Prepared for: Office of Science, U.S. Department of Energy

pCloud: A Cloud-based Power Market Simulation Environment

A Final Scientific/Technical Report on the Phase I of the Small Business Innovation Research (SBIR) Grant, Award No. DE-SC0007472.
Principal Investigator: Aleksandr Rudkevich

Prepared by: Newton Energy Group LLC
47 Huntington Road, Newton, MA 02458
12/2/2012

Table of Contents

1 Summary

This research conducted by the Newton Energy Group, LLC (NEG) is dedicated to the development of *pCloud*™: a Cloud-based Power Market Simulation Environment. *pCloud*™ is offering power industry stakeholders the capability to model electricity markets and is organized around the Software as a Service (SaaS) concept -- a software application delivery model in which software is centrally hosted and provided to many users via the internet.

During the Phase I of this project NEG developed a prototype design for *pCloud* as a SaaS-based commercial service offering, system architecture supporting that design, ensured feasibility of key architecture's elements, formed technological partnerships and negotiated commercial agreements with partners, conducted market research and other related activities and secured funding for continue development of *pCloud* between the end of Phase I and beginning of Phase II, if awarded.

Based on the results of Phase I activities, NEG has established that the development of a cloud-based power market simulation environment within the Windows Azure platform is technologically feasible, can be accomplished within the budget and timeframe available through the Phase II SBIR award with additional external funding. NEG believes that *pCloud*™ has the potential to become a game-changing technology for the modeling and analysis of electricity markets. This potential is due to the following critical advantages of *pCloud*™ over its competition:

- Standardized access to advanced and proven power market simulators offered by third parties.
- Automated parallelization of simulations and dynamic provisioning of computing resources on the cloud. This combination of automation and scalability dramatically reduces turn-around time while offering the capability to increase the number of analyzed scenarios by a factor of 10, 100 or even 1000.
- Access to ready-to-use data and to cloud-based resources leading to a reduction in software, hardware, and IT costs.
- Competitive pricing structure, which will make high-volume usage of simulation services affordable.
- Availability and affordability of high quality power simulators, which presently only large corporate clients can afford, will level the playing field in developing regional energy policies, determining prudent cost recovery mechanisms and assuring just and reasonable rates to consumers.
- Users that presently do not have the resources to internally maintain modeling capabilities will now be able to run simulations. This will invite more players into the industry, ultimately leading to more transparent and liquid power markets.

2 Project Team

Aleksandr Rudkevich, Ph.D., Newton Energy Group, Principal Investigator

Evgeniy Goldis, Newton Energy Group

Konstantin Derenstein, Newton Energy Group

Charles Russel Philbrick, Polaris Systems Optimization, Subcontractor

Alec Goldis, WebOracle, Subcontractor

3 Glossary of Software Terms

Term	Definition
Azure Blob	A service for storing large amounts of unstructured data that can be accessed from anywhere using HTTP or HTTPS
Azure Management Service (AMS)	An Application Programming Interface (API) that provides programmatic access to help manage various Azure components such as Blobs, Queues, Azure Tables and Worker Roles
Azure Message Queue (AMQ)	A cloud-based queue service used to store messages that can be read by any application that is granted access to the queue
Azure SQL (Structured Query Language)	A service that provides a relational database management system for Windows Azure
Azure SQL Reporting Services	A cloud-based reporting service for Windows Azure built on SQL Server Reporting Services technologies. These technologies are used to host reports based on data from SQL Databases and allow users to view/download reports using just a browser
Azure Table	A service for storing large amounts of structured data. Azure Tables are ideal for storing structured, non-relational data
Microsoft ASP.NET	A web application framework that allows developers to build dynamic web applications and services
pManager	A cloud-based web application developed by NEG. This application manages interaction between multiple components of *pCloudi*™ and currently serves as an interface for developers
pWorker	A cloud-based, specialized console application developed by NEG. This application runs on individual Worker Roles and is responsible for running and managing multiple aspects of a power market simulation
Virtual Machine	A software implementation of a machine (computer) that executes programs like a physical machine
Windows Communication Foundation (WCF)	A framework for building service-oriented applications. WCF allows data to be sent asynchronously from one service to another.
WCF Endpoint	All communication with a WCF service occur through endpoints of that service. For the purpose of this report, the reader can think of a WCF Service that allows for two endpoints, pManager and pWorker, to communicate directly with each other
Worker Role	A virtual machine in the cloud. For the purpose of this report the reader can think of a Worker Role as a Windows machine preloaded with all necessary software to run power market simulators

4 Background

In January 2012 Newton Energy Group LLC (NEG) was awarded a Phase I Small Business Innovations Research Grant under the U.S. Department of Energy's Funding Opportunity Announcement DE-FOA-0000577 to develop *pCloud*^{TM}: a Cloud-based Power Market Simulation Environment. The grant has been awarded under Technical Topic 2: "Increasing Adoption of HPC Modeling and Simulation in the Advanced Manufacturing and Engineering Industries," Sub-topic a: "Turnkey HPC Solutions for Manufacturing and Engineering."

NEG, a start-up software and consulting company, commenced its commercial operations in March of 2012 with the development of *pCloud*^{TM}. In November 2012, the project team successfully met all Phase I objectives. The company's achievements during Phase I are summarized in this report.

5 *pCloud* and Power Market Simulators: Opportunities and Challenges

pCloud is a simulation environment offering power industry stakeholders the capability to model electricity markets and is organized around the Software as a Service (SaaS) concept -- a software application delivery model in which software is centrally hosted and provided to many users via the internet.

Power industry stakeholders are constantly faced with the ever-growing complexity when trading, investing or making policy decisions over time horizons spanning minutes to years. They understand that special features of power markets create unique analytical opportunities but also pose substantial challenges.

Power markets are built on a convergence of the laws of physics and economics. Economics define the order of generator dispatch and therefore electricity prices, while the laws of physics determine the flow of energy in the transmission network. Power system operators rely on *market engines*, which constantly solve specialized large-scale optimization problems based on a model representation of the economics and physics of the power system.

Power market simulators, replicate the logic of market engines and make it possible for market participants to evaluate likely future market outcomes, such as physical performance of generating plants, flows of power and locational prices. These simulators provide system planners, project developers, investors and consulting companies with the capability to rely on the detailed and unambiguous bottom-up physical/economic analysis of power markets not typically available in markets for other goods and services.

Power market simulators are complex and highly CPU intensive due to the size of the system (the Eastern Interconnection has over 5,000 generators and 70,000 transmission branches), the non-continuous and multi-period nature of generation scheduling, and the complex nonlinear behavior of power systems.

Power market simulators span time horizons ranging from minutes to years and need to reflect the uncertainty and operational characteristics of:

- Variable generating resources such as wind and solar
- Real-time price responsive behavior of consumers
- Fuel price volatility
- Planned and forced outages of generation and transmission facilities
- Addition and retirement of new generation units
- Grid expansion
- Implications of existing and future environmental policies

To address uncertainty, power industry stakeholders process multiple market-operation scenarios, but due to time constraints and expensive computational resources, only a handful of scenarios are generally analyzed. Under most circumstances Monte-Carlo models are impractical.

The high expense of simulators are limiting even to those entities that could afford to pay high licensing fees, support hardware infrastructure and maintain dedicated professional staff.

6 Potential Public Benefits of *pCloud*

Development of *pCloud* will create significant technical and economic benefits to the power industry, its stakeholders and, in the long run, to all consumers of electricity. As described below, *pCloud* will uniquely benefit each group of industry stakeholders.

By using *pCloud*, power system planners and operators, such as Regional Transmission Organizations and vertically integrated Transmission Planning Authorities, will more efficiently assess costs and benefits of alternative transmission and generation expansion options leading to better designed, more robust and ultimately less costly solutions.

Similarly, investors and generation and transmission project developers will benefit from market reports prepared with less turn-around time but with much higher quality due to analysts' ability to evaluate and quantify a much wider range of scenarios and better address upside and downside options associated with the investment at hand. This will reduce the investment risk, lower transaction costs and ultimately translate to lower electricity costs borne by consumers.

U.S. Federal agencies such as the Department of Energy (DOE), Federal Energy Regulatory Commission (FERC) and Environmental Protection Agency (EPA), State Public Utilities Commissions and Consumer Advocates will benefit from *pCloud*, which will help to evaluate alternative energy and environmental policy decisions, to perform power market monitoring and to access electric utility and consumer protection regulations. Availability and affordability of high quality power simulators, which presently only relatively large corporate clients can afford, will level the playing field in developing regional energy policies, determining prudent cost recovery mechanisms and assuring just and reasonable rates to consumers.

Traders of electric power based commodity and derivative products such as power futures and Financial Transmission Rights will significantly increase their ability to accurately assess the risk of their trading positions through conducting Monte-Carlo analyses with highly detailed simulation models by leveraging cloud computing capabilities. This will improve the liquidity of markets for commodity and derivative products, reduce price volatility and market risk and will ultimately improve the efficiency of the market.

Industry consultants will benefit from *pCloud* by focusing on directly serving their clients without the need to maintain expensive hardware and software for power market simulations which typically only a small set of consulting companies can afford. Access of a wide group of consulting companies to quality modeling tools will improve the quality of debate within the industry.

Developers of power market engines will benefit from *pCloud* by being able to compare and benchmark the performance of developed algorithms. This will facilitate the on-going improvement of market engines.

For all the user categories above, *pCloud* will significantly reduce the barrier to entry into power market modeling. Users that do not have the resources to internally maintain modeling capabilities will now be able to run simulations. This will invite more players into the industry, ultimately leading to more transparent and liquid markets.

7 Expanding the Project Objectives for Phase I and Meeting Objectives as Originally Stated

7.1 NEG's Vision of *pCloud*

Newton Energy Group (NEG) envisions *pCloud* as an electric power market simulation environment organized around the Software as a Service (SaaS) concept. SaaS is a software application delivery model in which software is centrally hosted and provided to many users via the internet.

Power market simulations are ideal candidates for parallel processing due to a weak interdependence of events. The decision cycle for power markets is typically within a two-day range and rarely exceeds a period longer than a week. Well-partitioned simulations can be processed in parallel, fitting perfectly within the cloud-computing framework.

pCloud users will take advantage of the computational power available on the cloud and analyze hundreds or even thousands of scenarios in a timeframe of minutes to several hours. *pCloud* will provide clients with standardized access to state-of-the-art power market simulators and support data structures and analytical tools organized around the Windows Azure environment.

