Paper 1684-2014

SAS® Grid – What They Didn't Tell You

Manuel Nitschinger, Analyst / Risk Solutions Manager, sIT Solutions

Phillip Manschek, Technical Architect, SAS Institute Austria

ABSTRACT

Speed, precision, reliability – these are just 3 of the many challenges that today's banking institutions need to face. Join Austria's ERSTE GROUP Bank (ERSTE) on their road from monolithic processing toward a highly flexible processing infrastructure using SAS® Grid Technology.

This paper focuses on the central topics and decisions that go beyond the standard material about the product that are presented initially to SAS® Grid Technology prospects.

Topics covered range from how to choose the correct hardware and critical architecture considerations to the necessary adaptions of existing code and logic - all of which have shown to be a common experience for all the members of the SAS® Grid community.

After making the initial plans and successfully managing the initial hurdles, seeing it all come together makes you realize the endless possibilities for improving your processing landscape.

INTRODUCTION

In 2012, s IT Solutions was faced with massive performance requirements and additional services, which their customers within ERSTE GROUP expected for the beginning of 2014. At that time, their SAS® platform was running on a SUN SPARC 6 system with three servers, split into eight zones (see Figure 1). Due to the existing processes and hardware landscape, providing those feats on the then current UNIX-platform was not possible. At that time, a decision was made to go for a new hardware-architecture and to start the search for an architecture which would fit best. During the first months, existing processes as well as upcoming needs in terms of requirements and performance in the near future were analyzed. Knowing all the relevant parameters, they were able to pick two different systems, which, according to opinion and analysis, would fit best.

The next step was a proof of concept phase, testing both choices to make a final decision, which was accompanied by a TCO calculation for a 5-year period. Six months after starting the project they closed the analysis and decided to implement a SAS® Grid Computing Platform based on an x86 server-architecture.

This paper gives an overview of the questions and challenges that the team was faced with, shows changes that had to made to processes and solutions, lists important decisions and summarizes, which improvements were gained during the implementation of the new platform.

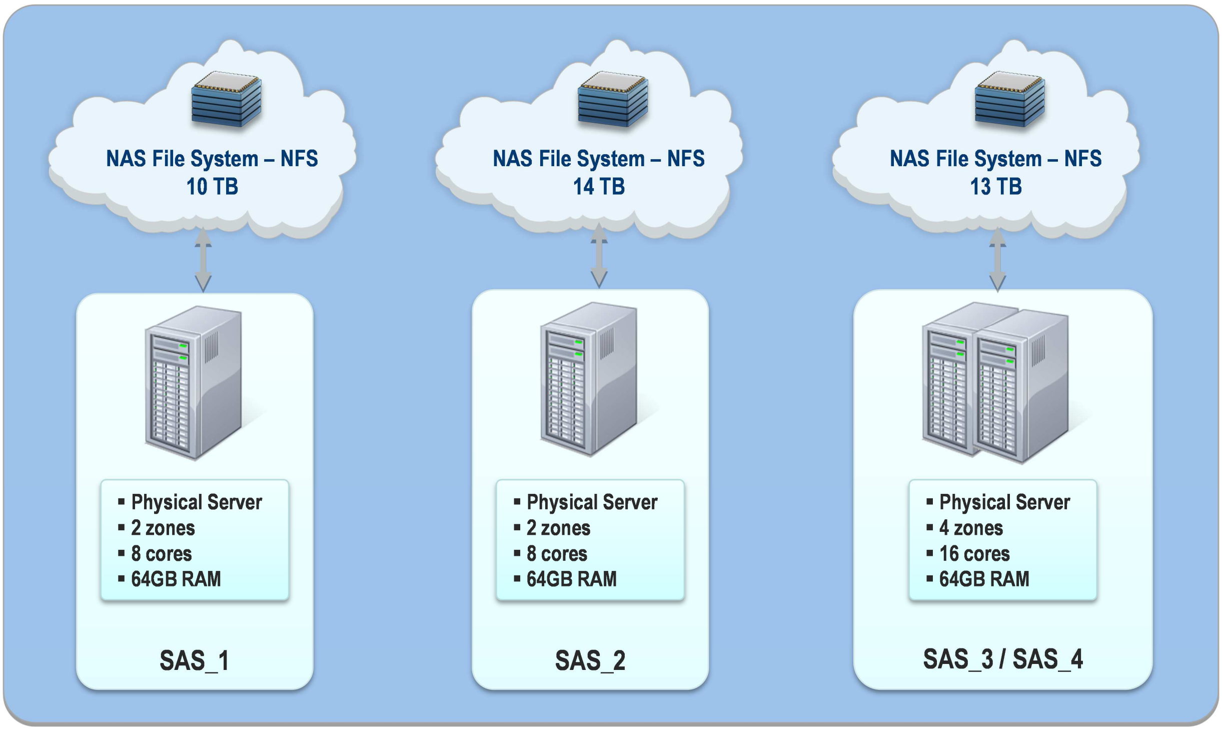

Figure 1. Technical architecture of the previous platform

UNIX OR X86 ARCHITECTURE?

