# Green Data Center Computing: A Demonstration Project
# NYSERDA Agreement # 22899
# Period of This Report: Oct 16, 2012 – Jan 25, 2013

## Introduction

The objective of this project is to demonstrate the feasibility of deploying a network of Performance Optimized Datacenters (PODs), geographically distributed to exploit the availability of renewable energy for their operation. A distributed system with PODs co-located with the power source has the potential to significantly enhance the energy efficiency, reliability, security, and overall performance of data centers. The system is capable of optimizing renewable power use for computing, by intelligently redistributing computational load depending on the availability of renewable energy, thereby minimizing the losses associated with power transmission. As the availability of wind power through New York State and other states grow, this concept would address transmission & distribution (T&D) constraints without expensive utility upgrades. By keeping the infrastructure and T&D costs low and by utilizing the wind power that is currently stranded (i.e. not-delivered to the grid due to the T&D constraints), implementation of this technology is expected yield significant energy and cost savings.

## Short Summary

This report presents details of the project work performed to date:

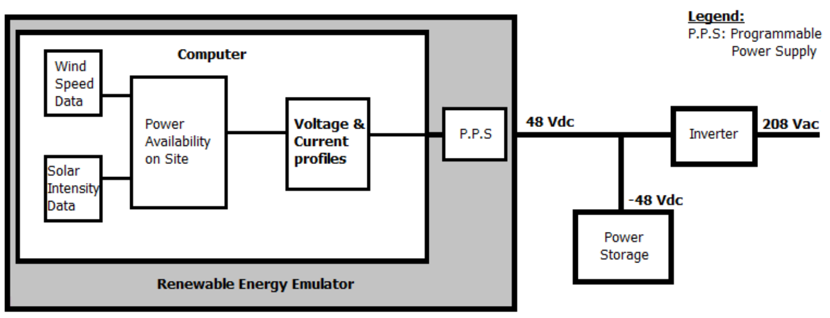
- **Task 1: System Design and Integration.** This task includes POD setup, Operation Management Unit (OMU) design and assembly, and Energy Management Unit (EMU) and Power Supply Unit (PSU) development. This task is currently in progress. Significant progress was made during the last quarter and includes the completion of the design of the full system and acquisition and installation of several components for the in-house laboratory. In addition, the renewable energy emulator architecture was improved to be consistent with the changes and updates made in the complete POD architecture.
- **Task 2: Characterization of the subsystems and the assembly to determine the correlation between Input Power (to POD) to Wind Variability.** In last report, this task is currently under investigation. An algorithm implemented in a numerical environment, was developed to model the power commitment scheduling. In contrast, this report presents the results of an algorithm assessing the power availability based on hourly wind speed in order to predict the battery state of charge's evolution. In order to do that, five scenarios were investigated where the battery was initially at a certain state of charge.
- **Task 3: Development of a system level model using experimental data for design optimization**, Work on this task is in progress. The initial experimental data from system components has been instrumental to this task and data analysis has commenced. Task 3 progress will be fully discussed in the next quarter report.
- **Task 4: Migration of Datacenter Workloads into and out of a POD in a controlled environment.** This task aims to characterize the type of workloads suitable for the POD architecture. Several existing virtual machine migration technologies and other tools have been investigated. Strategies for managing downtime at one POD datacenter and/or shifting work between PODs to take advantage of available renewable power have also been designed and initial experiments have been performed. A number of other tools and strategies for making POD hosting a viable alternative for a larger range of workloads have also been tested. Some workloads are a natural fit (e.g. hosting of static content) for POD datacenter operation, while others are not (e.g. write-intensive workloads with high availability and coherency requirements).

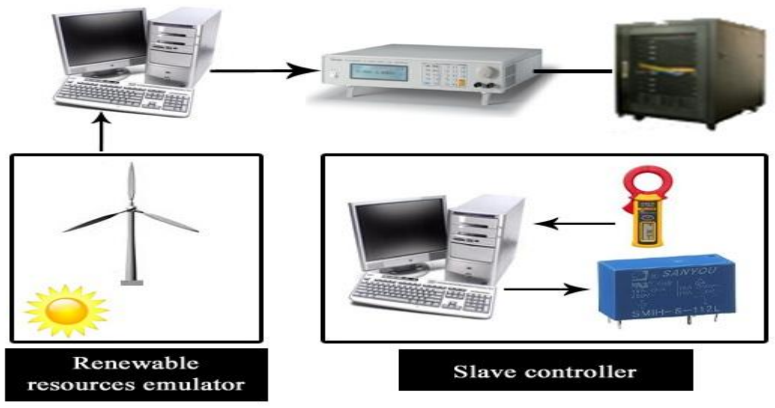
## a. Progress of Project to-Date

### TASK 1 - Progress to Date

### I. Emulator Block

**Development of the emulator:** As discussed in last report, this tool generates a power profile (current, voltage, and fluctuations) typical of a specified wind turbine generator from a given wind profile. Using real wind speed measurements, power profiles from a real generator, inclusive of aerodynamic, mechanical, and electrical losses, can be emulated. The original developed simulator also included the charge control, e.g. battery State of Charge (SOC), downwind control, and upwind control.


Legend:
P.P.S: Programmable Power Supply
Computer
Wind Speed Data
Solar Intensity Data
Power Availability on Site
Voltage & Current profiles
P.P.S
48 Vdc
Inverter
208 Vac
-48 Vdc
Power Storage
Renewable Energy Emulator



Figures 1 and 2 - Renewable Resources Emulator schematics

### Emulator update:

From a hardware standpoint it might be unsafe for the simulator to be connected directly to the battery system without any control, as hardware reacts to electric signals instantaneously while software might have latency, creating some unpredictable situations.

The proposed solution is the relaxation of the emulation algorithm by replacing the simulated charge controller code by a real charge controller device that will control the charging rate of the battery.

On the other hand, this device will protect the emulator from the back-feed current as shown in the figure below.

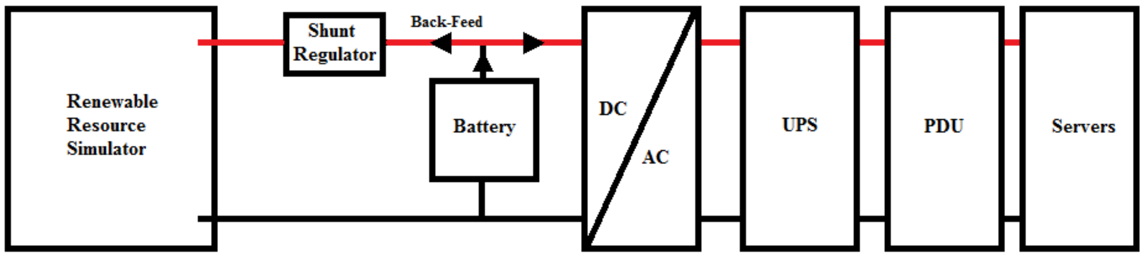


Figure 2 – Experimental setup schematic

When the available power is exceeding the load demand, the power flow will propagate from the simulator toward the servers and the exceeding power will be stored in the battery. On the other hand, if the available power from the renewable resources is less than the load demand, the battery will supply the servers. In that case, the shunt regulator introduced in the system is protecting the simulator from the back-feed current.

## Experiment setup:

The renewable emulator performance has been enhanced to meet the demand from the electricity production standpoint and to avoid electric hazard that could damage the emulator. To validate the upgraded system a laboratory setup has been assembled and tested.

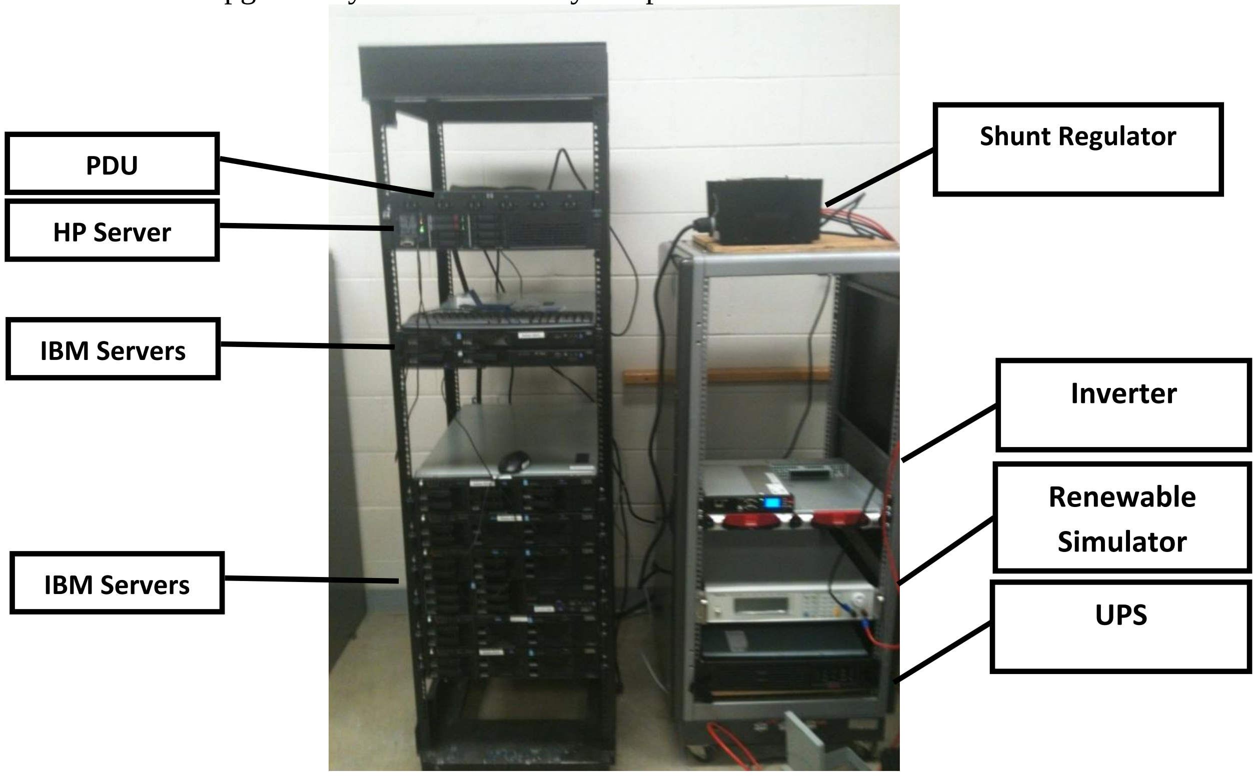


Figure 3 – Experimental setup (battery storage are not displayed)