Conceptually (and simplistically), *pCloud* is depicted on Figure 1below.

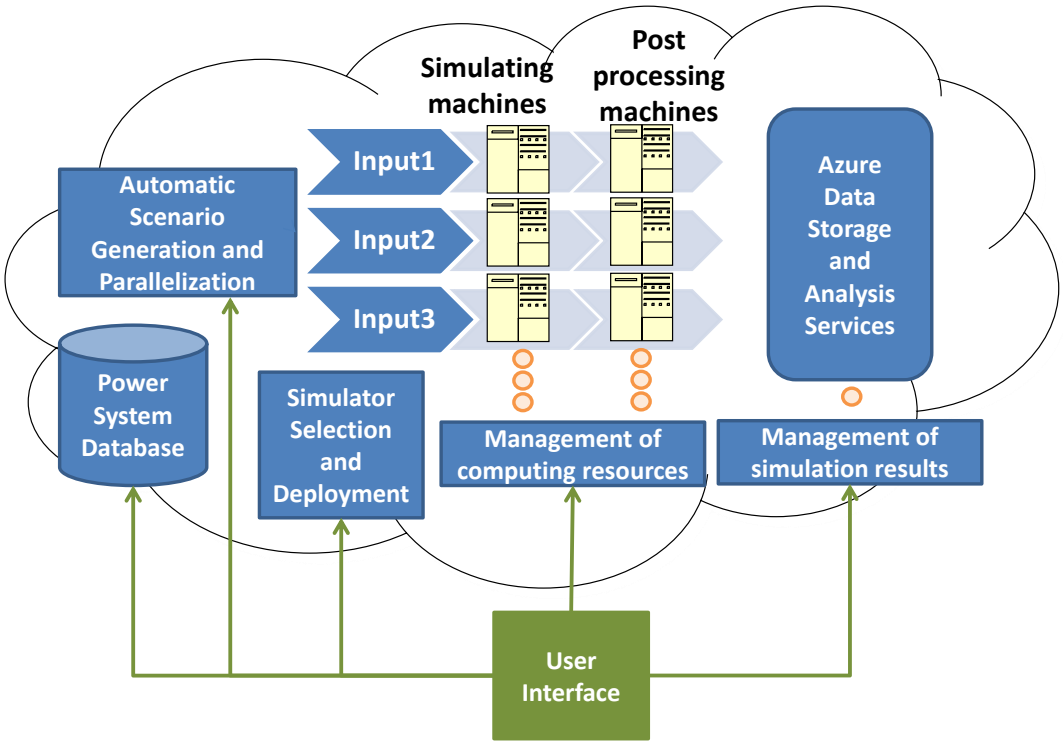

Figure 1. Conceptual Representation of *pCloud*

As shown in Figure 1, users of *pCloud* have:

- Access to multiple power market simulators supplied by third parties
- Access to simulation ready power systems data
- The capability to control the creation and execution of simulation scenarios
- Ability to analyze manage simulation results
- Capability to manage computing resources.

7.2 *pCloud* Key Components

7.2.1 Windows Azure

Windows Azure Cloud Services provides an effective environment for distributed computing applications such as *pCloud*. Windows Azure handles Worker Role (virtual machines in the cloud[1]) deployment details from provisioning and load balancing to health monitoring for continuous availability. Azure based applications are backed by an industry leading 99.95% monthly availability guaranteed by the Microsoft Service Level Agreement (SLA).

Using the Azure environment, *pCloud* will allow customers to use their current on-premises Windows accounts for authentication and authorization. This will provide for a single sign-on experience across all *pCloud* services while maintaining the same high level of security that is required within their organization.

Additionally, the Azure environment offers a large variety of tools to efficiently store and manage large data sets in a distributed environment; seamlessly share data between the cloud and on-site applications; automate database backups; and a breadth of other features.

7.2.2 Power System Optimizer (PSO): a Market Simulator

During the Phase I of this project, NEG worked with only one power market simulator, Power System Optimizer (PSO). PSO is based on the same logic and technology that is used in actual market engines in PJM, Midwest ISO and ISO New England and is provided by Polaris Systems Optimization, Inc. (Polaris). Dr. Philbrick, a Founder and President of Polaris, is an active *pCloud* team member.

PSO accurately captures the economics and operations of the power grid in an unparalleled level of detail. PSO offers a number of unique features that are currently unavailable in most competing simulators. Through the use of a rolling-horizon approach, PSO captures the impact of operational decisions on time scales ranging from minutes to years. Modeling multiple decision cycles, PSO replicates the actual process by which grid operators respond to time-varying market conditions. It has logic for addressing the impact of uncertainty on operational decisions and accounts for the important contributions that flexible resources make through their ability to respond to changing system needs.

PSO algorithms are based on Mixed Integer Linear Programming (MILP) formulations allowing the computation of true optima and enabling detailed modeling of complex operation modes such as those for combined-cycle power plants.

[1] NEG will use the terms Worker Role and machine interchangeable throughout this report.

7.2.3 AIMMS & Gurobi: Modeling Language and MILP Solver

The majority of the currently-deployed market engines (and *pCloud* simulator, PSO) are built using the AIMMS modeling language and MILP solver. AIMMS is an advanced commercial modeling tool provided by Paragon Decision Technologies (PDT).

The Gurobi Optimizer is a state-of-the-art solver for many types of mathematical programs including the MILP used by PSO. Benchmarks consistently show Gurobi finding both feasible and proven optimal solutions faster than competing solvers.

7.3 Expanding the Phase I Objectives

As outlined in the Phase I proposal, the major goal of this phase was to determine an efficient way to use cloud computing to reduce turn-around time for power market simulations, lower simulation costs for end users and solve power market problems that were previously infeasible. From a technical standpoint, NEG focused on the architecture and implementation of scenario parallelization and management of computing resources. However, as the project evolved, it became apparent that NEG would need to properly assess the entire SaaS architecture to develop a technologically and commercially successful product. As a result, NEG substantially expanded the scope of Phase I objectives beyond those initially proposed. The team's accomplishments can be placed in the following categories:

1. A prototype design for *pCloud*™, a SaaS-based commercial service offering
2. Development of the system architecture supporting the design of SaaS-based commercial service and implementation of critical elements of the system architecture to ensure the feasibility of key elements of the developed architecture
3. Outline of the business processes for the on-going support of *pCloud*
4. Obtaining necessary certifications from the Federal Energy Regulatory Commission (FERC) granting NEG personnel access to Critical Energy Infrastructure Information (CEII) which is essential for the informational support of *pCloud*
5. Formation of key technological partnerships and negotiation of commercial agreements with partners, which give the *pCloud* service a competitive advantage over prevailing technologies both technologically and economically
6. Outreach to potential clients representing different types of power industry stakeholders and confirmation of a significant interest on their part in services that could be offered via *pCloud*
7. Securing continuing private funding for *pCloud* development to make sure that development of this technology continues between the end of Phase I and beginning of Phase II, if awarded.

7.4 Meeting the Original Phase I Objectives

In this section of the report, we summarize the results of NEG's activities pertaining to the original goal: implementation of critical elements of the system architecture to ensure the feasibility of key elements of the developed architecture, activities primarily under the second category of the above list. Per the Phase I proposal, NEG planned to create a prototype software application that would address four key objectives:

- Partitioning of modeling scenarios
- Data processing and storage
- Managing Worker Roles
- Integrating individual components into a single application.

These objectives have been successfully met as summarized in Table 1below.

Objectives	Goals	Phase I Results
Partitioning of modeling scenarios	Market simulations over sufficiently long time horizons could be performed independently. The goal was to determine an optimal way to divide the modeling time window depending on the problem size and users' economic constraints.	NEG created a test model and explored various partitioning algorithms. A heuristic was implemented that balances cost and performance. This objective is closely tied with the Managing Worker Roles objective
Data processing and storage	NEG expects that the data created by *pCloud* will be orders of magnitude greater than what the industry currently deals with The goal was to determine how and in what format to store model related data, taking economic considerations into account	NEG developed a three-tiered data storage and management structure using 1) Blobs for raw data produced by the simulator in CSV format, 2) Azure Tables and CSV formats for aggregated results; 3) On-demand created SQL Azure relational database for querying aggregated results. The selected structure provides the necessary balance between usability, efficiency and the economics of storing and managing data on the cloud.
Managing Worker Roles	Managing thousands of Worker Roles (virtual machines in the cloud) in real-time will be a complicated process, requiring algorithms to cost-effectively scale the number of machines and the processing power allocated to each of them The goal was to determine a set of metrics that could be used to effectively manage the number of Worker Roles	NEG learned that managing the number of Worker Roles is closely related to the methodology used to partition the modeling time window NEG developed a heuristic to spawn and kill Worker Roles based on characteristics of those Worker Roles and estimated run times of individual model partitions. NEG plans to continuously improve the Worker Role management logic as actual *pCloud* use data becomes available

		through beta-testing and commercial usage.
Component Integration	*pCloud* relies on 3rd party components such as market models, solvers and mathematical programming environments The goal was to integrate the three external components used in Phase I within a single application that would allow all components to communicate needed information amongst each other. Additionally, the *pCloud* architecture needs to be general enough to accommodate additional components (other market models, for example) in the future	*pCloud* presently integrates the market simulator (PSO), mathematical programming environment (AIMMS) and optimization solver (Gurobi) into a single cloud-enabled application. NEG is able to call specific functions from each component to retrieve error and warning messages as well as to check on the execution status. NEG intends for *pCloud* to support multiple market simulators and created a design that allows for their integration with the least amount of effort from both NEG and external vendors.

Table 1. Meeting Technical Objectives of Phase I.

8 Description of Phase I Project Activities

In this section of the report, we describe NEG's activities in all seven areas identified above.

8.1 A prototype design for *pCloud* as a SaaS-based service offering

From the outset of this project, NEG envisioned *pCloud* as a SaaS-based service offering. Although Phase I was primarily focused on testing the technical feasibility, it quickly became apparent that the system architecture would not be adequately developed without a clear understanding of the structure of commercial services in will support. Therefore, the prototype design of *pCloud* as a SaaS-based service has been developed in parallel with the design and testing of the software system architecture. This high level design of *pCloud* with all anticipated service offerings is depicted in Figure 2 and briefly discussed below. It is important to note that most of this design is yet to be implemented and the underlying architecture will likely undergo some revision.

