. 27-46.

131. Sydelko, P.J., Hlohowskyj, I.,Majerus, K., Christiansen, J. and Dolph,J., 2001. An object-oriented framework for dynamic ecosystem modeling: application for integrated risk assessment. The Science of the Total Environment, 274, pp.271-281.

132. Tarboton, D.G., Bras, R.L. and Rodriguez-Iturbe, I., 1991. On the extraction of channel networks from digital elevation data. Hydrological processes, 5, pp.81-100.

133. Taylor, I.J., Shields,M., Wang, D.I.,2003. Distributed P2P computing within Triana: A galaxy visualization test case. In: 17th International Parallel and Distributed Processing Symposium (IPDPS), Nice, France, 22-26th April, pp.16-27.

134. Thornthwaite, C.W., 1948. An approach toward a rational classification of climate. Geography Review. 38 (1), pp.55-94.

135. Tsai, V.J.D., 1993. Delaunay triangulations in TIN creation: an overview and a linear-time algorithm. International Journal of Geographical Information Science, 7(6), pp.501-524.

136. Tucker, G., Lancaster, S., Gasparini, N.M., Bras,R.L. and Rybarczyk, S.M., 2001. An object-oriented framework for distributed hydrologic and geomorphic modeling using triangulated irregular networks.Computers & Geosciences, 27, pp.959-973.

137. Tyagi, N. K., Sharma, D. K., and Luthra, S. K., 2003. Determination of evapotranspiration for maize and berseem clover. Irrigation Science, 21(4), pp.173-181.

138. USGS, 2009. Status of ModFlow versions and ModFlow-related programs available on USGS Web Pages. Office of Groundwater,U.S.Geological Survey, last accessed on 10th January 2010 at http://water.usgs.gov/nrp/gwsoftware/modflow-status.pdf

139. USGS/EROS, 2010. Download Web Service Users Guide. Revision 2. 26th February.

140. Valcke, S., Declat, D., Redler, R., Ritzdorf, H., Schoenemeyer, T. and Vogelsang, R., 2004. The PRISM coupling and I/O system, In:Proceedings of the 6th International Meeting: High performance computing for computational science, Valencia, Spain, 28-30th June, pp.845-850.

141. Valcke, S., Guilyardi, E. and Larsson, C., 2006. PRISM and ENES: A European approach to earth system modelling. Concurrency and Computation: Practice and Experience, 18(2), pp. 231-245.

142. Vogelmann, J.E., Howard, S.M., Yang, L., Larson, C.R., Wylie, B.K. and Van Driel, J.N., 2001. Completion of the 1990's National Land Cover Data Set for the conterminous United States. Photogrammetric Engineering and Remote Sensing, 67, pp.650-662.

143. Voinov, A., Zaslavskiy, I., Arctur, D., Duffy, C. and Seppelt, R., 2008. Community modeling, and data-model interoperability. In: Proceedings of 4th Biennial Meeting of International Environmental and Software Society, Barcelona, Catalonia, 7-10th July, pp.2035-2047.

144. Wagener, T., Reed, P., van Werkhoven, K., Tang, Y. and Zhang, Z., 2009. Advances in the Identification and Evaluation of complex Environmental Systems Models, Journal of HydroInformatics, Vol 11, No 3-4, pp 266-281, IWA Publishing, doi:2166/hydro.2009.040

145. Watson, F.G.R., Rahman, J. and Seaton, S., 2001. Deploying environmental software using the Tarsier modelling framework. In: Proceedings of 3rd Australian Stream Management Conference, Brisbane, Australia, 27-29th August , pp.631-637.

146. Watson, F.G.R. and Rahman, J.M., 2004. Tarsier: a practical software framework for model development, testing and deployment.Environmental Modelling & Software, 19 (3), pp.245-260.

147. Whiteaker,T. and To, E., 2008. CUAHSI Web Services for Ground Water Data Retrieval. Ground Water, 46(1), pp.6-9.

148. Wilson, J. P. and Gallant, J. C., 2000. Terrain Analysis: Principles and Applications. 1st edition, John Wiley and Sons, New York.