## TASK 2 - Progress to Date

As a part of Task 2 (Characterization of the Subsystems and the Assembly to Determine the Correlation between Input Power (to POD) and Wind Variability) in the third quarter report an algorithm was presented to meet the demand and schedule the power commitment. The program provided a schedule of running several electricity units in order to meet a pre-deterministic workload. Based on available power in each location and associated price, the code generates a schedule and a power distribution that can meet the desired load demand. Significant improvements were made during this quarter. A strategy to make smart decisions based on stored power instead of following the intermittency of renewable resource such as wind power was proposed and proper algorithms have been developed.

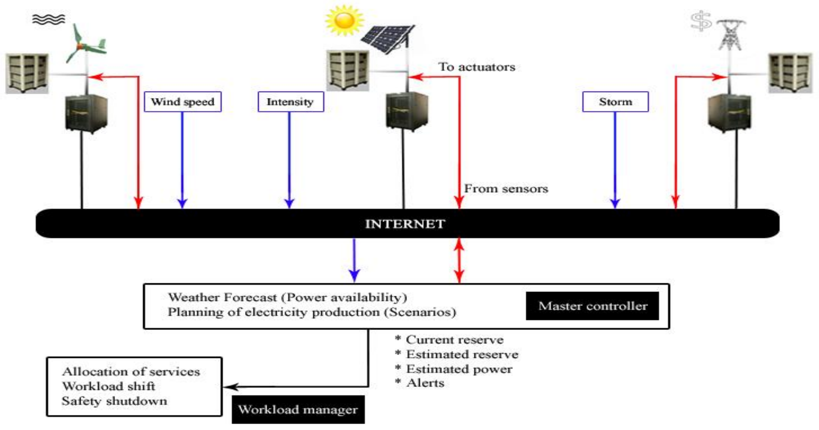
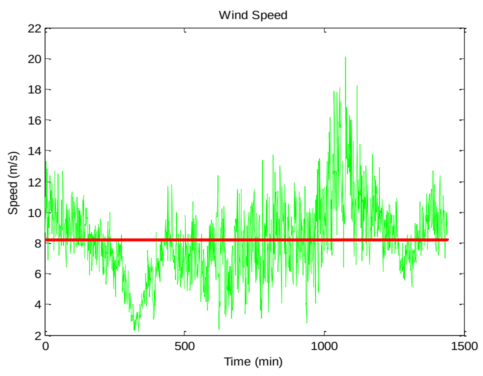


**Figure 4 – GDC Architecture**

### I. Wind power availability:

This part of the report is dedicated to describing the use of average wind speed from real field measurements. As both wind speed and power are unpredictable due to large fluctuations in wind characteristics during a day cycle, at least for small wind turbines, the use of a deterministic equation describing relationship between wind speed and wind power is virtually useless. Examples of real wind speed and power are reported in Figures 6 and 7, respectively. Instead of working with rapidly changing wind and power profiles, the selected algorithm implemented in the OMU and EMU are based on average wind speed.



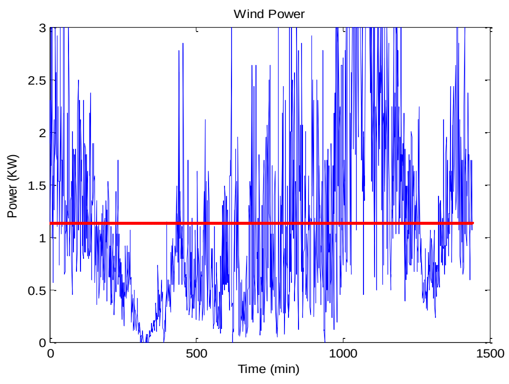


**Figure 5 - Wind speed profile (Clarkson site)** **Figure 6 – Estimated wind power profile (based on wind profile)**

## Wind speed average vs. wind power average:

Average values of wind speed and power can be hourly predicted. Quite often, forecast prediction tools provide hourly wind speed, while average wind power is estimated.

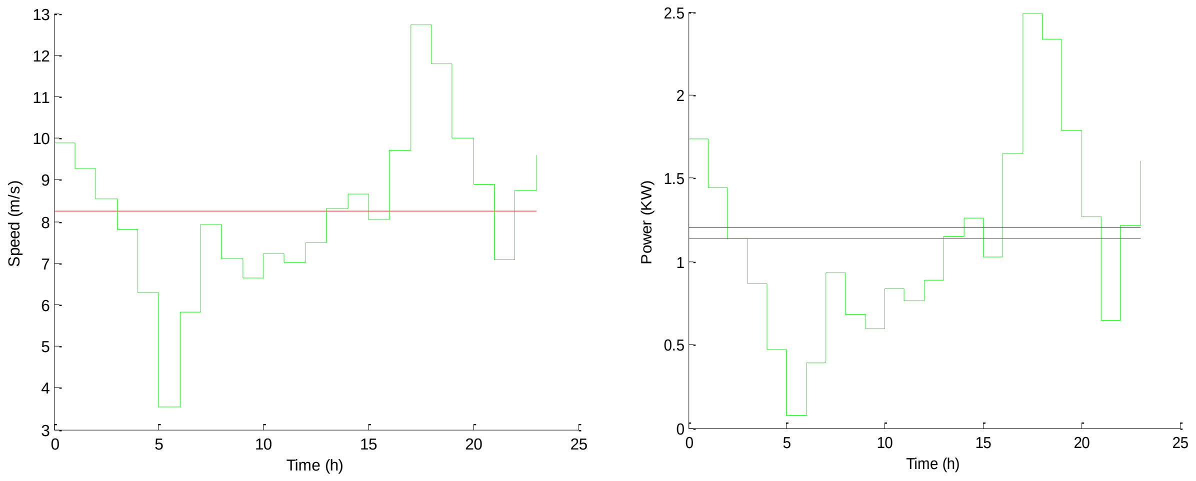


**Figure 8 – Real wind speed and estimated average power profile**

The figures above present real wind speed measurement and hourly averages along with estimated power availability and corresponding hourly averages. The hourly average (green line) and the daily average (red line) are presented along with the daily load demand average (blue line).

## II. Emulation planning:

In the emulator, wind availability is estimated using wind speed average, and an algorithm simulates the evolution of the battery for five initial conditions, 0%, 25%, 50%, 75%, and 100% state of charge. The simulations are used to study the evolution of the battery charge/discharge under loading conditions, to help taking decision about which site would have available power to host workloads. The studies also provide a great deal of information on how the system perform under variable conditions, we are planning on acquiring significant data to improve and optimize the algorithms that are currently under testing.

The following scenarios are illustrative of preliminary results obtained from a simulation. For June 21st, day selected for the simulations, it was found that with a battery SOC below 50%, the servers are not ready for use. In order to exploit them, it would be better to wait for at least one hour, when wind power can partially recharge the battery. If the battery SOC is above 50%, then the servers are ready for use for a period of 24 hours. In days when forecasted wind speed (i.e. forecasted power) is below a predefined threshold, the OMU might determine to use a different site.

| Scenario | Initial S.O.C | Runtime (h) |
|---|---|---|
| 1 | 0% | 23 |
| 2 | 25% | 23 |
| 3 | 50% | 24 |
| 4 | 75% | 24 |
| 5 | 100% | 24 |

