# Scalable Ensemble-based Oil-Reservoir Simulations using Blue Gene/P as-a-Service

Moustafa AbdelBaky*, Hyunjoo Kim*, Manish Parashar*, Kirk Jordan**, Hani Jamjoom**, Vipin Sachdeva**, Zon-Yin Shae**, James Sexton**, Gergina Pencheva***, Reza Tavakoli***, and Mary F. Wheeler***

*Center for Autonomic Computing, Rutgers University
**IBM T.J. Watson Research Center
***Center for Subsurface Modeling, Institute for Computational Engineering and Sciences, The University of Texas at Austin

## 1. Introduction & Motivations

Extreme-scale systems provide science and engineering applications the ability to explore large parameter spaces in order to simulate multi-scale and multiphase models and minimize uncertainty. This is achieved by simultaneously running ensembles, consisting of multiple realizations, and then assimilating their results. Such ensemble applications represent a significant class of applications that require effective utilization of high-end Petascale and eventually Exascale systems. For example, achieving an optimal Enhanced Oil Recovery (EOR) production strategy can more than pays for several hundred thousand cores of a large computer system. Industry refers to this data assimilation as the Instrumented Oil Field of the Future.

Running ensemble applications demands a large and dynamic pool of HPC resources with fast communication. From the users' perspective, ease of use, as well as programmatic dynamic scaling of expensive HPC resources is desirable. However, while current state-of-the-art HPC systems provide enough resources with fast communication, they require relatively low-level user involvement and expert knowledge of such systems. As a result, only a few "hero" users are able to effectively use these cutting edge systems. In addition, current HPC resources do not support dynamic scalability, a key requirement for most ensemble applications.

On the other hand, cloud computing is rapidly emerging as a dominant computing paradigm. Clouds provide easy-to-use as-a-service abstraction. They also support on-demand scale up and scale down. However, they have largely been ineffective for realistic HPC applications. Reasons for this include the limited capabilities and power of the typical underlying hardware and its non-homogeneity, the lack of high-speed interconnects to support data exchanges required by HPC applications, as well as the physical distance between machines. Even newer HPC cloud approaches, built on high-end clusters with faster interconnects (e.g. Amazon HPC service), are still outperformed by the high-end supercomputers that are specifically designed to support HPC applications. These supercomputers provide faster high-speed interconnects, and are far more scalable than cluster hardware, a necessity for ensemble applications. Therefore while ensemble applications could significantly benefit from a cloud abstraction, in particular from the ease of use and dynamic allocation of resources provided by the cloud paradigm, the current clouds are not suitable for these parallel applications.

## 2. Objective of the Demonstration

The proposed demonstration explores how a cloud abstraction can be used to provide a simple interface for current HPC resources. In particular, the benefits of the cloud paradigm, such as ease of use and dynamic allocation, and their application to supercomputers, specifically, the IBM Blue Gene/P system, are discussed, tested and validated. The underlying framework essentially transforms the Blue Gene/P supercomputer into an elastic cloud, supporting dynamic provisioning and efficient utilization while maximizing ease-of-use through an as-a-service abstraction.

## 3. Application Scenario

Ensemble applications modify large parameter spaces in order to optimize strategies and minimize uncertainty. This is achieved by running multiple realizations simultaneously and then combining their results. These results are then analyzed before an updated set of realizations is run. Each instance of the application (ensemble member) is a traditional parallel HPC application, which requires a varying number of processors and fast communication among these processors. In addition, typically a large and varying number of ensemble members are also required to achieve acceptable accuracy, which in turn requires a large and dynamic pool of resources. The key components of the demonstrated ensemble application workflow, the application (IPARS) and the filter (EnKF) are described below.

**IPARS** (Implicit Parallel Accurate Reservoir Simulator) provides a framework and a growing number of physical models suitable for research and practical applications. Both oil reservoirs and aquifers can be simulated using either black oil or compositional equation-of-state fluid models. IPARS can solve problems involving several million grid elements in parallel. It can handle multiple fault blocks with unaligned grids and problems that involve different physical models in various regions of the reservoir.

