# Process integration, data management, and visualization framework for subsurface sciences

**KL Schuchardt, GD Black, JM Chase, TO Elsethagen and L Sun**

Pacific Northwest National Laboratory, PO Box 999 Richland, WA 99320

karen.schuchardt@pnl.gov

**Abstract**. Applying subsurface simulation codes to understand heterogeneous flow and transport problems is a complex process potentially involving multiple models, multiple scales, and spanning multiple scientific disciplines. A typical end-to-end process involves many tools, scripts and data sources usually shared only though informal channels. Additionally, the process contains many sub-processes that are repeated frequently and could be automated and shared. Finally, keeping records of the models, processes, and correlation between inputs and outputs is currently manual, time consuming and error prone. We are developing a software framework that integrates a workflow execution environment, shared data repository, and analysis and visualization tools to support development and use of new hybrid subsurface simulation codes. We are taking advantage of recent advances in scientific process automation using the Kepler system and advances in data services based on content management. Extensibility and flexibility are key underlying design considerations to support the constantly changing set of tools, scripts, and models available. We describe the architecture and components of this system with early examples of applying it to a continuum subsurface model.

## 1. Introduction

Advancements in computer hardware for massively parallel computation and subsequent storage for the volumes of data produced have signalled the need for development of subsurface modeling algorithms, techniques, and codes that fully utilize this new generation of hardware. But, without a software infrastructure, in the form of a user environment on top of the low-level computational codes that enables researchers, experimentalists, and site modelers to organize and carry out such complex investigations, much of this potential to change how subsurface modeling is performed will remain unrealized. Discovering the critical pieces of this user environment and developing an infrastructure that integrates them with underlying subsurface modeling codes and supporting tools in an extensible fashion is the goal of the Process Integration Framework for Subsurface Simulations (PIFSS).

Driving the design of the PIFSS environment is a set of requirements that, if met, allow the advancements in computing hardware and subsurface modeling software to be fully exploited. The first requirement is being able to capture and automate complex processes in a standard framework. The complexity of these processes is a result of different computational models that must be applied and the volume and variety of data that is associated with each of these models, all across heterogeneous computing environments. The second requirement is being able to associate the input and the output from these complex processes for the purpose of verifying correctness of what has been done, discovering problems in processes as a result of dependent data being updated or corrected, and

being able to easily repeat past simulations. The third requirement is to support shared repositories of standard data, tools, and even complex processes integrating multiple models and tools. The development of these shared repositories will allow a wider audience of users including across disciplines (e.g. continuum and pore scale modelers) and across job functions (e.g. researchers, experimentalists, and site modelers) to take advantage of better and more appropriate software quickly and with more accurate results.

## 2. Approach

In collaboration with subsurface researchers, we have elaborated on the overall research process that will be applied to the hybrid models currently under development by Scheibe et al. (this issue). This process model, shown in Figure 1, includes code development, input preparation, analysis, decision points, and iteration based on analysis. It is our goal to support these processes through an integrated, extensible framework that enhances repeatability and traceability while fully leveraging advanced hardware and software tools. While the hybrid model that combines pore scale, continuum, and potentially other scales is under initial development, we start by examining the component models, the STOMP[1] continuum code and the SPH[2] pore-scale code, and applying them to calcite precipitation experiments described in Scheibe et al.