**Scenario 1**: The battery is initially 0% charged.

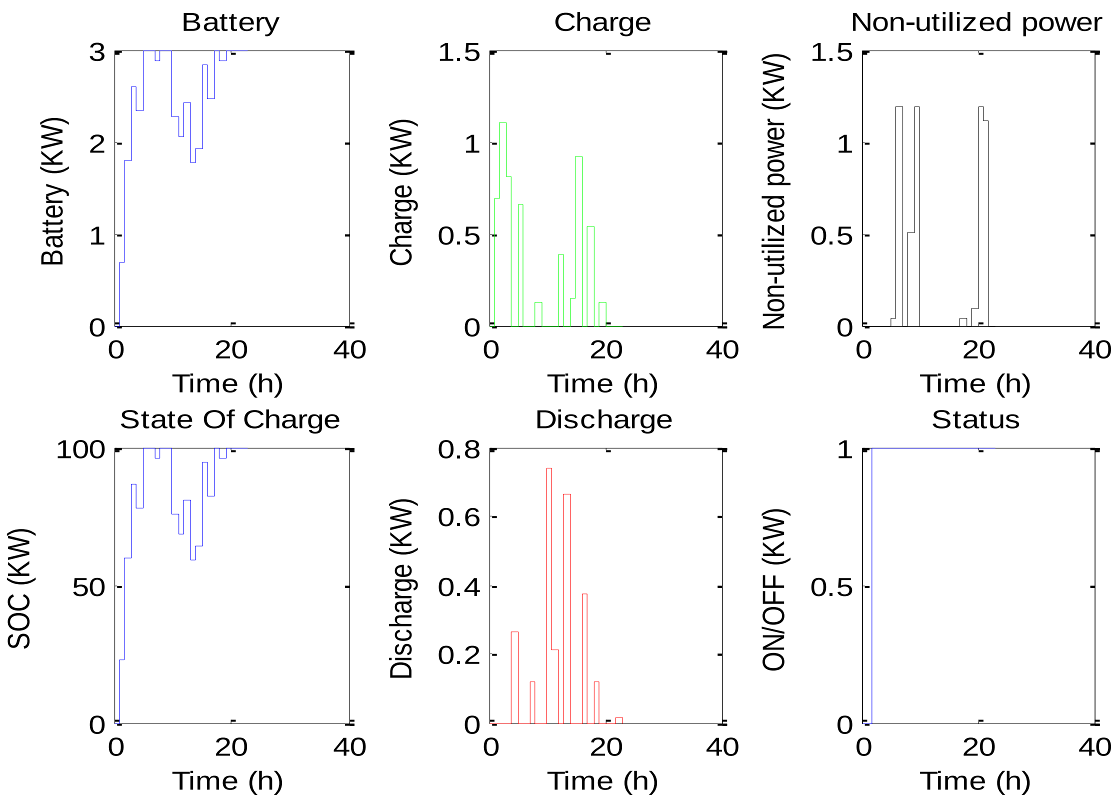


**Figure 9 – Scenario 1**

**Scenario 2:** The battery is initially 25% charged

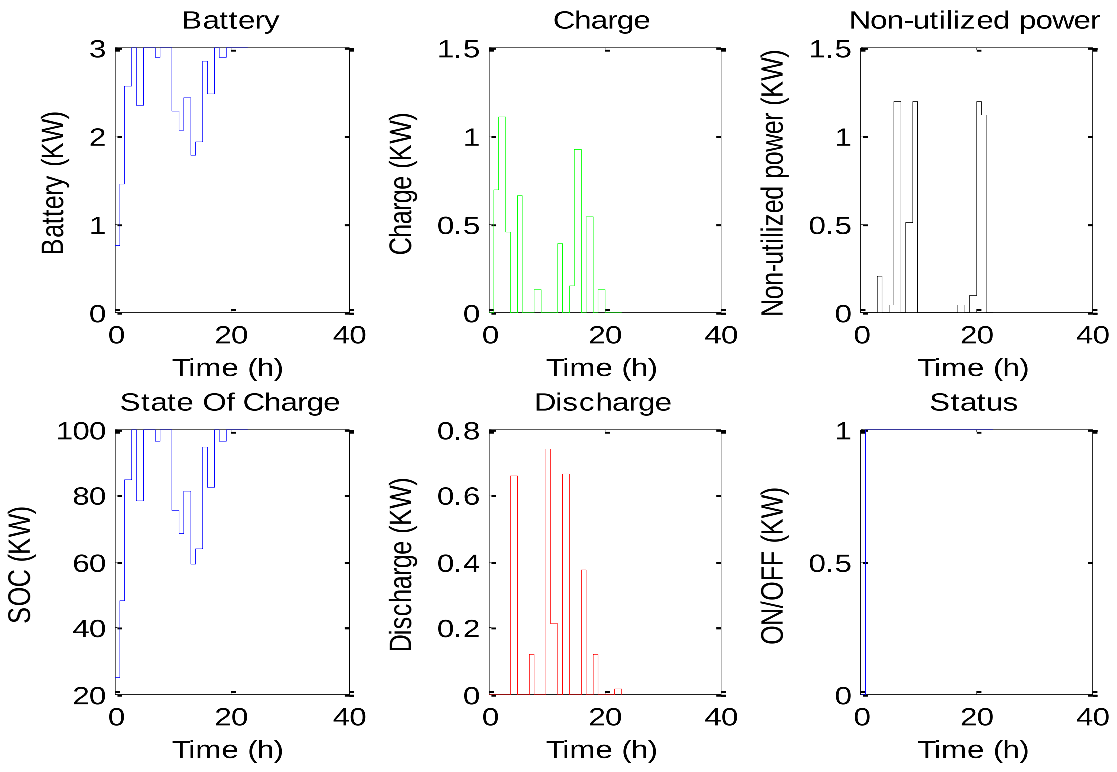


**Figure 10 – Scenario 2**

**Scenario 3:** The battery is initially 50% charged

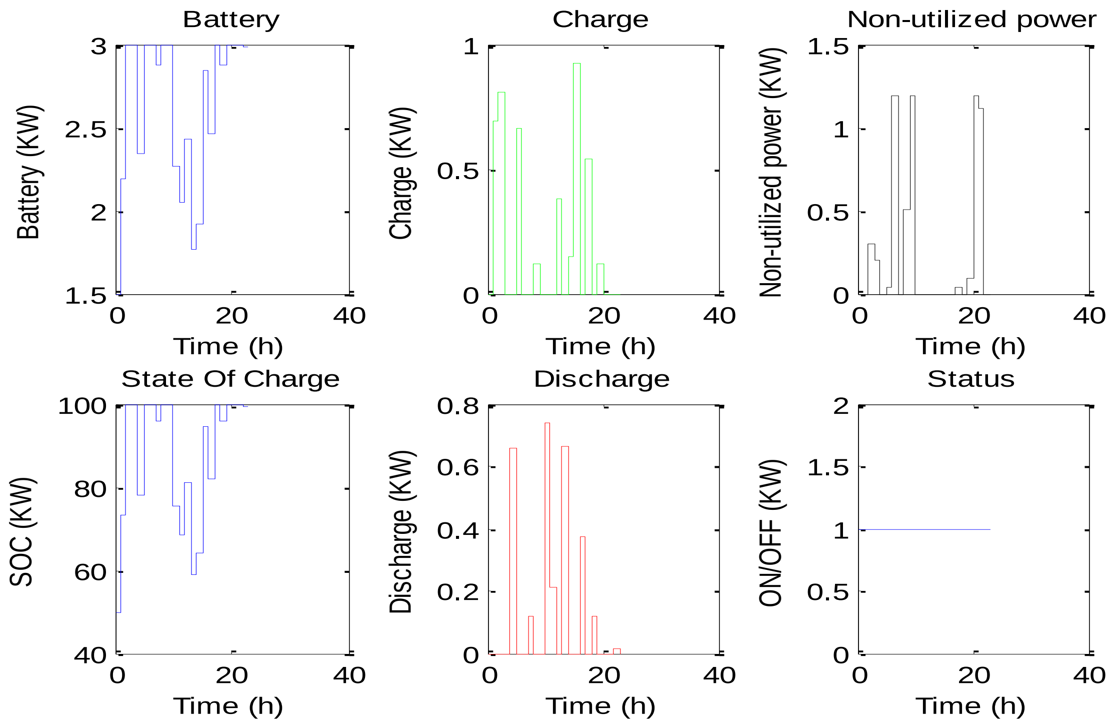


Figure 11 – Scenario 3

**Scenario 4:** The battery is initially 75% charged

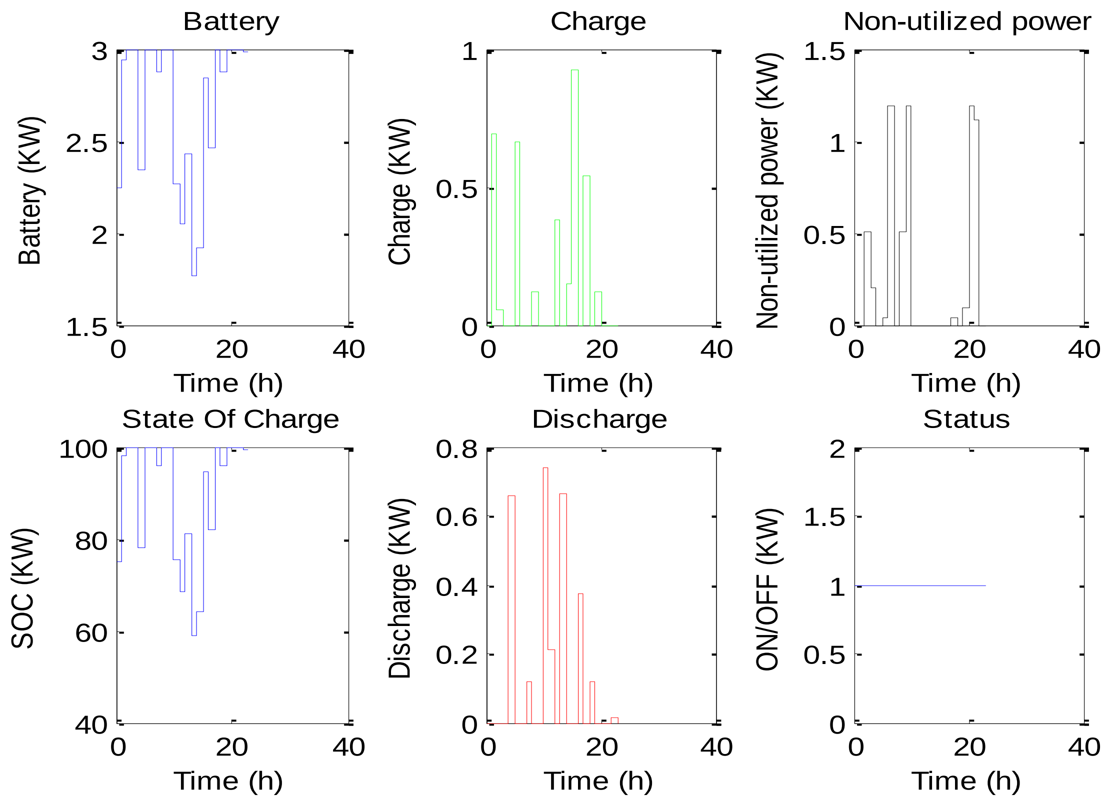


Figure 12 – Scenario 4

**Scenario 5:** The battery is initially 100% charged

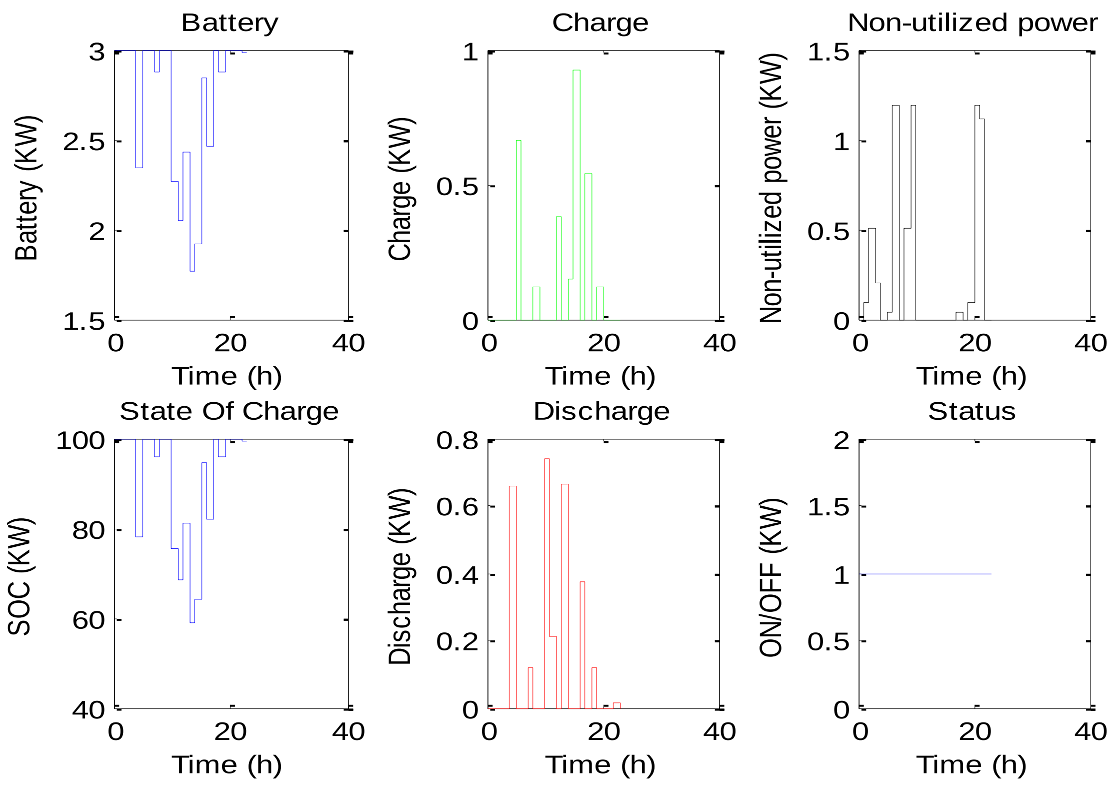


**Figure 13 – Scenario 5**

The preliminary findings and scenarios selected are used as a learning tool for our OMU development. Obviously many different simulations will be performed to potentially cover all the scenarios one would see in real operative conditions during the year; June 21st is a particularly favorable day, when wind power is available and is harvested to charge the batteries for the servers operation.

In proposed algorithm uses the one-day-ahead wind speed forecast profile to compute the next day expected power production. Then, the algorithm start with 5 initial SOC and computes the superposition of each initial SOC plus the expected power profile. The results are presented as given in the chart above. At the beginning of each new day, a reading of the SOC will be performed. For instance, a reading of 68% SOC (50% < 68% < 75%) could be conservatively considered as a 50% SOC, then based on the chart, decisions can be taken about the site reliability. In addition, each scenario will provide the evolution of the SOC (reported on the bottom left corner figure) and the servers' status (bottom right corner figure).

From the computations and forecasting information periods without available power recorded, giving a clear indication about the best period of the day to migrate workload toward a green site and when to migrate workload out of the same site. One could use the information acquired to decide of not relying at all on a particular site, instead the workloads and VMs will reside in other renewable powered or grid tied sites.

**TASK 3 - Progress to Date**

The work on Task 3 is currently in progress. The initial experimental data from system components has been instrumental to this task and data analysis has commenced. A detailed progress report will be discussed in the upcoming quarter report..

**TASK 4 - Progress to Date**

The goal of Task 4 is to characterize the workloads for which hosting in a POD datacenter would be a good match. Part of this task includes a detailed study of existing virtual machine migration technologies and an evaluation of the role virtual machine migration could play in shifting workloads from one POD datacenter to another to take advantage of available power. In addition, we are exploring a number of other tools and strategies for making POD datacenter hosting a viable alternative for a larger range of workloads. We expect that there will be some workloads for which POD datacenter hosting is a natural fit (e.g. hosting of static content) and other workloads for which POD datacenter hosting may never be a good fit (e.g. write intensive workloads with high availability and coherency requirements).