When you think about a new architecture for your platform, many different considerations have to be taken into account. Before starting the design of a new system, you have to learn more about your current environment first. An existing hardware-platform can also influence your decision making process due to facts like upcoming reeducation cost if you choose another platform – maybe an extension of your current system would be the best solution. You have to know exactly which functionality your new system has to have and how to prioritize them. Objectives for consideration are:

- performance (run-times, number of users, …),
- reliability and availability (are outages allowed and for which time period),
- scalability (possibility needed to scale your system up and down – also within a short time frame and in various sizes) and
- cost of the system (which regulations are set by stakeholders).

Also company policies can influence your decisions (e.g. OS preferences). You should classify which processes are business critical and which ones are not, how many users are power users and how many of them are standard users with less performance needs.

During the first phase of the project, we have analyzed our processes, solutions and the needs of our users to exactly define our objectives. We figured out, that our batch-processes and solutions could be separated into smaller processing-parts, which gives us the opportunity to distribute them to several servers. The same can be done with interactive users, both of which were main reasons for choosing the SAS® Grid technology. The capability, to be able to scale up in a very quick time frame, was also an important factor in choosing new hardware. In our company there was no strict policy regarding the employed operating system, nevertheless there were preferences to Linux and smaller machines within a virtual server-network (as can be seen in many enterprises today). Nowadays, cost reduction is always a key decision factor and so it was in our company.

Taking all those parameters into account, as well as the measurements that were made during our proof of concept phase, we decided, that a physical server-assemblage of x86 servers would fit better to our needs of smaller, distributed processes and a very flexible architecture. Also business case calculations of the return on invest showed benefits with smaller servers compared to big Unix boxes. The possibilities of avoidance of outages due to the split onto several machines and features that the SAS® Grid technology brings were enough to ensure our service level agreements in the near future.

WHICH SHARED FILE SYSTEM FITS BEST?

As already stated in previous sections, it is crucial for the success of a new architecture to define the goals it will have to achieve. The key metrics for deciding on a shared file system in the project were the following:

- number of nodes in the grid system and potential future growth
- amount of data to be handled
- OS platform
- employed storage system
- access to documentation, trainings and support resources
- previous experiences from other projects
- procurement time
- vendor relationship and existing contracts
- cost

There is a growing number of shared file systems on the market today and SAS has been testing a variety of the available choices as well as collaborating with some providers. The results of these tests are shared in a regularly updated paper called "*A Survey of Shared File Systems"* which also contains configuration best practices.

IBM's GPFS can be called the no-brainer choice in most cases simply because it is available on most operating systems, has been in the market for quite some time and delivers good results while being extremely scalable. However, all of this comes with a price tag and in order to produce a viable business case, other alternatives were looked at as well.

NFS is well known, often used and enjoys a good reputation within IT departments but experience in high performance computing environments is mostly scarce. Since speed was an essential design criterion in the project, the attractive price did not make up for the risk of not completing calculations in time.

Redhat as a vendor was already part of the architecture by providing the server's operating systems, so naturally its shared file system GFS2 came into consideration in hopes of getting an integrated solution.GFS2 does not share GPFS's rich history but has come very far in the past. It is currently limited to 16 nodes accessing a file system and 100 TB of administered storage, both of which fit the project's needs. In terms of performance and stability, there were known issues from other projects in the past but SAS has been collaborating with Redhat to improve on GFS2 so that it can now be considered a viable choice. Combined with an attractive price point, these considerations led to GFS2 being the file system in the project.

It is also important to note that simply using a shared file system does not make your solution a compute cluster. Distributed locking does not solve problems that result out of concurrent access to files. Adaptions to code/jobs have to be made to avoid concurrency issues.

CLOUD OR COMPANY DATACENTER?

Cloud computing currently echoes throughout the planet! The benefits of this concept seem to be endless and so we also investigated, and still do, the advantages and the technical possibilities of integrating a cloud computing platform into our SAS® Grid system. The good news is that technically this is feasible with little effort and within a very short time frame. If you choose a standard cloud vendor (e.g. Amazon EC2), there are even standardized interfaces ready and in place which give you the opportunity to add and remove additional capacities within minutes. This solution has huge benefits in terms of server costs because you only have to pay for the hardware during the time it is actually needed and running.