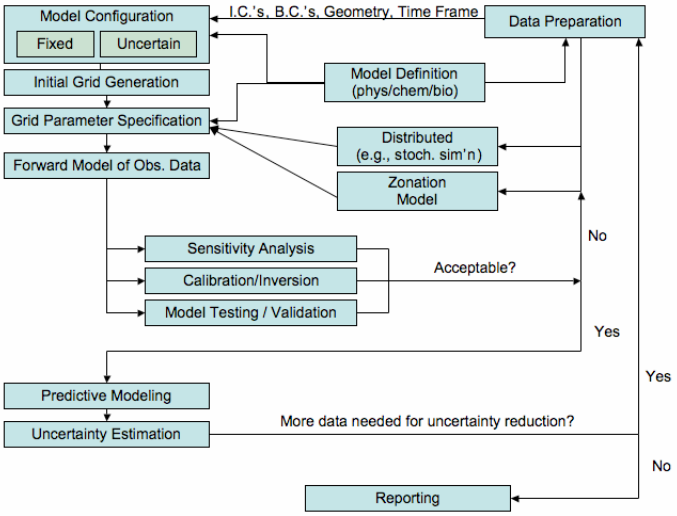


**Figure 1**. General model of subsurface simulation workflow. The workflow includes iterative processes based on user analysis and decisions as well as parallel executions of the models for parameters studies and sensitivity studies.

## 3. Process Integration Framework

The four primary components of the PIFSS architecture as shown in Figure 2 are the workflow component, data services component, the "organizer" component, and the visualization component. The workflow and data management components are built on existing open source technologies but are customized and extended for the purpose of fitting into a more expansive environment tailored specifically for subsurface modeling. The organizer component has been developed on the PIFFS project as a central integrating tool for accessing data, executing workflows, and performing analysis and visualization. Each of these is described in the following sections. Visualization will primarily involve integrating tools provided by external collaborators, including parallel visualization capabilities for large data sets, and is therefore not discussed in detail in this paper.

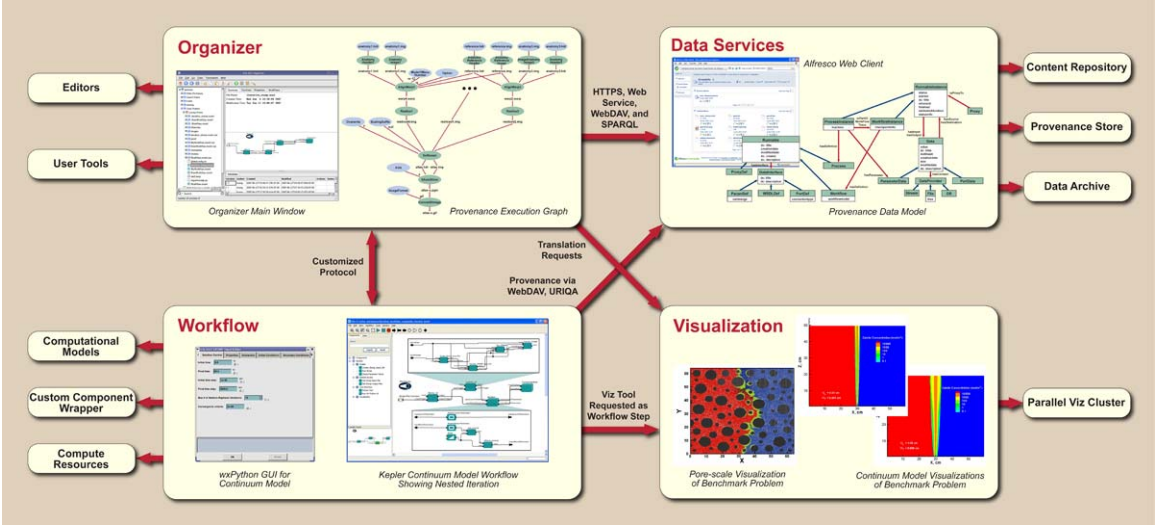


**Figure 2**. PIFSS Component Architecture

### 3.1. Organizer

The organizer is the central integrating tool of PIFSS providing the user's view of the information stored on the data server, the point of access for workflow development and execution, and an interface to the shared knowledge base of data, tools, and processes. It is written using Python and wxPython, providing the benefits of a rich client application, extensibility, platform portability, simple deployment, and fast prototyping and code development. The organizer application communicates with the content management system through the WebDAV (webdav.org) protocol and web services. It currently interacts with the data management system to upload and download files, view data and metadata, and view data version information. In the future, it will support user account creation, management of access controls, metadata editing, version management, and provenance queries.