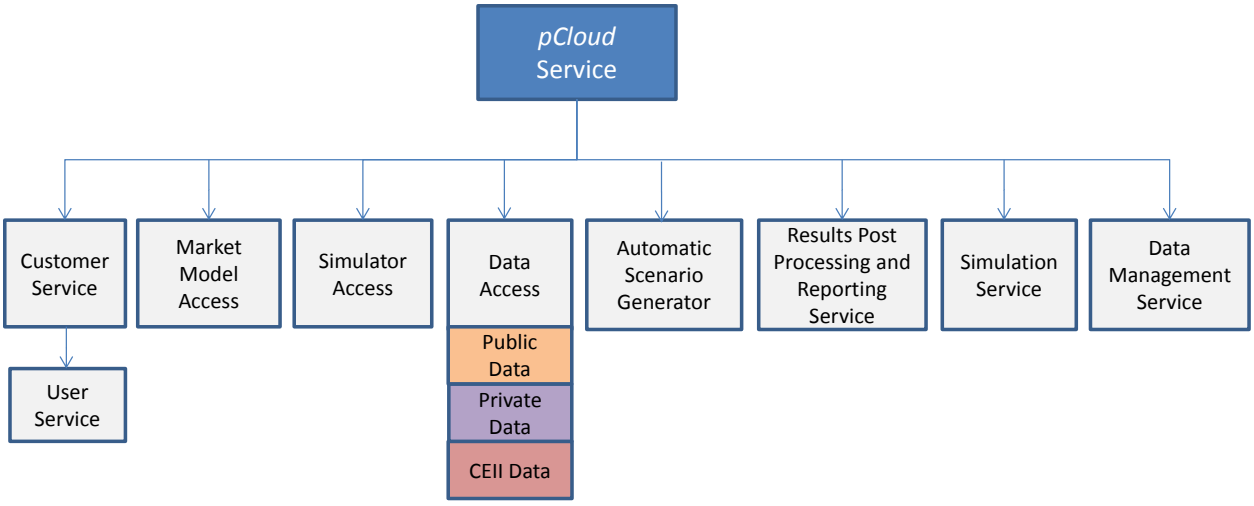

Figure 2. Major Components of the pCloud Service

8.1.1 Customer and User Services

Customers are entities that buy the *pCloud* service. They determine the type and level of service required and identify users - individuals within their organization who should receive access to the service. Customers are financially responsible for the actual usage of *pCloud*. Essential components of the Customer Service are presented in Table 2 below. Essential components of the User Service are presented in Table 3.

Customer Service	Description
Authentication	Authentication system for the customer representative
CEII Certification Verification	Verifying whether the users authorized by the customer representative are certified to have access to CEII information and granting/not granting such access to CEII protected data

	through *pCloud*
Service Selection and Subscription	Mechanism for the customer to subscribe for specific *pCloud* services
User management	Mechanism for adding/deleting users associated with the Customer's organization and managing their permissions.
Usage reporting	Periodic, or on demand reporting of usage metrics (e.g. processor-hours, use of storage space by type, use of other cloud-based services) and associated financials. Reports could be generated by Project, User, Time period, Regional model, etc.
Billing and invoicing	Service of generating and sending invoices to the Customer for services rendered, tracking AR and outstanding invoices.

Table 2. Essential Customer Service Components

User Service	Description
Authentication	Authentication system for authorized users
Project Access and Management	Providing users with access to existing or new projects they are authorized to work on by the Customer
Usage reporting	Periodic or on demand reporting of usage metrics (e.g. processor-hours, use of storage space by type, use of other cloud-based services) and associated financials. Reports could be generated by Project, Time period, Regional model, etc.

Table 3. Essential User Service Components

8.1.2 Market Model Access

pCloud will offer users access to pre-set, ready-to-run models of regional markets. The term "model" in this context refers to the combination of all input data and model run parameters which are necessary and sufficient to perform a long-term simulation of the selected regional market under a set of plausible assumptions.

The user will be able to choose the market model that is most suitable to the problem at hand. Each model could be customized. The degree and level of detail to which the model could be customized will vary depending on the type of service subscribed to by the customer.

8.1.3 Data Access

Data Access will allow users to review, analyze and modify input data associated with a particular model.

As presented in the diagram, there are three categories of data:

- Public data – information assembled by NEG from a variety of public and commercially available sources
- Private data – information provided by the user which will be made available only to other users which have access to the same project, but not visible to other users within the same customer organization, nor to other customers
- Critical Energy Infrastructure Information (CEII) data – information received and processed by the NEG personnel which are subject to the Non-Disclosure Agreement with the US Government

In general, all users will have access to public, model-ready data. Private data will be provided by users themselves. An example of such information may include user-defined characteristics of generating plants, fuel price forecasts or power flow models. *pCloud* will protect this information by making it non-visible and non-accessible to users without appropriate permission.

CEII data will be visible to only those users who have provided NEG with appropriate documentation proving that they have Government approved access to that information. Users with no CEII certification would still be able to perform simulations relying on CEII information; however, these users will not be able to see certain data inputs or obtain certain output results.

8.1.4 Simulator Access

pCloud will offer users a choice of simulation engines standing ready to be applied to the same (or almost the same) market models. While the core aspects of market models will be simulator independent, certain model elements will be simulator-specific, reflecting differences in algorithms and data structures. During Phase I, NEG worked exclusively with PSO, but has already negotiated the inclusion of PROBE, simulator provided by PowerGem, into the *pCloud* service offerings.

8.1.5 Simulation Service

To start a simulation, a user must first create a Project. A Project will represent all simulations that intend to solve the customer's problem at hand. For example, in the consulting framework, a Project will be created for a particular engagement the consulting company has with a client. Each Project is further subdivided into Tasks, which may reflect a certain staging of the analysis performed by the customer and will likely be handled by an individual user. Tasks are made up of the individual Scenarios (simulations) that a user will perform. Each scenario will then be partitioned into Segments by the Simulation Service and deployed across the Azure Cloud.

It is important to note that these segments originate from multiple customers, each representing multiple users who themselves are working on multiple projects and scenarios.

The simulation service begins when scenarios are formulated, inputs are generated and the user submits a simulations request. The simulation will be processed in four major steps:

1. Parallelization. Each submitted scenarios is split into a set of segments for concurrent execution.

2. Diagnostics. Each segment is diagnosed for errors. Messages are generated. Based on the results of the diagnostics and user defined policy, the segment could be submitted for processing or rejected. The user is notified accordingly.
3. Segment Processing. At this step, *pCloud* provisions Worker Roles necessary for the execution of each segment, runs simulator software and generates output results.
4. Segment merging. When all segments representing the same scenario are completed, *pCloud* merges output results into a scenario-specific set of output files and stores them in Blobs on the cloud.

The simulation service records the duration of the simulation of each segment along with a wide variety of problem size characteristics and algorithm performance metrics which will be used for future improvements of *pCloud*.

8.1.6 Automatic Scenario Generation

Scenario generation and management is a critical feature of *pCloud*. The capability to simulate a large number of scenarios must be paired with the capability to formulate such scenarios, automatically generate associated input datasets, and post process completed simulations. *pCloud* will provide a flexible and intuitive interface for building scenarios through a combination of user selected, pre-defined and Monte-Carlo based approaches.

A scenario is a meaningful combination of input and modeling assumptions defining a *pCloud* simulation spanning over a defined period of time. Table 4 below presents two relatively simple examples of the categories of scenario formation options available to the user of the *pCloud*.

Scenario Formation Option	Description
System expansion assumptions • **Generation** • **Demand-side resources** • **Transmission**	User is offered a choice of multiple alternative options for future generation expansion, upgrades, retirements, transmission expansion and demand response resources. User is provided with a list of options by type. For a given scenario, user selects options to include into simulation, time in service and location (by bus ID). In addition, *pCloud* should provide an interface through which resource (generation and demand-side) and transmission expansion alternatives could be imported into the model. Transmission expansion alternatives are imported along with the expanded load flow model. User submit a populated table of resource options which are added to pre-defined system expansion alternatives, which are then used jointly for scenario development purposes.
Forecasts • **Fuel prices** • **Emission allowance prices** • **Peak demand and energy use**	An interface through which alternative forecasts of various inputs could be imported into the model. Forecasts may include peak demand and energy use forecasts by region or service territory, fuel price forecasts, emission allowance prices.

Table 4. Examples of Scenario Formation Options

pCloud will provide a rule-based scenario generating engine for creating scenarios using the above identified formation options. Through this approach, the user will be able to define scenarios through various combinations of options that jointly satisfying a set of rules. When the user selects such combinations, *pCloud* will generate all corresponding scenarios. The user would be able to further augment the rule or adjust the resulting list manually by removing undesired or manually creating additional scenarios. The user could additionally "record" this process and re-use it in future work.

When a set of scenarios is formed, *pCloud* will automatically generate input datasets required for simulating each of the defined scenarios.

8.1.7 Results Post Processing and Reporting Service

pCloud will offer a highly sophisticated structure of data management, providing users with efficient and flexible access to simulated data, while controlling data storage costs. A single scenario can generate from several to several dozen GB of output results. Raw results are stored in Azure Tables or comma separated value (CSV) files – a low cost storage options for large volumes of data. These data are stored in Blobs on the cloud as long as the user needs to keep them.

Typically, the scope of reported results is much greater than the user initially needs. Automated post processing generates a set of summaries in CSV or Azure Tables format. These summaries are user requested aggregations (sums, averages, etc.) of power market indicators reported by PSO.

8.1.7.1 Aggregation Scopes

Output results are characterized by temporal and spatial granularity. Temporal granularity determines the smallest time step over which the output data are defined (produced by the simulator).

Spatial granularity determines the smallest footprint over which the output data are defined (produced by the simulator).