## I. Virtual Machine Migration

Our first set of subtasks includes a series of quantitative experiments of virtual machine migration. Most of the popular server class hypervisor/ virtualization technologies including KVM, VMware, XenServer, and Hyper-V offer a form of virtual machine migration. However, in many instances, it is assumed that the migration will take place between machines in the same datacenter and possibly even on the same network switch. In many cases, the machine initiating the migration must have access to the same network attached storage device (NAS) as the machine accepting the migrated VM and only the memory state is migrated from one machine to another. Some hypervisors also support live migration with storage migration where the disk state of the VM is transferred as well as the memory state.

Our first task is to perform a thorough evaluation of these migration technologies in a wide range of environments – between two machines in the same rack sharing a NAS, between two machines in the same rack not sharing a NAS, between two machines located in two different labs on campus with and without NAS and finally between an on-campus machine and an off-campus machine with and without NAS.

We are in the process of collecting a variety of interesting measurements of virtual machine migration including the total time to accomplish the migration (from start to finish), the downtime or time the virtual machine is unresponsive during migration (typically during at least the last stage of migration) and the total amount of data transferred to accomplish the migration. Eventually, we would also like to collect data on the total power required on the transmitting side to complete the migration.

We will also collect these same measurements of two extreme configurations – “cold” migration and a high-availability pair. In cold migration, the VM is suspended, the files transferred to other side and the VM resumed. This represents a worst-case scenario for the amount of time a VM will be unresponsive but makes the fewest assumptions about the environment (no NAS required for example) and we are able to manually compress all data to be transferred before the transfer begins. In a high-availability pair, VMs run constantly on both machines so full VM migration is never required. In some cases, data will be shipped from one VM to another to keep the two VMs in sync. The only thing that changes is where requests are sent (to both VMs if available, to just one, etc.).

**1. Progress to Date**

- Surveyed advertised features/requirements of several major hypervisors and cloud orchestration tools, and added this information into a master spreadsheet
- Collected the migration data for Microsoft Hyper-V and Citrix XenServer.
- Being able to initially migrate VMs between CS lab and Mechanical lab.
- Studied the migration data and proposed a system model for GDC cluster
- Studied current mainstream distributed system such as Amazon Dynamo and Google GFS and further categorized common GDC workloads

**2. Next Steps**

- Acquire Citrix XenServer License to finish XenServer Storage migration testing
- Complete the migration result comparison with cold migration.
- Complete experiments that vary the available network bandwidth
- Explore several puzzles discovered by the current results of live migration
- Complete the live migration quantitative tests and publish the results for external review

## II. Current Status (More Detailed Status)

### 1. Summary

In the previous (i.e. third quarter )of this project, the HP DL165G servers arrived and we deployed them in our testing. We optimized the testbed environment to make it less dependent on the specifics of some hypervisor systems and instrumented it more fully to measure additional important live migration parameters. More details can be retrieved from NYSERDA Third Quarter Report.