In addition to providing a view onto the data server, the organizer provides extensible mechanisms to access tools that can view or operate on data. For each data type (MIME-type), there is a default 'open' action, a dialog to configure tool invocation, and finally, an easily extended class hierarchy that supports custom integration methods. The latter is used to create a very tight, virtual binding with the workflow designer tool. Typically, user registered tools make it possible to invoke different analysis tools or even the same analysis tools with different start-up scripts.

Besides simply invoking tools, it is sometimes necessary to first transform data into a format understood by the tool. We envision both simple two-step tool invocations (translate, invoke) and configurable, multi-step workflows to prepare data for analysis or visualization. Finally, we plan to explore custom views, navigation, and manipulation of workflow execution provenance.

### 3.2. Workflow

Process workflow is a description of the steps or tasks in a process including dependencies between those steps such as order of execution and data that needs to "flow" between the steps. The complete description is in a form that can be saved, executed, and later called back and re-executed. While workflow technology originated in the business world, it has more recently been applied within the scientific community. Workflow systems typically consist of two parts: a designer for defining the process steps, and an execution engine capable of scheduling and invoking the steps. Scientific workflow poses several unique challenges: the target user of the designer is a scientist who requires an intuitive abstract representation of process steps not available with existing workflow tools, distributed execution of frequently long running processes, the need to move large amounts of data, the frequent evolution of workflows, the need for iteration and debugging, and frequent integration of new tools. Extending existing tools to meet these challenges and integrating the tools into a more extensive user environment is a challenge to be addressed on the PIFSS project.

The Kepler open-source workflow system has been chosen as the initial workflow tool both for workflow design and execution. Kepler has been applied to several scientific domains, supports concurrent execution, and significant effort has gone into building components tailored for scientific workflow. Figure 3 shows a workflow that concurrently executes a parameter study using the continuum STOMP model. For each iteration, a parameter value such as porosity is updated over an incremental range while holding all other parameters constant. Once all iterations and thus the workflow has completed, the data can be analyzed to choose the parameter value that best matches the desired model behavior. The two insets to figure 3 are nested workflows that hide the complexity of moving files and submitting jobs to remote systems and iteration control.

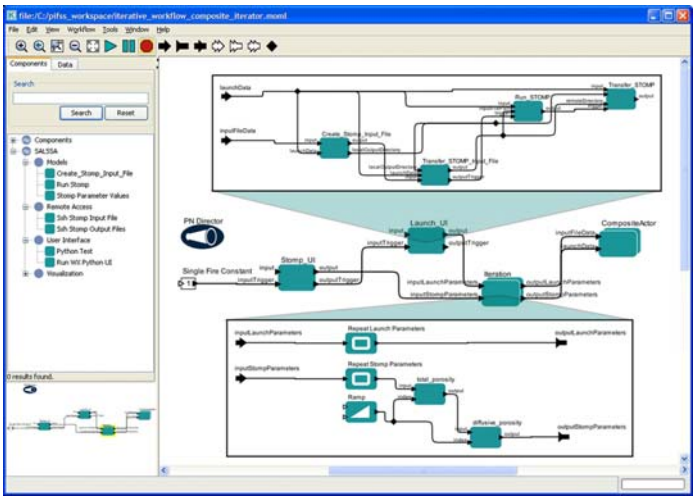

**Figure 3**. Workflow designer tool showing an iterative process for running STOMP. The details of workflow complexity are hidden from users through the use of nested workflows (insets).

Kepler supports end user extensibility through the creation of new components that can be used in a workflow. This typically involves writing java code and registering it with the system. Our goal is to identify the common, general components or wrappers such that this type of extension mechanism is rarely if ever needed by the subsurface scientists. One example of a generic component we have added is the wxPython wrapper. This component enables a scientist to create a wxPython based user interface (for example to prompt for some model inputs) without explicit knowledge of Kepler extension mechanisms. Future efforts will center on providing a higher level of abstraction for defining and invoking research processes.