Typical temporal aggregations are defined as a combination of the calendar period and time-of-use. Calendar periods:

- Day (24 hours)
- Week
- Month
- Season
- Year
- Full scenario horizon

Time of Use:

- Round the clocks (all hours within the calendar period)

- OnPeak (on-peak hours within the calendar period, generally 16 hours during weekday. The selection of the on-peak hours varies by the market)
- OffPeak (off-peak hours within the calendar period, generally 8 hours during weekday and 24 hours during the weekend and NERC holidays)
- OffPeakWD (off-peak hour during weekdays only)
- Weekend (all hours during the weekend and NERC holidays)
- OnPeakWE (on-peak hours during the weekend and NERC holiday)
- OffPeakWE (off-peak hours during the weekend and NERC holiday)

Typical spatial aggregations are defined across the following groupings of electrical buses:

- Pool or Market
- NERC region
- Area
- State
- Generation owner
- Generation technology
- Generation fuel

8.1.7.2 Aggregation Types

Users could submit customized criteria for creating temporal and spatial aggregations.

Typical aggregation types produced by *pCloud* on demand include:

- Sums
- Averages
- Standard deviations
- Maximums
- Minimums
- Percentiles

8.1.7.3 Aggregation across Scenarios

All aggregation types across all aggregation scopes could be computed across scenarios pertaining to the same project. Table 5 below shows a small subset of data that could be aggregated within and across scenarios. The total number of such data items is on the order of 60 to 70.

Data Category	Temporal Scopes	Spatial Scopes	Types
Demand + Losses	All	Area, Pool, NERC region	All
Locational Marginal Price (LMP)	All	Area, Pool, NERC region	All except Sums
Reserve Price by	All	Reserve Area, Pool,	All except Sums

Type of Reserve		NERC region	
Fuel Costs by fuel type	All	All	All
Emission costs by type	All	All	All
Energy Revenue	All	All	All
Constraint flow	All	All	All
Congestion Rent	All	All	All

Table 5. Sample Output Data Categories and Applicable Aggregations

The user can further decide to export a selection of aggregated summaries into a SQL Azure database, which *pCloud* will create on demand. While this option is more expensive, users will gain access to powerful tools such as Azure SQL Reporting Services, Excel's PowerPivot and customizable SQL queries to meet their business analytic needs.

8.1.8 Data Management

pCloud provides users with an interface to manage input data, raw output results, aggregation tables, SQL Azure databases and other files created on the cloud.

The interface can allow the user to download, archive or delete selected data directly, set up data management schedules and decision rules.

8.2 *pCloud* System Architecture and Implementation of Critical Elements in Phase I

The design described above encompasses all the services that NEG envisions for *pCloud*. During Phase I, NEG focused its development efforts on the Simulation Service. However, portions of other services were implemented to ensure a successful proof of concept. The architecture below details the software level implementation in which only certain elements of the *pCloud* service depicted in Figure 2 have been realized.

The *pCloud* Simulation Service is built for Windows Azure and uses Azure tools for managing computing resources (referred to as Worker Roles or machines) and user authentication and authorization. *pCloud* provides a Web Interface and a set of business rules that facilitate creation and execution of scenarios broken up into model segments. A model segment is identified by the model simulator (PSO in Phase I) and a set of input files required by that simulator. The light green blocks in Figure 3 below indicate portions of the system design that were developed for Phase I. Figure 4 then describes the Simulation Service architecture in detail.

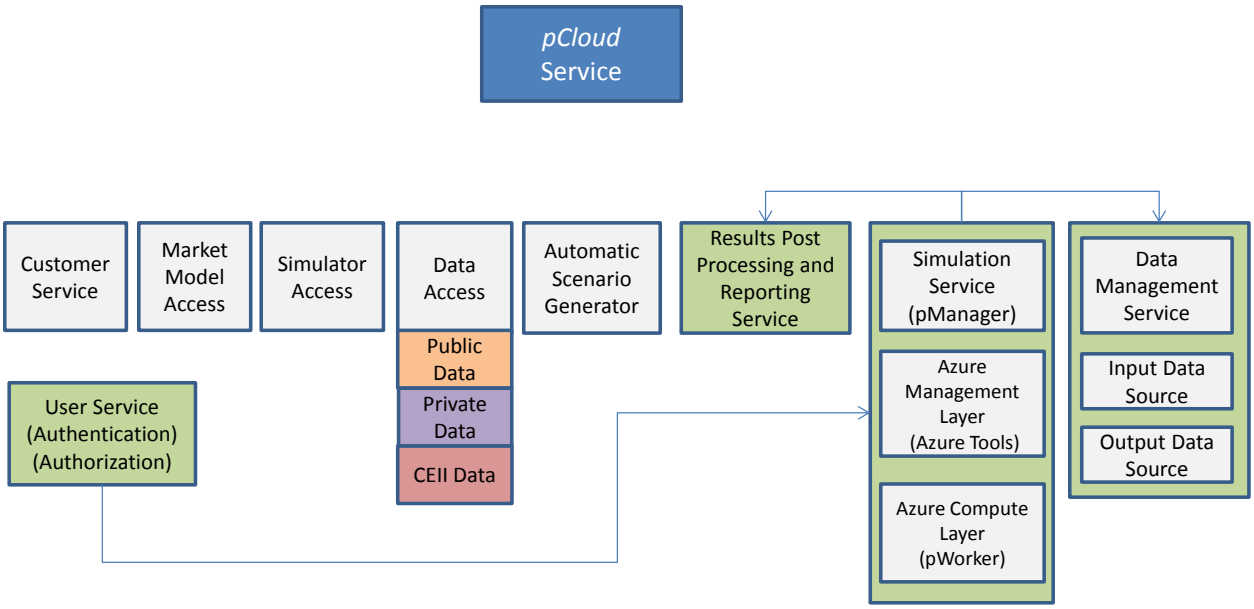

Figure 3. Developed components (light green) of the pCloud Service

In the final product, users will communicate with all *pCloud* service components. For now, only a basic developer interface was implemented for accessing the Simulation Service components. However, NEG also partially implemented or conducted feasibility tests for other services to evaluate alternate architectures.

Figure 3 shows that the realization of the Simulation Service consists of three main components:

- The *pCloud* Manager (pManager) - the front end web application, which also serves as the user (developer) interface for the Simulation Service which directly communicates with
- Azure Management Layer – Layer comprised of an NEG developed Windows Communication Foundation (WCF) endpoint[2] (a framework that allows asynchronous data transfer between services) and Microsoft Azure tools such as the Management Service, Message Queue, Tables and Blobs
- Azure Compute Layer – Layer comprised of Worker Roles used to execute simulations and managed by NEG's pWorker application, a specialized console application.

These three components are expanded in Figure 4 below

[2] WCF is a runtime and a set of APIs in the .NET Framework for building connected, service-oriented applications.

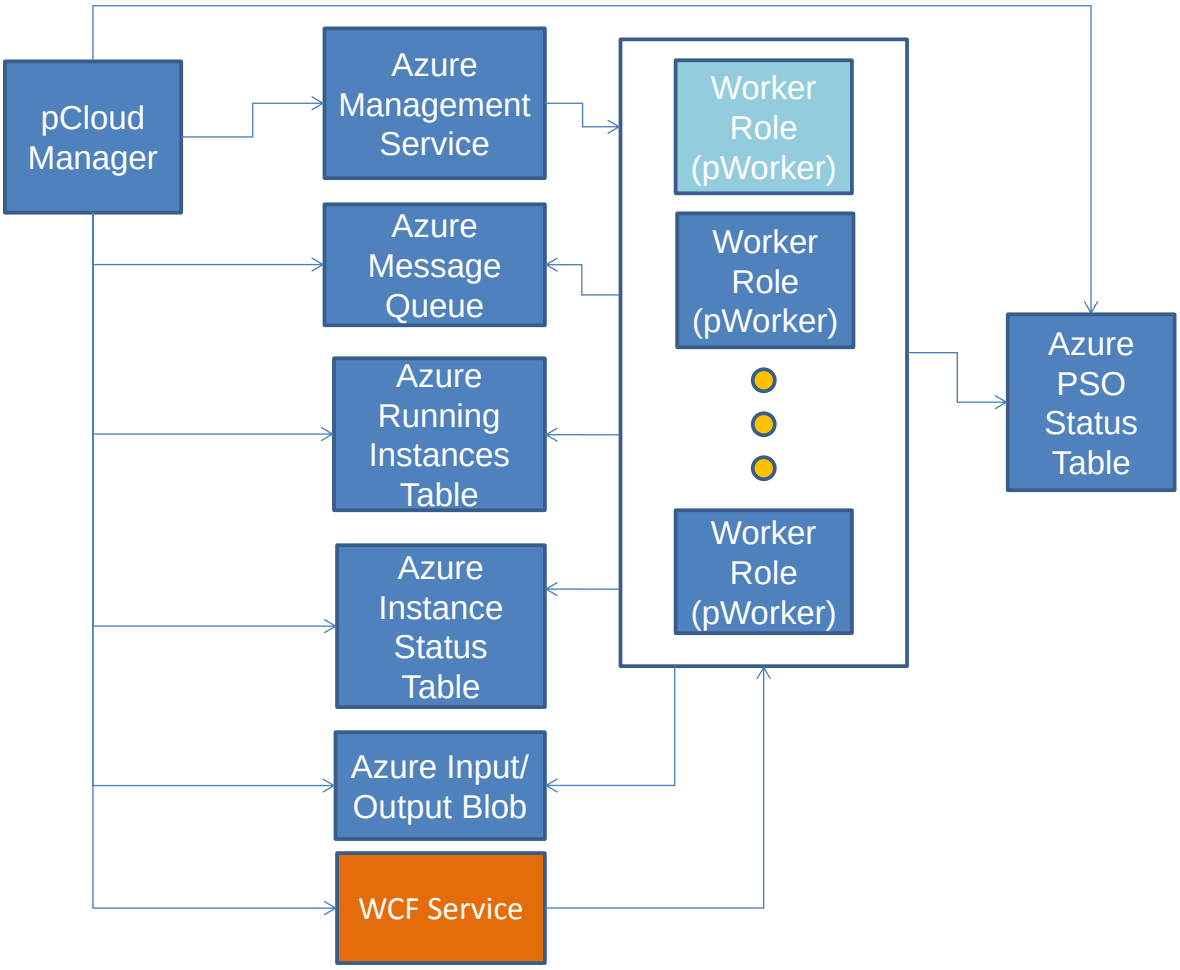

Figure 4. Simulation Service Architecture

The components in Figure 4 work together to accomplish the four Simulation Service goals described in the system design

- Scenario Parallelization – the user interacts with a pManager interface, which communicates with the Azure Management Service and Message Queue to parallelize a scenario into segments
- Segment Processing – segments are processed by Worker Roles; the Azure Management Service deploys the pWorker application to each Worker Role where the application sits and communicates with the Azure Message Queue and an Azure Table that NEG created to help track assignment of segments to Worker Roles (Azure Running Instance Table). The pWorker application also acquires model segment data from appropriate Input Blobs.
- Worker Role Management – With the potential to have thousands of Worker Roles simultaneously processing model segments for multiple users, effectively managing these Worker Roles is clearly important to ensure a seamless scenario execution process. To that end, pWorker tracks each Worker Role's performance and stores this data in another NEG created Azure Table (Azure Instance Status Table). Additionally, NEG created a WCF

endpoint on each Worker Role that allows the user to communicate directly with specific Worker Roles

- Simulator Diagnostics – pWorker tracks logs written by PSO and stores relevant messages in an Azure PSO Status Table
- Segment Merging – pManager merges output files from each segment into a single set of output files for a scenario.