This quarter, continuing with last quarter’s Redhat KVM and VMware vSphere migration test, we extended the tests to Microsoft Hyper-V and Citrix XenServer so that we have now covered the mainstream enterprise-level virtualizations in this project. With the new data gathered this quarter, we now have data to compare live migration on all of our target virtualization systems. Based on our results, we discuss constraints on deploying virtualization migration in a GDC and propose a better architecture model for GDC. Additionally, we have studied the current existing models of distributed system management from published papers by Google, Amazon, Facebook and others. We also give an initial evaluation of the possible workload space for GDC.

### 2. Enterprise-Level Virtualizations

Deploying virtualization in the datacenter has become the prevalent trend. Datacenter administrators can easily gain the flexibility of distributing the services across servers or consolidate services on fewer servers by virtualization.

In GDC, virtualization is especially helpful in shifting the workload from lower-power GDC node to rich-power node to avoid the services of being locked down in a specific node and cause services unavailable due to node power outage.

Virtualization varies by its approaches and implementations. Current main virtualization options are summarized in the Table below along with their migration capabilities.

| | **Redhat KVM** | **VMware vSphere** | **Microsoft Hyper-V** | **Citrix XenServer** |
|---|---|---|---|---|
| **Version** | 3.2 | 5.1 | 2012 | 6.1 |
| **Platform** | Linux | Linux-based | Windows | Linux-based |
| **Approach** | Full Virtualization | Full Virtualization | Para-Virtualization | Para-Virtualization And Full Virtualizaton |
| **VM driver** | No | Yes | No | Yes |
| **Memory Migration*** | Yes | Yes | Within Failover Cluster | Within Cluster |
| **Storage Migration**** | Not officially | Yes, but only storage | Yes | Yes, but only storage |
| **FS support** | All | VMFS/NFS/ iSCSI | NTFS/Samba3.0/ iSCSI | NFS/iSCSI/FC |
| **Management Portal** | No | vCenter on Windows | No | XenCenter on Windows |

*Memory migration always requires that there be a VM image on shared storage accessible to both the source and destination machine.

**Some can migrate VM state and its image all together, some only migrate VM image.

Memory migration in all virtualization systems requires VM image residents on a shared storage so both source and destination hosts can access to the same VM image. When live migration happens, the VM's memory and CPU state must be transferred to the destination host , but theVM image need not be migrated because it is accessible to both hosts through shared storage. When this shared storage is place in one data center, this would imply high latency storage access from other data centers.

We elaborate on the virtual machine migration capabilities below using the unique terms used by each hypervisor.

KVM supports two types of live migration:

1. Memory Migration: Migrate CPU and memory but storage stays on shared datastore.
2. Storage Migration: Migrate all CPU, memory, and storage. Storage will also be migrated together with CPU and memory. Storage migration capability is recently available in KVM and not yet officially supported.

VMware supports Host Live Migration and Storage Live Migration:

1. VMware calls their memory migration feature "Host Live Migration". Host Live Migration only migrate CPU and memory; storage has to be on shared datastore.
2. Storage Live Migration: Storage Live Migration only migrate Storage while CPU and memory stays on the host. Unlike KVM, this does not migration CPU and memory..

   If VMware wants to migrate all CPU, Memory, and Storage of a VM, it has to be split into two separate phases: first host then storage or first storage then host.

Hyper-V supports Live Migration and Shared-Nothing VM Migration:

1. Hyper-V Live Migration only migrates the VM's CPU and memory to another host in the same cluster. Shared storage has to be on Samba 3.0 share or iSCSI.
2. Share-Nothing VM Migration migrates CPU, memory, and storage to any other host and VM storage has to be on iSCSI.

VM storage has to be on NTFS partition since Hyper-V is on Windows.

XenServer supports VM migration and storage XenMotion:

1. VM migration only migrates CPU and memory, storage has to be on shared datastore like NFS, iSCSI, Fibre Channel.
2. Storage XenMotion is similar to VMware Storage Live Migration,  only migrate Storage while CPU and Memory stays on the same host.

## 3. Live Migration Stress Test  Results

We are trying to study how VM activities will affect the live migration, which will help us build a clearer picture for estimating migration time when GDC is facing power shortage or outage. We find that memory activity is a bigger factor during live migration than CPU activity or read/write activity to disk.  Logically, if am VM is actively writing to memory during migration, then the newly written pages will have to be written many times until a point of diminishing returns is reached where the VM must be paused to complete the migration without allowing any additional memory to be changed.

Our memory stress test is programmed to allocate a specific memory size and then dirty a specified portion of that total allocation. We can control both the amount of memory that is written or dirtied as well as the rate at which data is dirtied.

### i. Virtualization Migration Results

The tested VM specification is 1 vCPU, 2G memory, and 40G virtual disk. VM is running Ubuntu Server 12.04 and will be migrated between two HP DL165G7s on the same gigabyte network. We have measured migration time, migration data, and migration downtime.

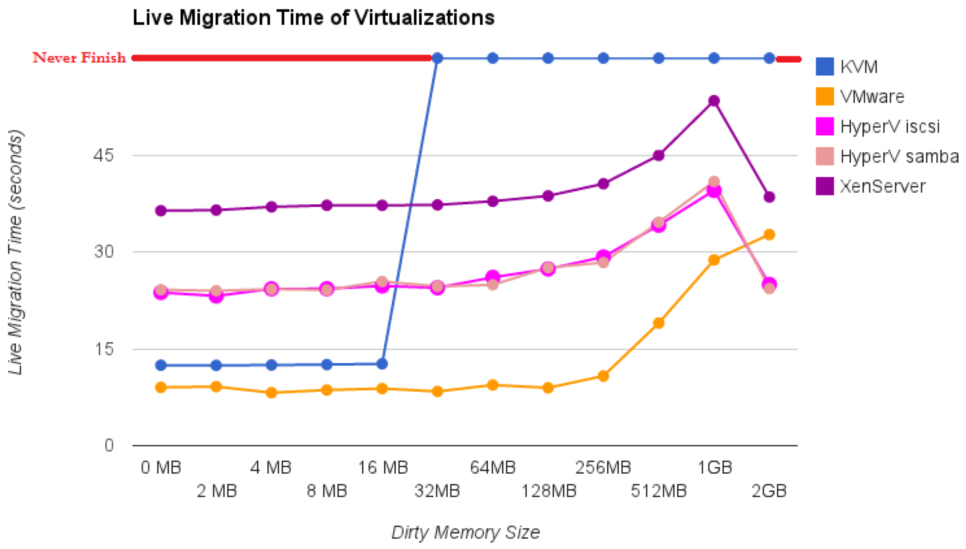


**Figure 14 – All Virtualization Memory Migration Time over Gigabyte Network**

The dirty memory size is tuned inside VM and it affects the VM migration as above.

- As dirty memory size goes up, the migration time also goes up. For VMware, the migration time is almost tripled while KVM even cannot finish the migration for 32 MB of dirtied data and above.
- The migration time varies from 15 to 45 seconds by different virtualizations. VMware has the best migration time performance while XenServer is the worse.
- Hyper-V and XenServer has the same migration pattern but Hyper-V has beats XenServer by 50%.
- For Hyper-V, which storage hosts the VM image doesn't affect the migration that much. Both iSCSI and Samba3.0 have the same performance considering the testing deviations.

Surprisingly, the migration time dropped a lot when dirty memory reaches the upper bound of VM memory size. This may due to their storage synchronization mechanism. Because when memory is over requested, Linux will put recently-unused memory onto disk. When VM is in migration, Hyper-V synchronizes the storage on both ends at the same time so most of dirty data doesn't need to be synced over the network.

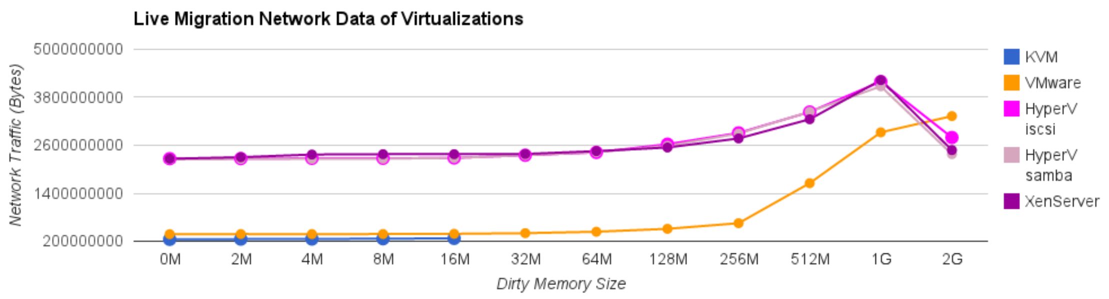


**Figure 15 – All Virtualization Memory Migration Network Transfer Data over Gigabyte Network**

The figure above well demonstrates the total amount of data transferred during VM memory migration.

- VMware and KVM migration takes the least time because they only migrate currently used memory.
- Hyper-V and XenServer migrate the whole configured memory of a VM regardless of how much memory is really in use. So the migration time will take much longer than KVM and VMware because they have more data that needs to be transferred. This is quite wasteful – why transfer the contents of a page of unused memory when the contents are just "garbage" from the perspective of the OS anyway.
- Compared from the migration time, it also indicates that XenServer has worse network/sync performance than Hyper-V because XenServer takes longer time while transferring the same data. Also the same judgment applies to VMware and KVM. VMware needs to transfer more data but takes less time than KVM.