Kepler has been tightly integrated with the organizer and data server. Any changes to the workflow description are saved as versions of the workflow, similar to the concept of versioning of source code, thus providing history of workflow descriptions. Kepler has also been augmented with an initial provenance capture mechanism. When a workflow executes, a provenance component receives events from the execution engine including process execution, data moving between processes, start and stop of execution, etc. This provenance, along with data artefacts (input and output files), is captured in our provenance store and together with the workflow description fully document data provenance. This provenance record will play an important role in later work to support repeatability and verifiability.

### 3.3. Data, Metadata, and Provenance Services

The PIFSS architecture builds on the capabilities of content management technology to provide data services. Content Management (CM) is a set of processes and technologies that support the evolutionary life cycle of digital information. For scientific data, lifecycle support includes storage and access of arbitrary data in its raw format, versioning, ownership, modification history, support for arbitrary metadata, provenance, user supplied annotations, and semantic information on the type of content. We have selected the Alfresco (alfresco.com) CM system, an open-source and open-

standards content repository. Alfresco incorporates favored open-source technologies such as the Spring (springframework.org), ACEGI for security (acegisecurity.org), Lucene (lucene.apache.org) for indexing, content management standards, and interface standards such as FTP, WebDAV, web services, CIFS (http://www.microsoft.com/mind/1196/cifs.asp), and contributions from a strong open source community.

Alfresco provides a web-client application to manage, configure, and explore the repository as well as a development framework to extend or add new functionality in support of a project's requirements. To date, we have tailored Alfresco by adding custom metadata extraction for STOMP data files, versioning rules, default user account settings, and a special provenance store that enables us to capture and query provenance about workflow execution histories. The current provenance store is the SESAME (openrdf.org) Resource Description Framework (RDF). RDF provides a way to express arbitrary statements about arbitrary resources. In our case, it is used to fully document the processes that execute in a workflow, parameters and data used, and relationships between processes, data, and parameters. An initial provenance data model has been developed detailing class and relationship definitions. The model includes class definitions for workflow elements, such as, *Workflow*, *Port*, *Parameter*, and *Data*. Relationship definitions include *hasInput*, *hasOutput*, *hasParameter*, and *isPartOfWorkflowTrace*, among others. Provenance can be queried using the graph query language SPARQL (http://www.w3.org/TR/rdf-sparql-query/), which enables us to begin answering questions such as "what steps led to the creation of a specific data file" or "if this input data changes, what results might need to be re-examined".

### 3.4. CCA Integration

The new hybrid models supported by PIFSS are being developed using the Common Component Architecture (CCA) [3] described in Palmer et al. (same issue). CCA-based codes have some interesting differences from standard codes in that they can be constructed dynamically. Initially, we plan to treat the CCA codes as "black boxes" that can be submitted as jobs like other codes through a job submission workflow component. However, we will develop special mechanisms to capture information about the model components at execution.

## 4. Summary

Our initial implementation includes individual software components that together make a very flexible, extensible, and powerful user environment for subsurface simulations for both research and predictive modeling. However, there are many challenges to be met especially in terms of providing the right level of abstraction for designing workflows, discovering the most useful level of detail for workflow provenance capture, and the best ways to present, use, and manipulate provenance. We plan to harden and deploy the system to experimentalists with limited simulation experience to model the calcite precipitation problem described in Scheibe et al (this issue) for initial evaluation.

## 5. Acknowledgements

This work was conducted at the Pacific Northwest National Laboratory, a multiprogram national laboratory operated by Battelle for the U.S. Department of Energy under contract DE-AC05-76RLO 1830.

## References

[1] Monaghan, J.J., *Smoothed particle hydrodynamics*. Reports on Progress in Physics, 2005. **68**: p. 1703-1759.

[2] White, M.D., M. Oostrom (2000), *STOMP Subsurface Transport Over Multiple Phases, Version 2.0, Theory Guide*. Pacific Northwest National Laboratory, PNNL-12030, UC-2010, Richland, Washington.

[3] Allan, B.A. *et al.*, *A component architecture for high-performance scientific computing*. The International Journal of High Performance Computing Applications, 2006. **20**: p. 203-231.