The implementation of each goal is described below, which is followed by a discussion of work done on the Data Management, Results and Reporting and User Services.

8.2.1 Scenario Parallelization

8.2.1.1 Architecture and Implementation

As discussed earlier, a scenario represents a request to simulate operations of a certain power market over a user-defined period of time – the scenario's horizon. Depending on the user's specification, the scenario horizon may span several hours or several years with an internal time step ranging from seconds to hours. The parallelization is performed by segmenting the scenario horizon into smaller subsequent time periods and processing each segment independently. To assure continuity of boundary conditions, subsequent segments are made to overlap with some smoothing of overlapping results.

The *pCloud* Manager (pManager) is the front end application, currently serving as the user interface for the Simulation Service. Through pManager, the user submits requests to process scenarios. For each scenario, the user specifies a segmentation rule.

pManager prepares a scenario for parallelization by placing a separate message in the Azure Message Queue (AMQ) for each model segment. Based on the segmentation rule, pManager communicates to the Azure Management Service (AMS) the number of Worker Roles required for each scenario. This is done by first reading a configuration file (a file used by the Azure environment to track general information on Worker Roles) from the AMS, modifying it to reflect the new number and types of machines that need to be created and writing the file back to the AMS. The AMS then appropriately scales the number of Worker Roles per this file.

8.2.1.2 Business Rules for Scenarios Parallelization

At the start of the project NEG assumed that scenario segmentation rule could be solely determined by the mathematical formulation of the scenario – size of the system, scenario horizon and the length of the time step. Once NEG conducted research on the process to create Worker Roles, it became clear that partitioning the modeling time window should be directly linked to the approach for managing Worker Roles. As a result, an efficient parallelization algorithm must depend not only on the structure of the scenario itself, but also on the timing and economics of Worker Role management and on the in-flow of scenario processing requests received by *pCloud* from all users. With the completion of Phase I, NEG is able to partition a simulation using both an automated and user specified approach taking into account the time required to spawn and kill new machines.

New machines take 7-10 minutes to spin-up and Microsoft charges for them on the basis of whole hours. Therefore, once a machine is spawned, it is always economic to keep it running for hourly blocks. The spin-up time and pricing policy provide a natural guide for when to spawn new machines and for how to divide the modeling time window. Working with a test model, assembled for Phase I, NEG evaluated various options and implemented the heuristic below, which captures the practical implementation questions that NEG aimed to address:

1. Based on pre-established statistics on problem size vs. segment run time, partition each simulation into lengths of time that would run for approximately one hour

2. Using simulation progress tracking, implemented in Phase I, determine whether remaining run time is less than the 10 minutes required to spawn a new machine. All machines that have less than 10 minutes remaining are kept available for the next simulation (provided that there are enough new segments to utilize these machines) while remaining machines are killed once they complete the current simulation.

While this is a simple heuristic, it is effective at keeping steady processor utilization, which in turn reduces costs. The statistics used in step 1 are based on NEG's experience with running similar simulations. As NEG gains more experience with running various problem sizes on the cloud, more statistics will be collected, which in turn will drive the refinement of existing heuristics. This heuristic will keep the expected turn-around time for the user request at approximately 1 hour. Some users may have a preference to further shorten this turn-around time, increase the number of segments and be willing to pay additional costs of Worker Roles staying idle for the remaining part of the hour. Business rules for processing (and pricing) such requests will be determined during Phase II, if awarded.

8.2.1.3 Dealing with Boundary Conditions

Scenario parallelization should reliably produce adequate results. Splitting a modeling time window creates certain mathematical problems related to boundary conditions at the start and at the end of the segment's time period due to inter-temporal constraints imposed on power generating capacities such as minimum times between starts and shut-downs. Generator's state (on or off) should be set at the beginning of each simulation period. Moreover, when system is optimized over a finite time horizon, the optimization algorithm must be made aware of the fact that the end of the time horizon is not the end of the world and it is not necessary to deplete all storage inventory and shut down every generating unit.

A standard approach in power systems modeling for dealing with these issues is to split a scenario into segments with overlapping time horizons and make generic assumptions with respect to the most likely operational state of each generating unit at the beginning of each horizon (for example, a nuclear unit will likely be on while a gas turbine peaker will likely be off). If the simulation begins sufficiently long before the nominal start time of the segment and ends sufficiently long after the nominal end time of the segment, the results between the nominal start and nominal end times of the segment appear virtually identical to results obtained for the same period from the entire

scenario simulated sequentially. Based on our past experience with these problems, typically an overlap that is equal or greater to the length of the largest inter-temporal constraint is sufficient for multi-day simulations conducted with a time-step of 1-2 hours. To verify our initial assumptions with regard to overlaps along with the performance of the *pCloud* parallelization and processing, NEG conducted four series of numerical experiments with the test power market built over the IEEE 118 bus model in which no inter-temporal constraint exceeds 24 hours. In the first series the system was simulated sequentially without parallelization over a one-year horizon with the ultimate time step of 1 hour. In the second experiment, the system was split into 52 segments with no overlap. In the third experiment, the system was simulated with an extra 24 hours before the nominal start time, and 24 hours after nominal end time of the segment. The fourth experiment was performed with 48 hours overlapping periods. The tests statistics are summarized in Table 6:

Test Description	Average Segment Run Time with machine Setup(min:sec)	Longest Segment Run Time with Machine Setup (min:sec)	Number of machines used
1 year simulation w/o parallelization	172:00	172:00	1
1 year simulation split into week long segments w/o boundary conditions	10:16	11:45	52
1 year simulation split into week long segments with 1 extra day included at both ends of the segment time window	10:44	12:28	52
1 year simulation split into week long segments with 2 extra days included at both ends of the segment time window	11:17	13:13	52

Table 6. Parallelization Statistics

All four experiments were run simultaneously using a total of 157 Worker Roles so that all three parallelized cases completed in less than 14 minutes while the non-parallelized scenario took 2 hours and 54 minutes to finish. Parallelization achieves a 10x speedup for the 118 bus system. For a real system with thousands of busses, this speedup would be significantly greater. A single segment for an Eastern Interconnection (EIC) scenario could average 3 hours. Un-parallelized, this scenario would take 156 hours to run. With parallelization and assuming a 10 minute startup time per machine (for both configurations), the average time per segment is 3 hours, 10 minutes. This would result in an average 52x speedup. Even with a small model, there are advantages to parallelization and the benefits increase with the model size and number of scenarios, since all scenarios could be run in parallel.

For the four cases in Table 6, NEG analyzed generation of power plants, locational marginal prices and total production costs between these simulations. There was a substantial difference in all

metrics when going from no overlap resulting in inaccurate resetting of boundary conditions to using a 24-hour overlap; however, these differences become negligible in comparison to each other and to the sequentially executed simulation with increasing each overlap by additional 24 hours. PSO has additional features that impact the detail in which boundary conditions are evaluated and NEG will continue to work with Polaris to refine parallelization of scenarios for various models.

As a results of Phase I development efforts, NEG understands the economic and model parameters that need to be considered for effective parallelization. For the purpose of a proof of concept, NEG implemented a heuristic that tests communication between the PSO model, Azure Worker Roles and the end user ensuring that any heuristic relying on these components can be implemented. In addition, NEG established that compared to running a simulation as a single continuous time window, appropriate boundary conditions produce reliable results at significant time savings.

8.2.2 Segment Processing

A segment is processed by a Worker Role. This may be an already existing Worker Role that was idle for some time after it has finished processing some earlier submitted project/scenario or a new Worker Role created to for this segment.

When a machine is first created, the AMS deploys the *pCloud* Worker (pWorker) application to this machine. NEG created pWorker, a specialized console application, to facilitate communication between the Worker Role, the AMS and the pManager front end as well as to monitor the performance of PSO and the Worker Role itself. To speed up the creation of new Worker Roles, NEG keeps a single Worker Role running with the pWorker application at all times, making it easier for the AMS to clone the needed environment for additional Worker Roles.

pWorker deployed on an idle Worker Role checks the Azure Running Instances Table (more about this below) and the AMQ at regular intervals for new simulation segments. Each message in the AMQ contains information about the location of input files, the time interval to be simulated and a security token reflecting customer information. When pWorker reads a message from the AMQ, pManager makes this message hidden so that no other Worker Role can read it and writes the contents of the message to the Azure Running Instances Table. pManager uses this table to identify what each Worker Role is working on and the status of the machine. A specific Worker Role is successfully assigned to the segment when the message is written to this table. If there was an error during the write process (generally a result of temporary network communication problems), the message is un-hidden and another Worker Role can read this message from the AMQ. Once a Worker Role is successfully assigned, pManager deletes the message from the AMQ. It is important to note that pManager does not assign specific Worker Roles to model segments. It only creates the appropriate number of preconfigured Worker Roles and each Worker Role assigns a segment to itself.

Once pWorker reads a message from the AMQ, it loads appropriate input files from Azure Blob storage to the local drive of the machine. Each scenario in the *pCloud* design corresponds to a data collection and each collection is associated with a set of Azure Blobs. Once the input files are

copied locally, pWorker makes necessary changes to the input files (reflecting the start and end of the modeling window, for example) and starts PSO execution.