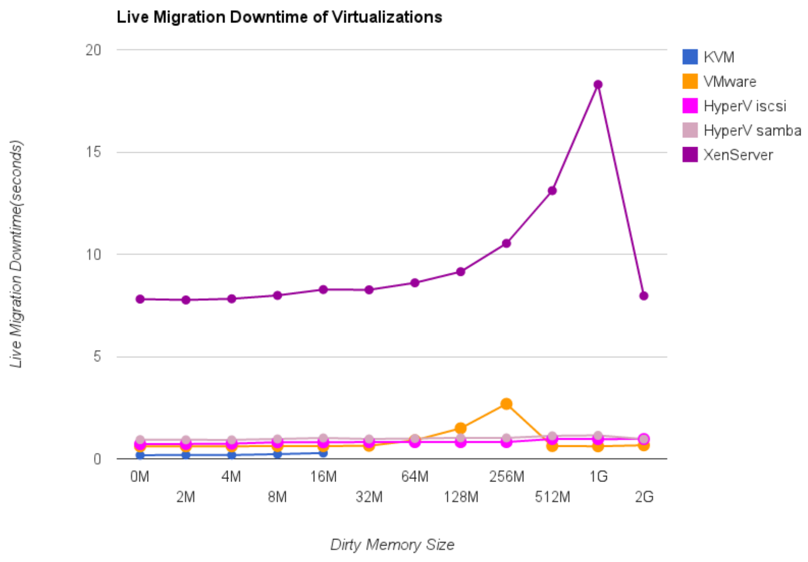


**Figure 16 – All Virtualization Memory Migration Network Transfer Data over Gigabyte Network**

We use ping test to represent the network services running inside VM. If VM failed to receive the network packets, then we assume the service is down. So during the migration, the service downtime shows as the figure above.

- KVM guarantees the least downtime to 200ms as it claims. The service may still remain connected with TCP control. Notice the tradeoff. KVM maintains a low downtime (good) but does not complete the migration for dirty memory of 32 MB or more. A hybrid approach is better – try to maintain low downtime if possible but identify when a point of diminishing returns is reached and transfer anyway evne if downtime rises. KVM does supply some manual controls for this but an adaptive, hybrid would be a substantial improvement.
- VMware faces downtime of about half a second. But as the dirty memory size goes up, the downtime faces a 2-second downtime spike, which is a repeatable mystery.
- Hyper-V generates the downtime up to 1 second, which definitely will cause service lose packets.
- The worst case is XenServer. It introduces the downtime up to 8 seconds which is similar to the time required for a full restart of the VM. That level of downtime would certainly introduce service interruption.

### ii. XenServer test issue

During our testing, we have encountered and documented a bug in Ubuntu triggered by XenServer. VM Ubuntu Server 12.04 64bit installation on Xenserver still reporting the error message

```
"Warning: Failure trying to run: chroot /target dpkg --force-depends --
install /var/cache/apt/archives/debconf_1.5.42ubuntu1_all.deb"
```

The root cause is Ubuntu-specific and lies in glibc. It may due to failure to recognize the avx CPU flags specific to AMD CPU by eglibc.

For more details, you can refer to the bug:

https://bugs.launchpad.net/ubuntu/+source/eglibc/+bug/956051 .

Ubuntu 10.04 32/64bit and Ubuntu 12.04 32bit are already fixed. But problem still exists in Ubuntu 12.04 64bit version. We use the daily-build Ubuntu 12.04 alternative 64bit to run through those tests. And the test needs to be validated once Ubuntu Server 12.04 64bit is fixed.

### iii. Memory vs Storage Migration

So far only Hyper-V officially supports Share-Nothing VM migration while KVM also supports it unofficially. Shared-Nothing VM migration means that hypervisors can live migrate a running VM instance along with its storage image all together. It has broken the barrier of live migration requiring shared storage on both hosts. Here is the result of the comparison of memory migration and share-nothing migration. Not surprisingly, storage migration takes more time than memory migration because storage additionally need to transfer VM image to the destination host.

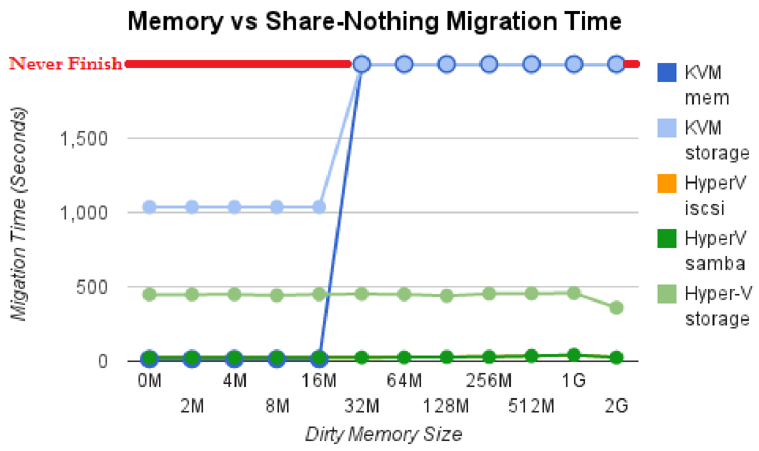


**Figure 17 – KVM vs Hyper-V memory vs storage Migration Time Comparison**

Compare the VM memory size and image size, the ratio is 2GB vs 40GB. Actually in our VM spec, virtual disk size is more conservative. In real scenario, the disk size can go up to terabytes. From the figure above, we can tell that storage migration maximize the time by the factor 500~1000. If the disk size is up to TB, then the time will be extreme huge and unaffordable to migrate by $10^6$, which may take days, weeks, or even months to transfer over gigabyte network. The only way to improve the performance is to enhance the network bandwidth. The time it takes to transfer a full VM over a small network pipe is a fundamental limitation to VM storage migration. Another viable possibility is to migration through small disk drives shipped from data center to data center. If integrated with live VM migration, it could be used to "pre-migrate" the majority of the storage and them use the network to send only the actively changed portions. We will investigate this strategy in more detail timer permitting.

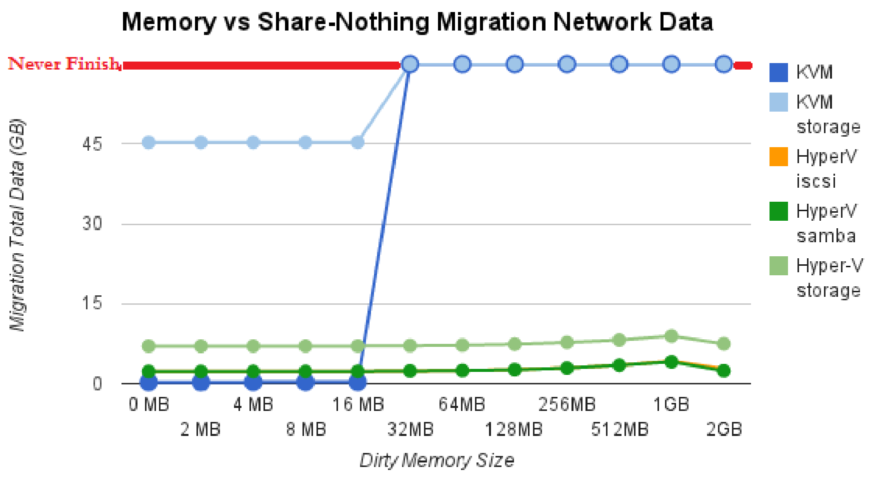


**Figure 18 – KVM vs Hyper-V memory vs storage Migration Transfer Data Comparison**

As previously mentioned, KVM cannot finish the migration after the dirty memory size goes to 32MB so the transfer data over the network may be forever.

Comparing memory and storage migration from KVM, you can find that the there is a big difference between the data size that is transferred over the gigabyte network. 45G includes KVM VM image size 40G and 2G memory size and the synced data while the memory migration data size is trivial, only including the used memory for about 200MB.

However, Hyper-V has some difference. The gap is small because Hyper-V memory migration migrates the whole configured memory size 2GB while storage migration is intelligent enough to only migrate the used disk rather than the whole disk, which is only 7GB. We saw a similar difference with memory migration and this is a key result from our testing. Just like there is no point in sending unused, garbage memory pages, there is no point in sending unused, garbage disk pages! Although only small portion of the virtual disk is transferred over the network, but it still takes a long time about 500 seconds to finish the migration.

When we manually do cold migrations of VMs between hosts, we further decrease the amount of data transferred over the network with compression. Time to compress on the source and decompress on the destination is non-trivial and adds to the total migration time. However, it can be a substantial win when transferring over a small network pipe.