149. WRF, 2010. The Weather Research and Forecasting Model. Accessed October 2010 at http://wrf-model.org/index.php.

Appendix A: ACTIVITIES IN HYDROLOGIC COMMUNITY MODELING SYSTEM

Library	Sub-Category	Activities
Data Access Library	USGS seamless data warehouse	1)Download National Elevation Data
		2)Download Land Cover Data
	EPA Geospatial Service	3)Access Watershed and Stream Shapefile
	SSURGO	4)Access SSURGO Soil Data
	HIS WaterOneFlow	5)Get Web Services within a Geospatial Area
		6)Retrieve Sites in a Specified Extent
		7)Get Time Series Data
		8)Variables Semantic Checking
	NLDAS-2	9)Access NLDAS-2 Forcing Data
		10)Extract Data within a Geospatial Extent
		11)Extract Data of Specified Fields
		12)Access, Extract and Aggregate Time Series for Sub-basins
	MPE	13)Access MPE data from NWS
		14)Parse MPE netCDF files
		15)Get Data within a Specified Area
	Read Data Files	16)Read Basin Data from a XML file
		17)Read Basin Data from Shapefiles
		18)Read Data from an XML file
		19)Read Data from an Excel File
		20)Read Data from SQL Database
		21)Read Data from CSV file
Data Processing Library	GeoProcessing	22)DEM Processing: Fill Sinks
		23)DEM Processing: D8 Flow Direction
		24)DEM Processing: D8 Flow Contributing Area
		25)DEM Processing: Grid Network and Flow Path
		26)DEM Processing: Define Stream Network
		27)DEM Processing: Watershed Delineation

Continued

Library	**Sub-Category**	**Activities**
Data Processing Library	GeoProcessing	28)DEM Processing: Watershed Grid To Shapefile
		29)Locate an Outlet Based on Given Sites
		30)Terrain Processing(The whole Dem processing procedure)
		31)Invoke DEM Processing Web Service
		32)TIN Generation
		33)Invoke TIN Generation Web Service
		34)Clip and Reclassify USGS Land Cover Data
		35)Merge Soil Shapes and Clip Soil Shape within the Boundary
	Time Series Processing	36)Prepare Soil Spatial Data for Watershed(from Soil Data Mart)
		37)Simplify Soil Groups into A,B,C,D
		38)HRU Builder
		39)Unit Converter
		40)Invoke Unit Converter Web Service
		41)Process USGS Instantaneous Data
		42)Spatial Interpolation: Inverse Weighted Method
		43)Time Series Interpolation: Linear Method
		44)Compute Mean Precipitaiton For Sub-basins(For MPE)
		45)Aggregate Time Series of Given Fields for Sub-basins(For NLDAS-2)
		46)Compute Maximum/Minimum/Average Value
Hydrologic Model Library	TOPMODEL	47)GRIDATB:Compute Topographic Index In(a/tanB)
		48)Computing Area-Distance Histogram
		49)Assigning Parameters& Initial condition
		50)TOPMODEL Main Computation
	SWAT	51)Snow Melt Component
		52)Modified Curve Number Method
		53)Overland Routing Component
		54)Soil Water Component

Continued

Library	Sub-Category	Activities
Hydrologic Model Library	SWAT	55)Groundwater Component
		56)Channel Routing Component
	Hydrologic Methods	57)ET: Hargreaves Method
		58)ET: Thornthwaite Method
		59)ET: Priestley Taylor Method
		60)ET: Penman-Monteith Method
		61)Runoff Yield&Infiltration: Green Ampt Method
		62)Runoff Yield&Infiltration: Curve Number Method
		63)Direct runoff routing: SCS Unit Hydrograph
		64)Direct runoff routing: Clark Unit Hydrograph
		65)Direct runoff routing: Synder Unit Hydrograph
		66)Baseflow: Linear Reservoir
		67)Baseflow: Recession method
		68)Channel Routing: Muskingum Method
		69)Channel Routing: Modified Wave Method
Analysis& Utilities Library	Performance Evaluation	70)Hydrograph Visualization and Statistics
		71)Correlation Analysis
		72)Water Balance Check
		73)Geospatial Data Viewer
		74)Time Series Viewer
		75)Decompress ZIP or TAR-ZIP files
	Utilities	76)Search Files
		77)Create Shapefile for Sites
		78)Sites Filter for a Spatial Extent
		79)Sites Filter for a Temporal Scale
		80)Write Time Series to an Excel File
		81)Write Time Series to an XML File
		82)Write Time Series to SQL Database
		83)Merge Grid Files
		84)Change Grid Format

VITA

Bo LU

Education:

PhD, Civil, Architectural & Environmental Engineering, (June 2011), GPA 4.00/4.00, Drexel University, Philadelphia, PA.

MS, College of Water Resources & Environment, (June 2006), GPA 3.5/4.0, Hohai University, Nanjing, China.

BS, College of Water Resources & Environment, (June 2003), GPA 3.5/4.0, Hohai University, Nanjing, China.

Publications:

Lu, B., and Piasecki, M. (2011). “A Review of Efforts to Establish a Hydrologic Community Modeling System” Submitted to: *Journal of Hydroinformatics.*

Lu, B., and Piasecki, M. (2010). “An Experiment of Using a Workflow Engine for Hydrologic Science” Proceedings of 9th International Conference on Hydroinformatics, Tianjin, China.

Lu, B., and Piasecki, M. (2010). “Development of OpenMI Compliant Components to Form a Hydrologic Model System” Proceedings of 9th International Conference on Hydroinformatics, Tianjin, China.