8.2.3 Worker Role Management

8.2.3.1 Architecture and Implementation

NEG is designing *pCloud* to be a highly scalable environment and as such, it should manage hundreds of scenarios running on thousands of Worker Roles simultaneously. While an individual Worker Role has a very low failure rate, any single Worker Role in a large distributed environment is assumed to fail on a more frequent basis, both permanently and intermittently. The *pCloud* system must be able to withstand the complete failure of several machines, possibly many happening at the same time. While performance may degrade proportional to the number of machines lost, the system as a whole should not become overly slow, nor should information be lost. For Phase I, NEG created a semi-manual process for managing Worker Roles. As NEG gains more experience with utilizing a large number of Worker Roles, the current management and monitoring process will evolve into a fully automated one.

When pWorker is deployed to a Worker Role, it creates a separate thread on that Worker Role and uses it to monitor performance metrics such as memory and processor usage. These metrics are collected at regular intervals and written to an NEG created Azure Instance Status Table that stores such data for all Worker Roles. The user can view this information for any Worker Role through a pManager interface. If the user thinks that a Worker Role is not responding (e.g., they observe that the processor is not utilized for a long period of time), they can click a button through the user interface to tell pManager that the machine has failed. pManager will then update the appropriate entry in the Azure Running Instance Table (see Segment Processing section above on how messages assigning segments to Worker Roles are written to this table) to indicate that the message associated with the failed Worker Role should be read by the next available Worker Role. Thus, when a Worker Role is not running PSO, pWorker first checks the Azure Running Instance Table to see if any messages need to be read and if no such messages are found, pWorker checks the AMQ for any new segments (as described in the Segment Processing section).

8.2.3.2 Business Rules for Managing Worker Roles

In addition to the reliability of the *pCloud* system, managing Worker Roles raises economic considerations and as mentioned in the Meeting Phase I Objectives section, these considerations are closely related to the way in which the modeling time window is partitioned. In addition to heuristic for spawning and killing machines, *pCloud* needs the ability for users to manage their computation costs directly. To that end, NEG implemented functionality to track total Worker Role costs for each user in real-time. NEG is able to provide this information directly to the user who will be able to use it to manage the scalability of Worker Roles running their scenarios. The coding required to implement such management policies will not be challenging but it is important to ensure that *pCloud* is able to support this type of interaction between users and Azure, which has been established during phase I.

The size of deployed Worker Roles can also be selected programmatically and can be incorporated into a cost/performance decision. For phase I, NEG tested small and large machines. The size and pricing for these machines is displayed below:

Small Machine CPU	1.6 GHz, 1.75 GB RAM, 225 GB storage	$0.12/Hr
Large Machine CPU	4x1.6 GHz, 7 GB RAM, 1000 GB storage	$0.48/Hr

NEG first ensured that Worker Role size can be selected programmatically and that users would be able to choose which type of machines they want to use. Next, NEG ran the same test case, once on four small machines and four times simultaneously on one large machine. There was virtually no difference in total run time and since the large machine is exactly four times more expensive, there is no difference in terms of a cost/performance tradeoff. Despite this, NEG is currently planning to use only small machines. As discussed in the architecture portion of this section, when thousands of Worker Roles are deployed, the chance that any one of them fails can be significant. If a large Worker Role running four segments fails, all four must be restarted, which would result in longer delays than restarting a single segment when only small Worker Roles are used. There is, of course, additional overhead to managing four times as many machines but both Azure and *pCloud* are designed with scalability in mind and since *pCloud* utilization is currently low, having more resources to manage helps to better test the system.

8.2.4 Simulator Diagnostics

During execution, PSO writes out log files with specific types of status, warning and error messages. As specified in the system design, the Simulation Service is responsible for providing various diagnostics on model segment execution. The *pCloud* system provides users with indicators that certain elements may have malfunctioned; however, in the current implementation, the user must manually confirm that an error occurred. As data is collected on the type, frequency and severity of various errors, NEG will develop rules that can be built into the *pCloud* system so that appropriate diagnostics can be handled without user intervention. Diagnostics related to Worker Roles are explained in the Worker Role Management section above. For model related errors, pWorker parses PSO logs as messages are written and extracts information regarding solution progress and any potential warning/error messages. All relevant PSO messages are written to an Azure PSO Status Table and can be viewed by the client through one of the pManager interfaces. PSO errors are generally fatal and while execution may continue, the user will need to resolve the error and re-run. Therefore, if pWorker encounters an error in the log, it currently shuts down the machine immediately after writing the encountered error to the Azure table. In the current implementation, all other segments associated with the scenario will continue running but there is nothing to prevent the implementation of a rule that would stop all scenario segments for any or some encountered errors. To interact with the simulator in this way, NEG implemented a standard file based communication protocol between pWorker and the simulator (PSO in this case). NEG expects that this approach will work for other simulators as well.

NEG also created a WCF endpoint on each Worker Role, allowing pManager to communicate directly with the pWorker application on any machine. The WCF endpoint allows pManager to

shutdown/restart the machine and to pass PSO input data and run instruction directly to a specific Worker Role in the event that a segment failed for a non-Worker Role related issue. If pManager communicates with a specific Worker Role in this fashion we say that the Worker Role is in Admin Mode. If a Worker Roles is placed in Admin Mode it does not check the AMQ or the Azure Running Instance Table but rather works directly with pManager to re-run. This approach circumvents the need to create a new scenario if the user only needs to re-run a particular segment.

pWorker also tracks total PSO run time and records it in the Running Instances Table. pManager can then read this table and monitor running time by segment or Worker Role. This information is needed for accounting purposes but also provides useful information for detecting simulator related errors (e.g., a particular segment is running significantly longer than expected)

8.2.5 Segment Merging

When a PSO segment finishes executing, PSO automatically writes a set of CSV output files to a pre-specified directory. pWorker reads these files, performs some processing and uploads them to the Output Blob for that segment's scenario. In the current implementation, the user tells pManager when a scenario is complete. pManager then goes through the Blob, merges the segment CSV files into a single set of files, zips this set, saves it back to the Blob and deletes the individual files.

Once the merging process is complete the user is able to download the zipped files to their local machines and perform any needed analysis. In the future, it will be more efficient for pWorker to perform the merging and zipping of files. Similarly to the Admin mode described above, NEG will add another mode dedicated to certain post-processing tasks. In addition to using Blobs, NEG experimented with writing an aggregation of some results to Azure Tables and to an Azure SQL Database. The use of Azure Tables and Azure SQL is geared toward report generation, which was not part of Phase I objectives. However, both will be used going forward and after testing their functionality, NEG is in a better position to incorporate them into the next version of the system architecture.

8.2.6 Data Management and Reporting

8.2.6.1 Architecture and Implementation

The Data Management Service described in the system design is meant to provide users a set of tools to manage and work with all data related to their simulations. For phase I, NEG implemented a portion of this service that allows users to generate PSO input files and to store the results generated by PSO within Azure.

The Reporting Service is responsible for providing users a set of standard reports as well as flexible pre-aggregated datasets that can be used to create custom reports in tools such as Excel or Access. The Reporting Service will allow users to save these reports within *pCloud* or to download them locally. For Phase I, NEG tested some of this functionality in a standalone setting and will continue developing the Reporting Service after the Phase I period.

Figure 4 above obfuscates some of the data management components that are involved in generating input files and processing simulator results. Figure 5 presents the architecture implemented in Phase I for handling input and output data

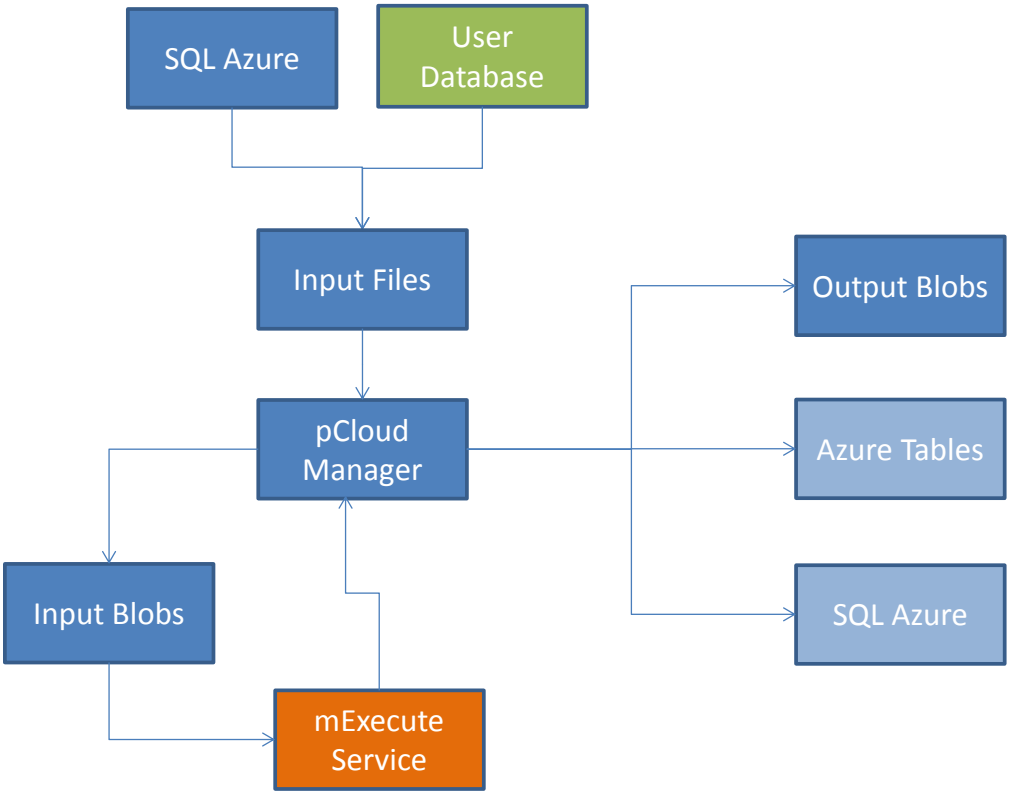

Figure 5. Data Management Components

The mExecute Service Block is an aggregation of components from Figure 4 that are responsible for scheduling and executing model segments.

In parallel to Phase I efforts, NEG developed a SQL database for storing input data for power market simulators. Through a set of SQL stored procedures, NEG generates input files for a PSO scenario and through one of the pManager interfaces these files are uploaded into appropriate Azure Blobs and passed to the mExecute Service. NEG is able to successfully port this database to an equivalent SQL Azure Database and in the future, pManger will have the capability to generate input files directly from such a database.