**Ensemble Kalman Filter (EnKF):** The EnKF methodology consists of a forecast step and an assimilation (update) step. The forecast step is equivalent to running the reservoir simulation (IPARS) for a suite of reservoir models (a set of realizations) independently to predict data at the next data assimilation time step. Based on difference between the observed and predicted data, in each assimilation step the Kalman update equation is used to update the model parameters (permeabilities, porosities) and simulation dynamic variables (pressure, saturation). The construction of the Kalman gain matrix requires collecting a very large state vector from each ensemble member. This process is continuously repeated in time as new dynamic data (production data) becomes available.

**Scenario:** The proposed demonstration models a two-phase flow (oil and water) in a three- dimensional horizontal reservoir with a 50x50x4 uniform grid. The model parameters are permeability and porosity. There are two water injection wells with constant injection rate, and seven producers constrained by constant liquid production rates. The injectors and five of the producers start operating at the beginning of the simulation time while the other two producers start production after 1000 days. The observation data to be assimilated in this demonstration include oil production rate, flowing bottom-hole pressure (BHP), and water-oil ratio (WOR) recorded every 100 days. The historical data are available for 3000 days and the total simulation time is 4500 days.

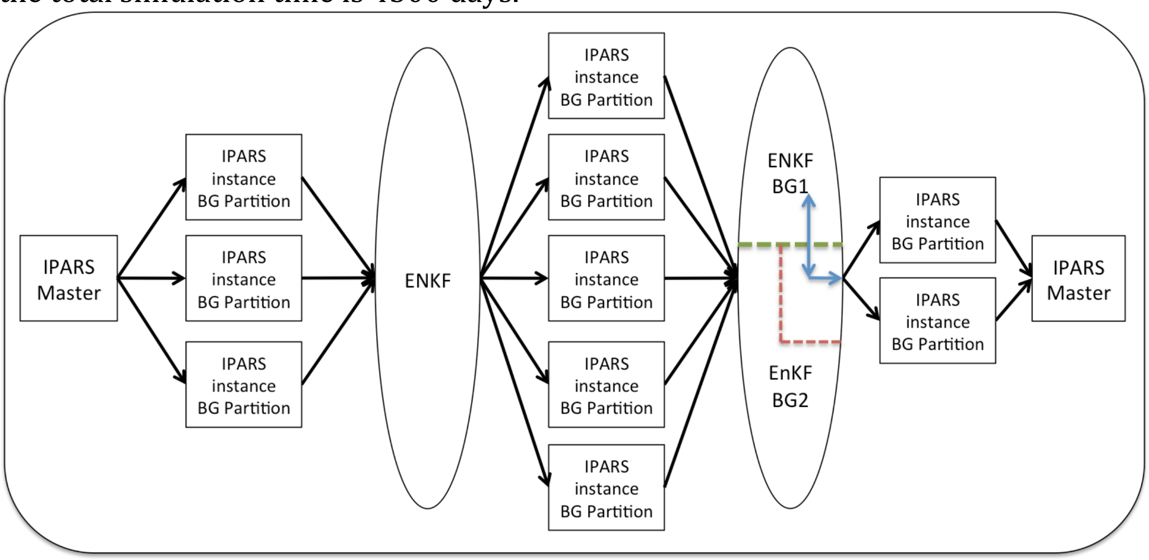



Figure 1. Application scenario

Figure 1 shows the overall flow of the ensemble application workflow. Multiple IPARS realizations run in parallel. The EnKF update step is done based on the results of all simulation runs. Finally a new set of realizations runs in parallel to constantly update the simulator. The process is repeated until the assimilation of last set of observed data.