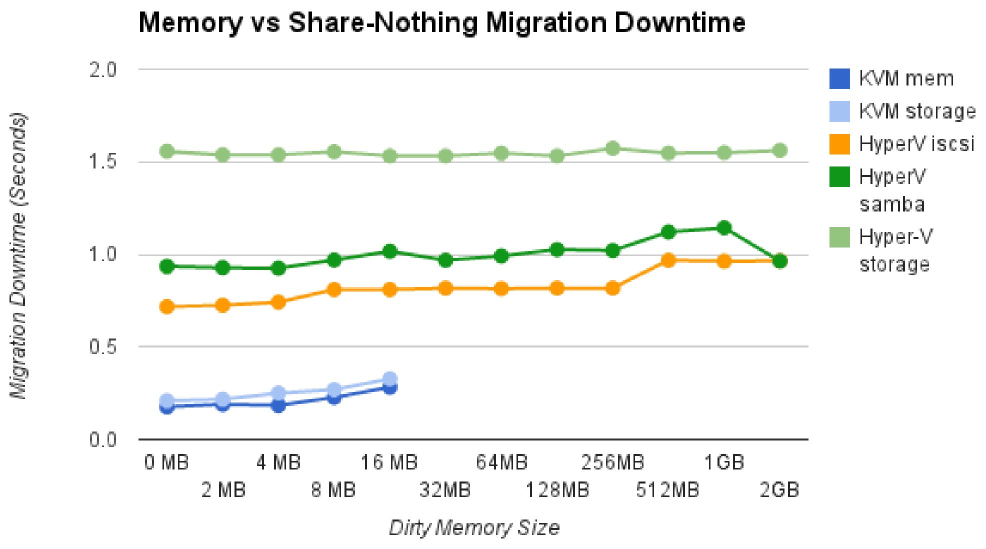


**Figure 19 – KVM vs Hyper-V memory vs storage Migration Downtime Comparison**

We expect total migration time to be much higher for storage migration than for memory migration (unsurprising as so much more data must be transferred). However, we expect the VM downtime to be similar. After all, the last phase of migration is the same, switching over the CPU state. We do see this pattern for KVM – both memory migration and storage migration show similar amounts of VM downtime. However, for Hyper-V, we see a distinctly different pattern: VM downtime is much higher for storage migration than for memory migration. One hypothesis is that Hyper-V must update its cluster wide configuration information to note the new location of the VM files. We also see some small difference in downtime between memory migration using iSCSI and memory migration using Samba 3.0 with the iSCSI migrations demonstrating lower VM downtime.

## 4. Distributed Storage System for GDC

Computation can be migrated by virtualization but large storage or bulk data is hard to move. To avoid data's inaccessibility due to GDC's power outage, there are two strategies: duplicate and synchronize. The following diagram illustrates the impact of those two strategies.

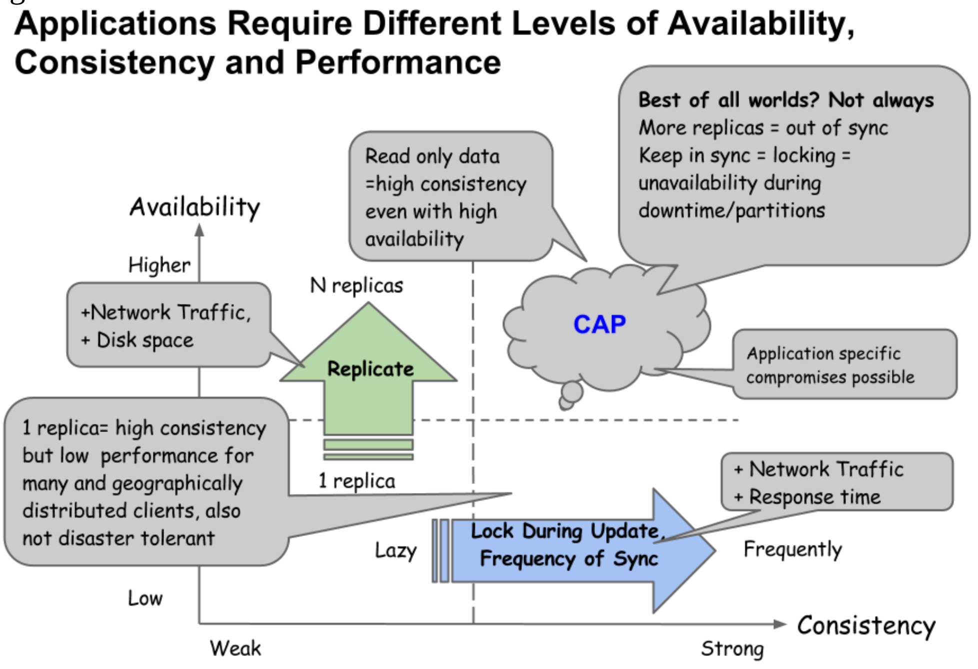

Applications Require Different Levels of Availability, Consistency and Performance
Best of all worlds? Not always
More replicas = out of sync
Keep in sync = locking = unavailability during downtime/partitions
Read only data =high consistency even with high availability
Availability
Higher
N replicas
+Network Traffic, + Disk space
Replicate
CAP
Application specific compromises possible
1 replica= high consistency but low performance for many and geographically distributed clients, also not disaster tolerant
1 replica
+ Network Traffic
+ Response time
Lock During Update
Frequency of Sync
Lazy
Frequently
Low
Consistency
Weak
Strong

**Figure 20 – Data availability, consistency, and Performance Relationship**

Making multiple copies of data across different GDC nodes can greatly improve the data's availability and response time by accessing to the nearest data. However if data is actively modified, then maintaining consistency among the copies (i.e. preventing conflicting updates or preventing some reads from seeing one copy and some reads from seeing another) can require time-consuming and complex locking procedures..

In fact, this tradeoff is fundamental and has been described in the "CAP Theorem[1]" proposed by Eric Brewer from UC Berkeley. **CAP** stands for

Strong **C**onsistency: multiple users see the same result when accessing the same data

High **A**vailability: a user can always access the data

Network **P**artition: ad-hoc network disconnections, hardware failure, or server power outage/crash

The CAP theorem states that for any distributed system containing multiple copies/replicates, it is impossible to maintain strong consistency, provide high availability, and tolerate network partitions at the same time.

In the GDC scenario, servers keep facing intermittent power outages which are one kind of network partition case, not to mention about node failure case. Theoretically it is already proved that the service in GDC have to relax its availability or consistency requirements if it tries to continuously run without interruption. This requirement relaxation is application-specific and acceptable to the end-user.

Within this space, there are excellent opportunities for GDC to host commercially important workloads. For example, read only data is a special category that may be a great fit for GDC. GDC could focus on holding cold, unchanging archival data or static content. If you consider your own data usage, you probably have a lot of data you've produced over your lifetime and most of it is unchanging day to day. You might at any time want to refer back to some of it, but you are typically changing only the more recently produced data. This model of GDC storing bulk archival data could be an excellent match for the tradeoffs required by the CAP theorem. Read only data does not change and therefore we get the consistency requirement by default.

Another important examples is workloads that are able to tolerate lower availability or lower consistency on an application specific basis. For example, some workloads may be able to tolerate a model where queries are queued and answers returned within 24 hours.

In fact GDC represents an interesting design point currently not covered by distributed systems literature. Large scale service providers such as  Google, Microsoft, Amazon and Facebook have been designed systems to tolerate frequent hardware failure and network partitions through a combination of weak consistency but high availability to achieve a user-acceptable service. However, the common failure models are within data centers – machine failure, rack failure, infrequent data center failure. The frequent data center failure model inherent in GDC is a novel design point we are excited to be exploring.

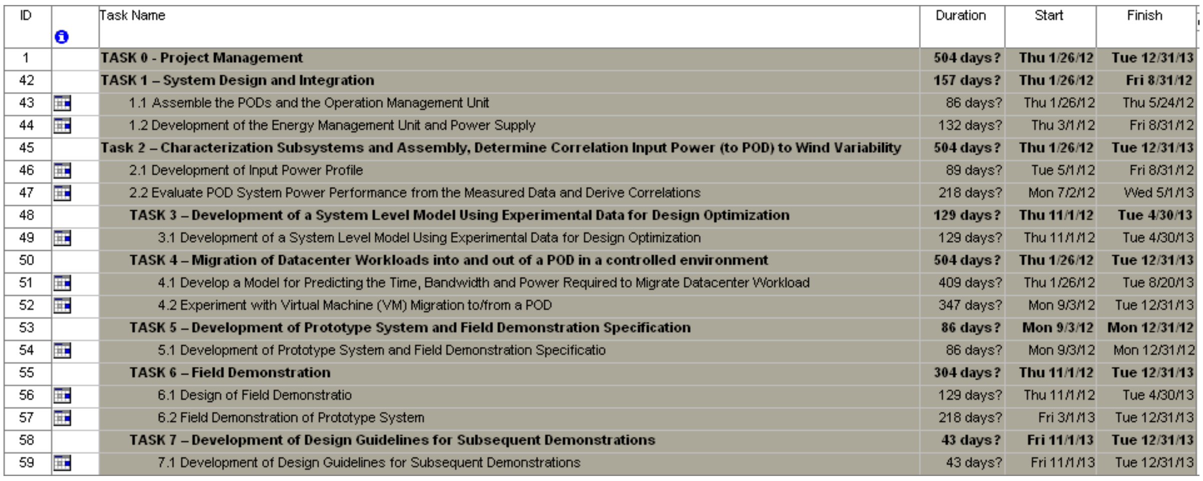
## b. Identification of Problems & Planned Solutions.

Tasks 2 through 4 are on schedule. Task 1 suffered a short delay associated with the lead time for some of the hardware components, customized for our specific application. We received all equipment including the ultra-capacitor bank, therefore we should not be encountering any further delay.