Figure 4 encompasses components from a few services. The merging of individual segment result files into scenario result files is included in the Simulation Service; creation of input files and management of input data via a SQL database will feed into the Data Management Service while the ability to download results and storage of aggregated results in Azure Tables and SQL will be part of the Reporting Service.

8.2.6.2 Other Considerations

At the start of this project, NEG considered various data storage implementations. While some decisions regarding storage of output files have been left for future work, NEG analyzed the use of Azure SQL as well as number of 3rd party applications during the course of this project.

As described in the system architecture, Azure SQL is currently being used to store input data and input files are physically generated before being read in by pManager. This will be a useful option for customers who have their own tools for generating simulator specific input files. Such users will have the capability to upload their files directly to Blob storage and avoid Azure SQL cost. On the output side, Blobs and Azure Tables are current being used for storage. All scenario data is archived as Blobs and can be stored on the cloud or downloaded by the user. A subset of results was also written to Azure Tables and Azure SQL to test reporting capabilities. While Azure tables are not expensive, they are not as efficient as SQL at processing highly relational data that is typical of power markets. SQL databases on the other hand do not scale well and would be prohibitively expensive if used exclusively and continuously. For one of the test cases performed, NEG created some aggregated summaries from the original CSV results and wrote them to an Azure SQL database that further processed these summaries and generated a report. A new Azure database takes less than a minute to generate and importing even a large quantity of data (1 million rows) can be accomplished within another minute or two. As long as only summary data is uploaded to Azure SQL, costs will be reasonable. However, a few scenarios can easily generate several GB of data and with data getting accumulated, permanently storing them in the Azure SQL may become prohibitively expensive. Given the short startup time required to get a database running, NEG is able to recreate these databases on demand, which would control costs. The details of this architecture for on-demand creations of SQL databases and management of such databases to control user costs will be further developed during Phase II of this project, if awarded.

In addition to the Microsoft tools, NEG assessed the Hierarchical Data Format HDF5[3] developed by the National Center for Supercomputing Applications and Apache Hadoop technologies for storing data. HDF5 allows for efficient storage of hierarchical data with complex relationships. In some ways this format is similar to a database in the sense that it has a user defined structure and allows for querying of data based on specific rules. HDF5 is ideal for sequential processing of data but cannot directly handle queries based on field matching. NEG experimented with creating a small HDF5 file from the pManager application and storing raw output files within a single HDF5 structure would be a more sophisticated approach than using a zip file.

Apache Hadoop is a distributed file system that is built to effectively store and process large quantities of data (terabytes to petabytes). Microsoft has recently incorporated this technology into the Azure offering, however, the exact costs are still unclear and there are security issues that Apache is still addressing within the implementation. NEG will monitor Hadoop developments and consider it as a future addition to *pCloud*.

[3] See for example information on the HDF Group web site: http://www.hdfgroup.org/

8.2.7 User Service

The next two sections describe the initial efforts that will be a part of the Customer and User Services. Every customer and user must be authenticated to use any *pCloud* service. Further, once authenticated, each user within a customer organization will be authorized to access certain projects and data. While a sophisticated authorization scheme is outside of the Phase I scope, an initial architecture was implemented for both authentication and authorization

8.2.7.1 Authentication

During Phase I NEG experimented with two types of authentication:

- Form based authentication using a Microsoft ASP.NET provider. The standard Microsoft provider requires SQL Server to store all user accounts. Given the relatively large cost of maintaining an Azure SQL database, NEG developed a prototype based on Azure tables, which are essentially free.
- Token based authentication. In this approach *pCloud* does not authenticate users against any internal storage. Instead, *pCloud* accepts pre-specified users from partner providers such as Google or Live. Users login to their partner account and *pCloud* only obtains the user's token (email address, for example). *pCloud* stores the association between tokens and customers in an Azure Table and when a user navigates to *pCloud*, their token is validated against this table. In this approach *pCloud* is not responsible for authentication and trusts that partner providers perform proper authentication.

NEG expects to support both forms of Authentication and will work to implement a more scalable version after completion of Phase I.

8.2.7.2 Authorization and Certificate Management

In the Phase I implementation, once a client is authenticated, they are associated with a storage account. Using this account, a client is able to access various types of Azure storage. While this functionality was tested, all Blobs are currently created in a shared access mode to simplify the overall prototype testing process. In general, NEG expects that in some cases clients may want to setup data firewalls within their own organization. To implement this kind of protection, advanced certificate management will need to be implemented in Phase II, if awarded. This functionality will allow clients to manage storage accounts directly rather than relying on the *pCloud* administrator.

8.2.8 Simulator Access Specifications

One of the key features of *pCloud* is the access it provides to multiple simulators. As explained in the system design, different simulators will have their own input and output structures. For Phase I, NEG worked with PSO to make it cloud compliant and developed a set of specifications for

additional simulators that NEG plans to add[4]. Table 6 below summarizes these specifications and indicates whether corresponding changes were required in PSO.

Specification	Justification	PSO Modifications
Simulator should be runnable through command line or exposed dll functions	Relying on a simulator's interface is not feasible in the cloud environment, all model parameters should be communicated to the simulator programmatically	Some modifications were made to add allow all required input and output parameters to be specified programmatically
Simulator should have an option to run in DataCheck and/or Full Mode	Each simulator has unique rules on validating input data. This validation should be a standalone task that can be performed without the need to fully run a simulation on hundreds or thousands of Worker Roles.	An additional input parameter was exposed to allow *pCloud* to run PSO in DataCheck mode programmatically.
Simulator should not rely on any 3rd party components that are not provided to NEG as part of the licensing agreement	Relying on software such as Excel for input or output data would require NEG to deploy Excel Tools to each Worker Role, slowing down model execution and significantly increasing costs	PSO implemented functionality to read from and write to CSV files.
Simulator should perform status/warning/error handling by writing log files or by allowing internal simulator events to be intercepted	Since the core logic of any simulator will not be available to NEG, each simulator should write verbose messages to provide feedback to users. In a more advanced setting, *pCloud* could communicate with a simulator directly via events to gain more in-depth diagnostics or to perform corrective action in real-time.	PSO already writes verbose messages to flat files. Some additional information was added to indicate the progress of a simulation. PSO also allows for direct communication with the model solver - to be implemented in the future.

Table 6. Simulator Specifications

Based on NEG's experience with power market simulators, most of them already meet a majority of the above specifications. Most of the changes required will be similar to those performed by PSO and will not be cost or time prohibitive.

While phase I efforts focused to ensure technical feasibility, *pCloud* is designed with the goal to bring power market simulation to a broader range of consumers; NEG carefully considered tradeoffs between computational efficiency and cost when developing Business Intelligence (BI) for *pCloud* Services. The section below outlines business processes for the ongoing support of *pCloud* and describes a set of BI rules that NEG analyzed and tested during Phase I.

[4] While preparing this report, NEG has reached an agreement with PowerGEM, a vendor of another power market simulator, PROBE. Under this agreement, NEG will provide *pCloud* users with access to PROBE in parallel to providing access to PSO.

8.3 Outline of the business processes for the on-going support of *pCloud*

In order to deliver *pCloud* to its clients, NEG will set up a business process supporting each service component. The high level outline of this business process is presented schematically in 6 below. In this section, we provide a very brief discussion of each element of the business process; the implementation remains to be developed during Phase II of this project. As outlined below, the business process focuses primarily on supporting SaaS for existing customers and presently does not include sales and marketing services associated with business development and service expansion.

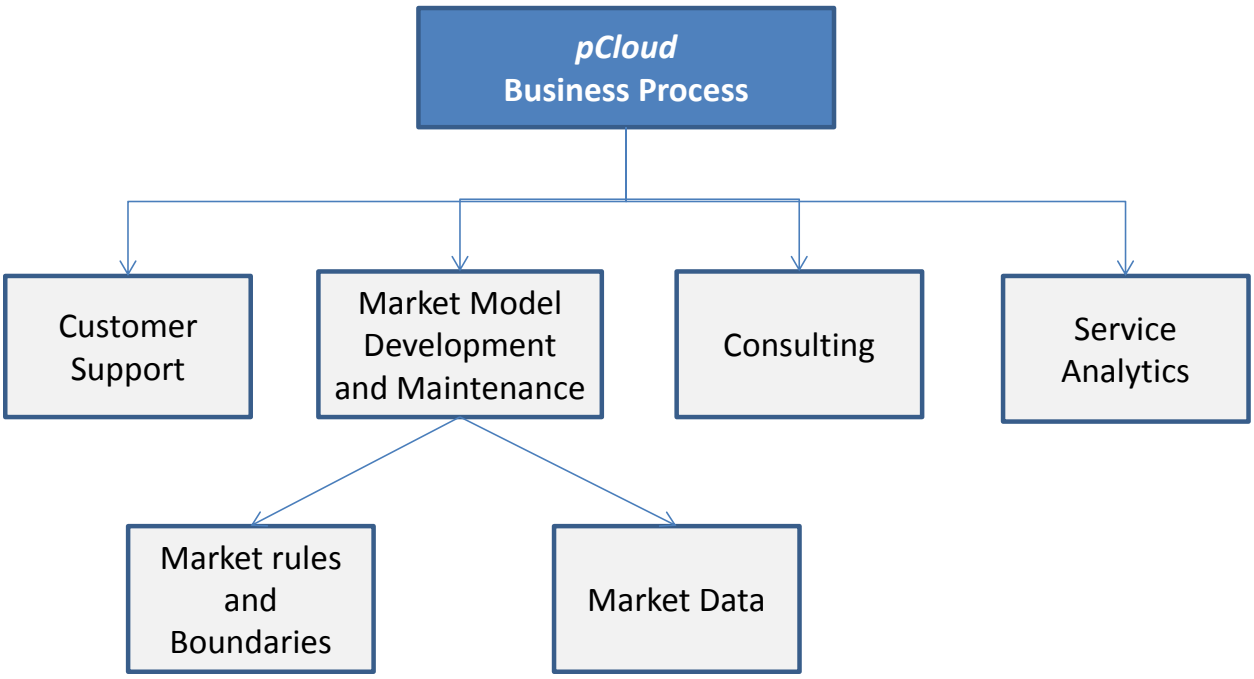

Figure 6. Outline of the Business Process supporting pCloud Service Components

8.3.1 Customer Support

NEG will be providing customer support for all commercial and technical aspects of *pCloud*. On the commercial side, this will involve responding to issues related to invoicing and billing of existing customers. This aspect is straightforward given the availability of real-time, on-line usage reporting as a part of the Customer/User Service element of *pCloud*.