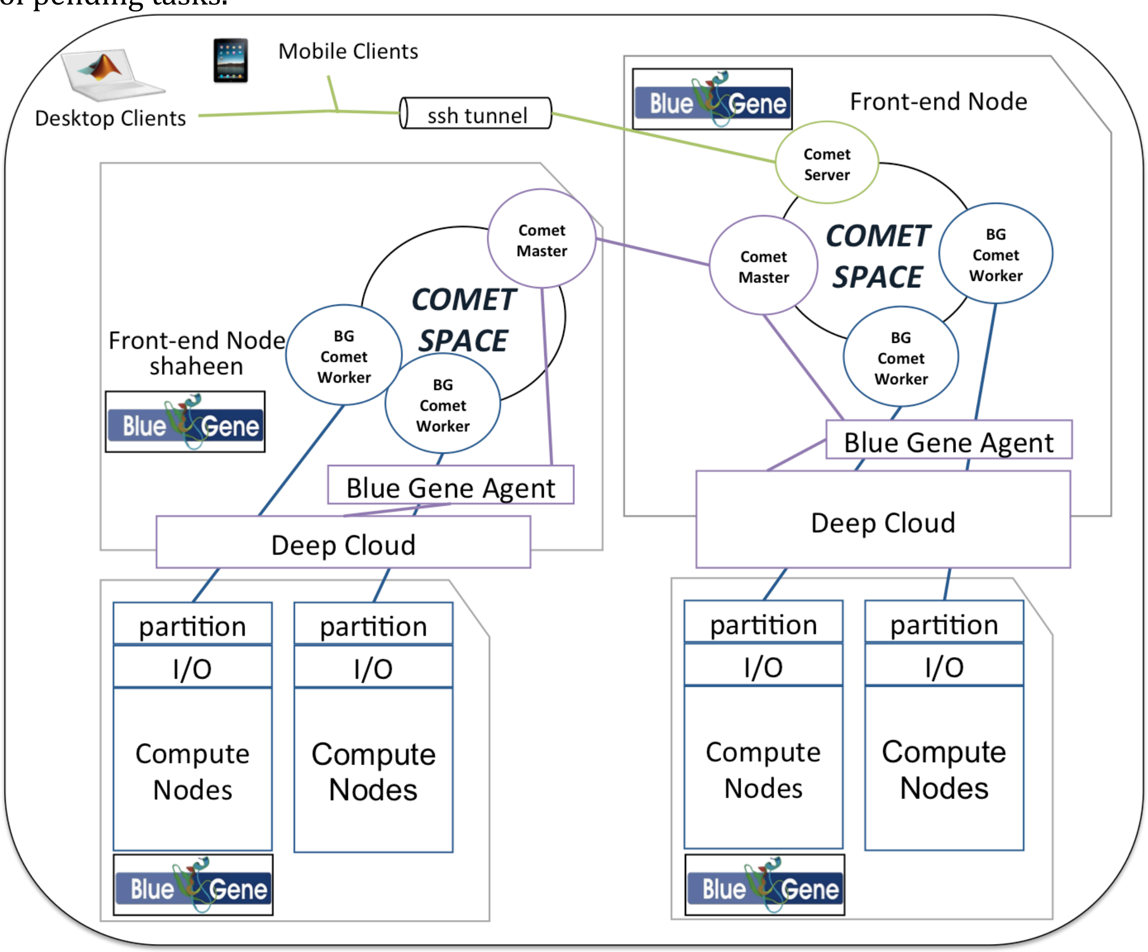
## 4. Technical Approach

The demonstrated solution integrates three key components, CometCloud, Deep Cloud and a Blue Gene Agent. The overall architecture is presented in Figure 2. These components are described below.

**CometCloud** is an autonomic computing engine that supports the development and deployment of applications on dynamically federated clouds. It provides the key services required for realizing the Blue Gene/P as-a-service abstraction, such as resource provisioning, application and system management services. CometCloud handles the scheduling and mapping of application tasks, the determination resources to be provisioned, the handling application and system level failures, etc.