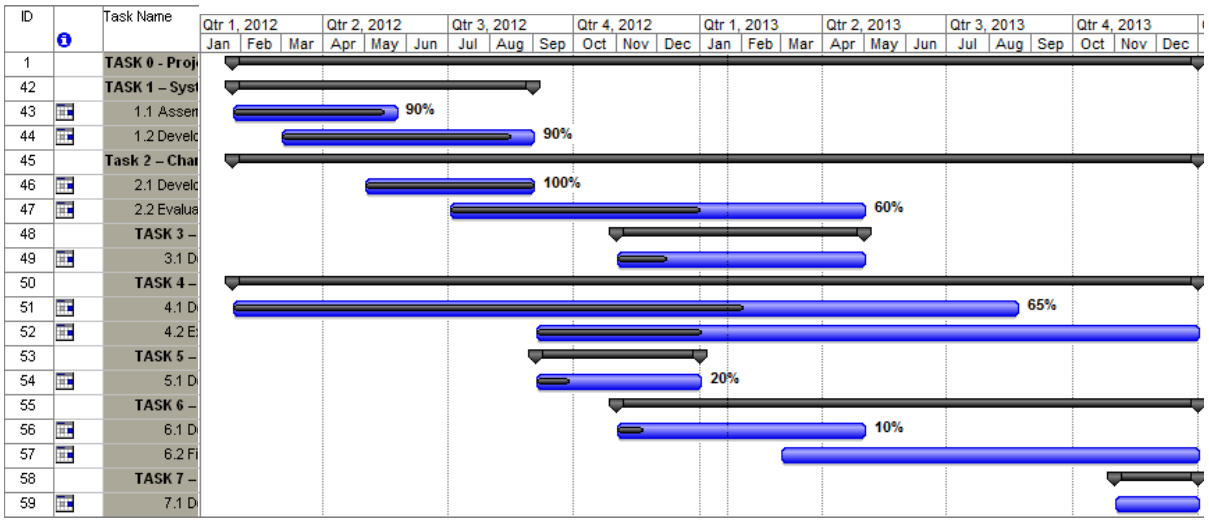
## c. Ability to meet schedule, reasons for slippage in schedule, unforeseen obstacles, promising research directions not originally identified in current SOW

No significant slippage or unforeseen obstacles encountered in the project as of now. Work is going according to the plans, although there is a short delay in completing Task 1 as indicated above. A set of HP servers with AMD processors and other equipment are currently used as testbed for our initial laboratory demonstration.

## d. Schedule - percentage completed and projected percentage of completion



| ID | ℹ | Task Name | Duration | Start | Finish |
|---|---|---|---|---|---|
| 1 | | **TASK 0 - Project Management** | **504 days?** | **Thu 1/26/12** | **Tue 12/31/13** |
| 42 | | **TASK 1 – System Design and Integration** | **157 days?** | **Thu 1/26/12** | **Fri 8/31/12** |
| 43 | ▦ | 1.1 Assemble the PODs and the Operation Management Unit | 86 days? | Thu 1/26/12 | Thu 5/24/12 |
| 44 | ▦ | 1.2 Development of the Energy Management Unit and Power Supply | 132 days? | Thu 3/1/12 | Fri 8/31/12 |
| 45 | | **Task 2 – Characterization Subsystems and Assembly, Determine Correlation Input Power (to POD) to Wind Variability** | **504 days?** | **Thu 1/26/12** | **Tue 12/31/13** |
| 46 | ▦ | 2.1 Development of Input Power Profile | 89 days? | Tue 5/1/12 | Fri 8/31/12 |
| 47 | ▦ | 2.2 Evaluate POD System Power Performance from the Measured Data and Derive Correlations | 218 days? | Mon 7/2/12 | Wed 5/1/13 |
| 48 | | **TASK 3 – Development of a System Level Model Using Experimental Data for Design Optimization** | **129 days?** | **Thu 11/1/12** | **Tue 4/30/13** |
| 49 | ▦ | 3.1 Development of a System Level Model Using Experimental Data for Design Optimization | 129 days? | Thu 11/1/12 | Tue 4/30/13 |
| 50 | | **TASK 4 – Migration of Datacenter Workloads into and out of a POD in a controlled environment** | **504 days?** | **Thu 1/26/12** | **Tue 12/31/13** |
| 51 | ▦ | 4.1 Develop a Model for Predicting the Time, Bandwidth and Power Required to Migrate Datacenter Workload | 409 days? | Thu 1/26/12 | Tue 8/20/13 |
| 52 | ▦ | 4.2 Experiment with Virtual Machine (VM) Migration to/from a POD | 347 days? | Mon 9/3/12 | Tue 12/31/13 |
| 53 | | **TASK 5 – Development of Prototype System and Field Demonstration Specification** | **86 days?** | **Mon 9/3/12** | **Mon 12/31/12** |
| 54 | ▦ | 5.1 Development of Prototype System and Field Demonstration Specificatio | 86 days? | Mon 9/3/12 | Mon 12/31/12 |
| 55 | | **TASK 6 – Field Demonstration** | **304 days?** | **Thu 11/1/12** | **Tue 12/31/13** |
| 56 | ▦ | 6.1 Design of Field Demonstratio | 129 days? | Thu 11/1/12 | Tue 4/30/13 |
| 57 | ▦ | 6.2 Field Demonstration of Prototype System | 218 days? | Fri 3/1/13 | Tue 12/31/13 |
| 58 | | **TASK 7 – Development of Design Guidelines for Subsequent Demonstrations** | **43 days?** | **Fri 11/1/13** | **Tue 12/31/13** |
| 59 | ▦ | 7.1 Development of Design Guidelines for Subsequent Demonstrations | 43 days? | Fri 11/1/13 | Tue 12/31/13 |



## e. Dissemination and Meetings

Several presentations were made during 2012 to different forums as presented in the past quarterly report. The team is working on a number of presentation/publications that will be used to disseminate the finding of this research and promote the GDC concept.

S. Bird completed a draft concept paper to be submitted shortly for publication. Contributors including faculty and students at CU and UT Austin and colleagues at AMD are currently reviewing the document. This paper outlines the general conceptual approach of the POD project and lays out the research agenda for the project. It also defines some of the challenges for this model in at-scale implementation. NYSERDA will be consulted prior submission to a journal. The team is evaluating potential opportunities for the GDC project with NSF including the System Science program and the Grant Opportunities for Academic Liaison with Industry (GOALI) project. The team has been invited by Brookhaven National Lab (DOE) to participate in the Advanced Energy 2013 Conference, currently scheduled at the end of April 2013 in NY City. The GDC concept will be presented in a session on Grid Integration. Graduate students are also attending this meeting and presenting posters. O. Aitmaatallah and P. Werden are presenting posters on Algorithm Development and on System Integration respectively.

## f. Additional Activities

S. Bird has brought another graduate Master's student, Paul Werden, from Queen's University (Canada) to assist in analysis of the project. Paul is currently being co-advised by Prof. Bird and Professor Andrew Pollard (Queen's Research Chair in Fluid Dynamics and Multi-scale Phenomena). Werden's research is focused on optimization models to determine ideal sizing of transmission capacity, storage capacity, and POD size for a given renewable energy source. His work will include empirical analysis of cases studies in both New York and Ontario, and include aspects of the grid system operations in his analysis (regulatory constraints, reliability standards, renewable incentives, and green energy policy). Paul is currently self-funded through his Canadian University. His research started in early September.

Two case study locations with solar PV systems have been selected for economic and policy analysis. The first is a 142kW system is located in Ithaca, New York on the roof of Tompkins County Public Library. The second is a 250kW system in Kingston, Ontario on multiple buildings at St. Lawrence College. Recorded hourly generation output data for multiple past years is available for both of these systems. The generation data will be paired with the corresponding price of electricity provided by the NYISO and IESO (Ontario).

An hour-by-hour simulation will model the operation of the GDC for one year using the collected data. This simulation will analyses the economic impact of various sizes of computing power, energy storage, and transmission capacity. The economic analysis will drive discussion concerning the effect of renewable energy policies on GDC systems. Policies include the Ontario Feed-in-Tariff, NYSERDA Renewable Portfolio Standard and Production Tax Credits.

End of Year Discussions with NYSERDA: Senior Investigators made two presentations to NYSERDA to discuss the end of the Year One status and discussion of an extended feasibility study by Professor Bird and colleagues. There are ongoing discussions with NYSERDA (Bryan Berry, Joe Borowiec) concerning a supplemental contract modification that incorporates a feasibility study, and economic and policy analysis to the existing contract.

## g. Analysis of actual cost incurred in relation to budget

Total cost incurred up to Jan 25th: 126,560.00 (NYSERDA portion)
Student Salary – $46,126.00 (budget allocation - $89,320.00)
Faculty Salary Statutory – 11,155.08 (budget allocation - $37,692.00)
Travel expense – $1,307.80 (budget allocation - $10,000.00)
Equipment - 29,717.00 (budget allocation - $40,000.00)
Indirect cost – $37,373.00 (budget allocation - $88,946.00)
Fringe - $880.00

Cost-share – **$86,000.00** (Clarkson) Estimated

| | |
|---|---|
| Student Salary | $17,500.00 |
| Student Tuition | $60,000.00 |
| Summer Students | $5,000.00 (McNair Scholar) |
| Travel | $3,500.00 |

Cost-share – **$65,000.00** (Partners) Estimated

| | |
|---|---|
| AMD Corporation | $40,000 |
| HP Corporation | $10,000 |
| WindE Systems | $5,000 |
| Ioxus | $5,000 |
| AWST | $5,000 |