On the technical side, NEG will be positioned at the front end of the customer support chain which will also involve vendors of the *pCloud* components. Most of the technical issues will be covered by detailed technical documentation for *pCloud* available on-line, contextually indexed and searchable. Technical documentation will be further supplemented by the Q&A page on the *pCloud* site, Blog and wiki-based user forum. Direct technical requests will be facilitated via telephone calls, email requests, and chat messaging. NEG will employ a standard two-tier approach in which Tier 1 specialists with basic knowledge of *pCloud* will process initial requests and determine whether pointing to the system documentation is sufficient or if 2nd Tier specialists with specific product knowledge should be engaged. When processing such requests, NEG Tier 2 specialists will diagnose

the problem and determine whether the issue is data-driven, specifically pertains to the *pCloud* system, Windows Azure, power market simulator (e.g. PSO) or other components, such as AIMMS or Gurobi solver. Depending on this analysis, the issue will be resolved internally or technical support from component vendors will be engaged.

8.3.2 Market Model Development and Maintenance

This is one of the core NEG services. As stated earlier, *pCloud* will offer users access to pre-set, ready-to-run models of regional markets. The term "model" in this context refers to the combination of all input data and model run parameters which are necessary and sufficient to perform a long-term simulation of the selected regional market under a set of plausible assumptions. As indicated in 6, this element of the business process entails three groups of activities:

8.3.2.1 Monitoring and Implementing Changes in Market Rules and Boundaries

Each regional market model will be set up to closely replicate rules specific for the market being simulated, such as the structure and geography of ancillary services, scheduling of interchanges with neighboring systems, price formation mechanisms and many others. NEG personnel will closely monitor these rules, adjust and document the model set-up accordingly.

8.3.2.2 Maintaining up-to-date datasets

The most challenging task is the collection and maintenance of up-to-date datasets required for market modeling. NEG is in the process of collecting such data from a variety of public sources. These data include detailed information on the topology and engineering parameters of the electrical grids (e.g., transmission lines, substations, transformers, phase shifters), technical and economic characteristics of generating units (technologies, capacities, heat rate curves, emission control equipment and emission rates, ramp rates, start-up and shut-down times and limitations), regional load profiles, fuel supplies and many others. These datasets are always changing with new generating and transmission facilities being built, existing generating facilities being repowered, retrofitted for emission control, mothballed, or retired and existing transmission branches being reconductored. Furthermore, these data come from multiple sources, which often provide non-matching and inconsistent information, requiring further verification. While NEG staff has many years of experience in dealing with such issues, it is important to establish and maintain a streamlined business process for collecting, maintaining, documenting and migrating updates into standardized data structures supporting *pCloud* Data Access services.

8.3.3 Consulting

Consulting is another essential area of *pCloud*-related services which can be divided into two broad categories:

- Consulting to support existing *pCloud* customers with the development of customized solutions. This may include set-up of specialized market models, implementing changes in supporting data structures to accommodate customers' private data or developing customized post-processing methods to address specific analytical problems

- Consulting to support decision-making processes for customers who are either not direct *pCloud* users or would like to rely on *pCloud*-based analysis prepared independently by a third party.

NEG staff and affiliated consultants are highly experienced in providing these types of consulting services and are presently developing key elements of the business process for provisioning of such services.

8.3.4 Service Analytics

This element of the business process presents new opportunities for the success of *pCloud* services. Presently used power market simulators are locally deployed by their users. As a result, model developers receive no information on the actual use of their tools. In contrast, with *pCloud* centrally administered as SaaS with all simulations scheduled and executed within the Azure environment, NEG will have access to information regarding simulation problem size, complexity, frequency of use of particular regional models, durations of simulation time periods, numbers of scenarios simulated, execution times and many others. While respecting the confidentiality of *pCloud* customers, NEG will be able to gather valuable service usage statistics. This statistics will be used for:

- Improving algorithms for allocation and scheduling Worker Roles
- Developing service pricing strategies that are responsive to revealed customer usage preferences
- Informing simulator and solver developers on the mathematical properties and model structures being simulated and their simulator and solver performances
- Informing users of *pCloud* on statistics regarding the types of problems being solved, general regional interests and other statistics which could provide useful business intelligence.

8.4 Obtaining CEII Certifications

Power market simulators require power flow models as an essential data input. Power flow models provide information on the topology, electrical characteristics and representative loadings of the high voltage transmission system. This information is annually filed with the Federal Energy Regulatory Commission (FERC) under FERC Form 715 by transmission owners in the United States and is widely used within the industry. FERC Form 715is considered Critical Energy Infrastructure Information (CEII) and is not available to the general public. However, this information can be obtained subject to demonstrating a legitimate business need and under the specific terms and conditions of the Non-Disclosure Agreement with FERC. NEG filed an application for getting access to this information in February of 2012 and obtained such a certification in August of 2012. In September of 2012, NEG received power flow cases for all three interconnections (Eastern, Western and ERCOT), which provide necessary modeling data for the entire U.S. and to those parts of Canada that are synchronized with the U.S. power grid.

Obtaining this certification represents a critical step in NEG's ability to develop regional market models and provide cloud-based simulation services.

8.5 Formation of key technological partnerships and negotiation of commercial agreements with partners

Another critical step in the development of *pCloud* services is the negotiation of technology partnership agreements with vendors of *pCloud* components. Licensing for power market simulators such as PSO, modeling environments such as AIMMS and optimization solvers such as Gurobi are generally provided on a per-machine or per-processor basis, which limits the number of instances of the software that can be executed simultaneously by the same customer. This licensing model is inconsistent with the business model envisioned by NEG for *pCloud* where the number of executable instances should be unlimited. In order to resolve this problem, NEG conducted a series of discussions with Polaris (a PSO vendor) and Paragon Decision Technologies (vendor of AIMMS and a reseller of Gurobi) services. Based on these discussions, NEG has been able to reach mutually beneficial agreements, which provide NEG with unlimited copy-able licenses for PSO, AIMMS and Gurobi deployable on the cloud.

NEG is in agreement with PowerGEM LLC, vendor of PROBE, a different power market simulator, to offer PROBE via *pCloud* in parallel to PSO. A partnership arrangement at terms that are similar to those negotiated with Polaris, Paragon and Gurobi are being negotiated.

NEG has also reached an agreement with SNL Financials, a data services company, which provides NEG with access to a wide variety of power, natural gas and coal industry information essential for building regional market models. NEG has negotiated a favorable licensing agreement with SNL Financials, which allows NEG to use certain data obtained from SNL in the *pCloud* service offerings.

8.6 Outreach to Potential Clients

During the course of Phase I, NEG reached out to a number of potential *pCloud* customers in order to gauge their interest in *pCloud*, to receive their feedback on the scope of services and types of analyses that best meet their business needs, and to validate our pricing assumptions. A summary of these outreach activities is presented in 8 below.

Prospective Customer	Market Segment	Issues Discussed
The Brattle Group, Cambridge, MA	Consulting	Service offerings, pricing, areas of potential collaboration
Synapse Energy Economics, Cambridge, MA	Consulting	Service offering
Energy Security Analysis Inc., Wakefield, MA	Consulting	Service offerings, pricing, areas of potential collaboration
Resero Consulting, Sacramento, CA	Consulting	Service offering
The Lantau Group, Hong Kong	Consulting	Service offerings, pricing, areas of potential collaboration
ISO New England,	ISO/RTO	Service offerings

Holyoke, MA		
System Operator of the Russian Unified Power System, Moscow, Russia	ISO/RTO	Service offerings
ArcLight Capital, Boston, MA	Private Equity Group	Service offerings, pricing
Deutsche Bank Commodity Trading, Houston, TX	Traders	Service offerings, pricing
Noble Americas, Stamford, CT	Traders	Service offerings, pricing
Edison Mission Marketing and Trading, Boston, MA	Traders	Service offerings, pricing
Altenex, Boston, MA	Energy Management Network	Services offering, areas of potential collaboration
EN+, Moscow, Russia	Independent Power Producers	Service offering
RUSAL, Moscow, Russia	Large energy buyer	Service offering
Bloomberg	Business information service	Services offering, areas of potential collaboration
Energy Forecasting Agency, Moscow, Russia	Business information service	Services offering, areas of potential collaboration

Table 8. Summary of Outreach Activities

These contacts forming a diverse group of potential customers confirmed a significant interest in services that will be offered via *pCloud*^{jTM}.

8.7 Securing Continuing Funding for *pCloud* Development

In order to ensure that development of *pCloud* technology continues between the end of Phase I and beginning of Phase II(if awarded) NEG secured $195,000 in bridge funding through several business loans. Additional sources of funding in terms of cash and in-kind financial commitments are being negotiated.

9 Conclusions

Based on the results of Phase I activities, NEG concluded that:

- The development of cloud-base power market simulation environment within the Windows Azure platform is technologically feasible
- The development and commercialization of *pCloud*™ could be accomplished within the timeframe and budget available through the Phase II SBIR award with a relatively modest level of external funding
- *pCloud*™ has a potential of becoming a game-changing approach to the modeling and analyses of electricity markets due to the following critical advantages of this technology compared to competing offerings presently prevailing in the market:
    - Standardized access to advanced and proven power market simulators offered by third parties
    - Automatic parallelization of simulation tasks and on-demand provisioning of computing resources on Azure cloud. This combination of automation and scalability dramatically reduces turn-around time while offering the capability to increase the number of analyzed scenarios by a factor of 10, 100 or even 1000
    - Access to ready-to-use data that is continually updated and verified by NEG market data experts
    - Access to cloud-based resources leading to a reduction in hardware, software and supporting IT personnel costs
    - Competitive pricing, making high-level usage of simulation services affordable, something which is not economically feasible with existing technologies
    - Availability and affordability of high quality power simulators, which presently only relatively large corporate clients can afford, will level the playing field in developing regional energy policies, determining prudent cost recovery mechanisms and assuring just and reasonable rates to consumers
    - Users that presently do not have the resources to internally maintain modeling capabilities will now be able to run simulations. This will invite more players into the industry, ultimately leading to more transparent and liquid markets