**Deep Cloud** is a reservation-based system and a pricing model, currently being developed at the IBM T.J. Watson Research Center for managing supercomputer resources to provide users with an abstraction of unlimited resources, and to maximize their satisfaction by managing demand. Deep Cloud allows Blue Gene/P resources to be reserved and instantiated programmatically. Deep Cloud supports on-demand provisioning, repartitioning of the current Blue Gene/P resources. Deep Cloud can run on top of the batch system on production systems to support future reservation of allocations. It can also bypass the queue and communicate directly with the resource database to allow instant allocation of resources.

**Blue Gene Agent** is the layer responsible for programmatically requesting Blue Gene/P partitions from Deep Cloud and integrating them into the CometCloud federated cloud. CometCloud then executes the different instances of the oil reservoir application on the allocated partitions. Blue Gene Agent is also responsible for releasing unused resources, thus providing an elastic cloud based on the number of pending tasks.



## 5. Experimental Setup

The experimental setup follows the application scenario described in Figure 1. The experiment starts by running the reservoir simulator forward with 100 initial ensemble members, where each requires 32-64 processors. The experiment will start using 50 partitions (32 processors each, a total of 1,600 processors (6,400 cores)). The user then has the option to increase the number of partitions to 100 to achieve a faster turnaround time (32 processors each, for a total of 3,200 processors, 12,800 cores). This part of the demonstration is intended to show the simplicity of use and dynamic scale up of the framework. In the next step, if the results of limited number (100) of ensemble members are not accurate (filter collapse), then the application will redo the step by dynamically increasing the number of filter ensembles to 150 members. In order to accommodate the increase in compute demand, the framework will dynamically scale up to the full potential of the IBM Blue Gene/P at Yorktown heights, NY (128 partitions, 32 processors each for a total of 16,384 cores). In addition the framework will also scale out to run dynamically on a second Blue Gene/P located in Saudi Arabia to run the remaining 22 ensemble members (22 partitions, 64 processors each, 5,632 cores). During the experiment, Blue Gene/P resources will vary from 1,600 to 5,504 processors (6,400 to 22,016 cores).

CometCloud is responsible for orchestrating the execution of the overall workflow, i.e. running the IPARS instances and integrating their results with the Ensemble Kalman Filter. Note that these application components are used as-is without having to modify them. Deep Cloud is responsible for the physical allocation of resources required to execute these tasks. The Blue Gene Agent monitors the size of the tasks in the CometCloud task pool and communicates with the Deep Cloud to obtain information about the current available resources. Using this information, the agent requests allocation of required Blue Gene/P partitions to Deep Cloud and integrates them into the CometCloud federated cloud. Note that partitions that are no longer required are deallocated. Once a set of tasks is complete, CometCloud runs the EnKF step on the results, and a new set of tasks is generated. The Blue Gene Agent then adjusts resources dynamically to accommodate the new set of tasks. The entire process is repeated until the final step is achieved, at which, all resources are freed and final results are returned to the user.

## 6. Team

**Moustafa AbdelBaky** is a Ph.D. Student at Rutgers University. **Hyunjoo Kim** is a Postdoctoral Associate at Rutgers University. **Manish Parashar** is a Professor at Rutgers University. **Kirk E. Jordan** is the Emerging Solutions Executive and Associate Program Director in the Computational Science Center at IBM T.J. Watson Research Center. **Hani Jamjoom** is a Research Manager at IBM T.J. Watson Research Center. **Vipin Sachdeva** is a Researcher in the Computation Science Center at IBM T.J. Watson Research Center. **Zon-Yin Shae** is a Researcher at IBM T.J. Watson Research Center. **James Sexton** is Program Director in the Computational Science

Center at IBM T.J. Watson Research Center. **Gergina Pencheva** is a Research Associate at the Center for Subsurface Modeling at The University of Texas at Austin. **Reza Tavakoli** is a Postdoctoral Fellow at the Center for Subsurface Modeling at The University of Texas at Austin. **Mary F. Wheeler** is Ernest and Virginia Cockrell Chair in Engineering at The University of Texas at Austin.

More information can be found at http://nsfcac.rutgers.edu/icode/