While from the perspective of the underlying business case, the idea of using cloud hardware is very attractive, it cannot be understated that the data being worked with belongs to a banking institution. This lead to company policies being carefully reviewed to get the position on the usage of external cloud vendors and their data storage. As you might know, banks have very strict restrictions regarding the privacy protection of their customer data, and so do we. Most of the time during the analyses of this topic, the security department of the company had to be involved and in the end, we were not able to integrate an external cloud vendor during the first phase of our SAS® Grid project. However, we are still in the process of evaluating the use of external clouds for other purposes.

Beside the big advantages of external clouds, there are also some drawbacks and potential architecture risk. The most important one is the transfer-rate of the network between your datacenter and the one of the cloud vendor. An analysis in the project has shown that there is no benefit in absolute run-times if data has to be transferred to the cloud for each processing and then recouped to the datacenter after results are available (normal batch-processing). This is because the transfer times cut the gains in runtime that were achieved by “cloud burst”. Constant mirroring between internal and cloud storage is also not technically feasible because of implied latency and bandwidth issues.

There are of course scenarios that benefit greatly from cloudbursts, for example if you only transfer the data once, process it several times and only load back the results, the benefits would increase greatly. Stress-test simulations with different calculation-scenarios, which are always based on the same input data, can be a good candidate for using external cloud systems. Also data preparation for user-requests might be a good candidate. These two scenarios are the ones we want to bring on external cloud systems, as they are only necessary a few times during the year. Potential savings throughout the rest of the year will of course only be possible if security issues can be properly addressed.

IS THERE A NEED FOR A TEST-GRID?

This question cannot be answered in general as it depends on the requirements you are facing.
The current project setup only holds one application in the Grid environment, which means that the impact on business is well known at every point in time. Emergency reboots for example, while painful, can be done because all involved parties are a tight-knit group within the company. With further growth, this will change and introduce the need for more environments but in the current state of the project, there was no need for separating all environments on distinct Grid systems.

When we designed our system, we wanted to achieve the maximum flexibility in terms of resource-usage, which ultimately means sharing as much as possible. We therefore put all environments (development, testing, production and simulation) on a single Grid instance.

As proper configuration management dictates, there has to be the possibility of testing changes without affecting the productive system. In our case, this means that in addition to the big Grid system housing all SAS environments, there is a smaller one used for testing patches and hotfixes for the OS, SAS Software and LSF.

This setup greatly increases stability because changes that affect the underlying platform can be tested in advance but it also enables the maximum use of available cores.

HOW ARE MULTIPLE ENVIRONMENTS SEPARATED ON A SAS GRID ARCHITECTURE?

Beside the topic of grid configuration and the need for a test instance, configuration management has to be established for the SAS implementation as well. There are many ways of separating SAS environments in the context of grid computing, but a small number of factors determines the choice in the end:

- IT policies regarding the storing of data (i.e.: randomized data and its original version may not reside on the same storage system)
- IT policies regarding security (i.e.: is logical separation with different user-accounts sufficient)
- SAS license (i.e.: which modules are licensed for which number of cores)
- Ease of maintenance (a single installation on a shared file system vs. one installation per node)
- High availability and time to deployment considerations (how fast do I want to be able to add a node)

In the end, it was decided to have four environments (development, test, prod, simulation) running in the production grid system. Each is installed into a separate folder structure with a specialty being that there is a single set of users accessing them.

This solution enables administrators to quickly add a node by setting up the OS and then mount the necessary file systems. SAS services are only running on one head-node, which means that the only references to actual hostnames are in the LSF configuration.

To ease access for end-users it was decided to not split up their accounts and make them change/remember 4 different sets of credentials. Since this could potentially mean that a test-user that can access production-data if he or she knows the location on the file system, a way of separating the folder structures was needed. Each environment belongs to a different OS-group and users that are allowed to access it belong to the group. To solve the problem of potential crossovers between environments, a UNIX mechanism called shadow groups is used to ensure that a user is only able to access data in the context of his or her current environment.

HOW WILL THE INSTALLATION LOOK LIKE AND WHERE WILL IT BE LOCATED?

As already described earlier, there is one set of installation + configuration for each of the four SAS environments. All resources are located on the shared file system and mounted on each node. One node is the designated “Grid Control Server” that is running LSF batch daemons, as well as SAS Services. This machine – unlike the others – is a VM, configured as highly available on a VMWare ESX cluster. Workspace Server and others are configured to launch into grid sessions as early as possible, which means that access to the SASCONFIG directory exists mainly to read solution specific content. The single exception to this setup is the standalone SPDS server machine, which is part of the overall landscape but not included into grid processing. Each node has its own local WORK directory placed on an SSD drive to ensure low runtimes for jobs that are not using the shared file system for checkpoint – restore functionality.

It is possible to run SAS services across all nodes, but if the installation is to be shared among them, there is additional effort needed to adapt start-scripts and make them hostname-sensitive.

A configuration like this enables the addition of a new grid node in little time, which was desired under the goal of being as flexible as possible.

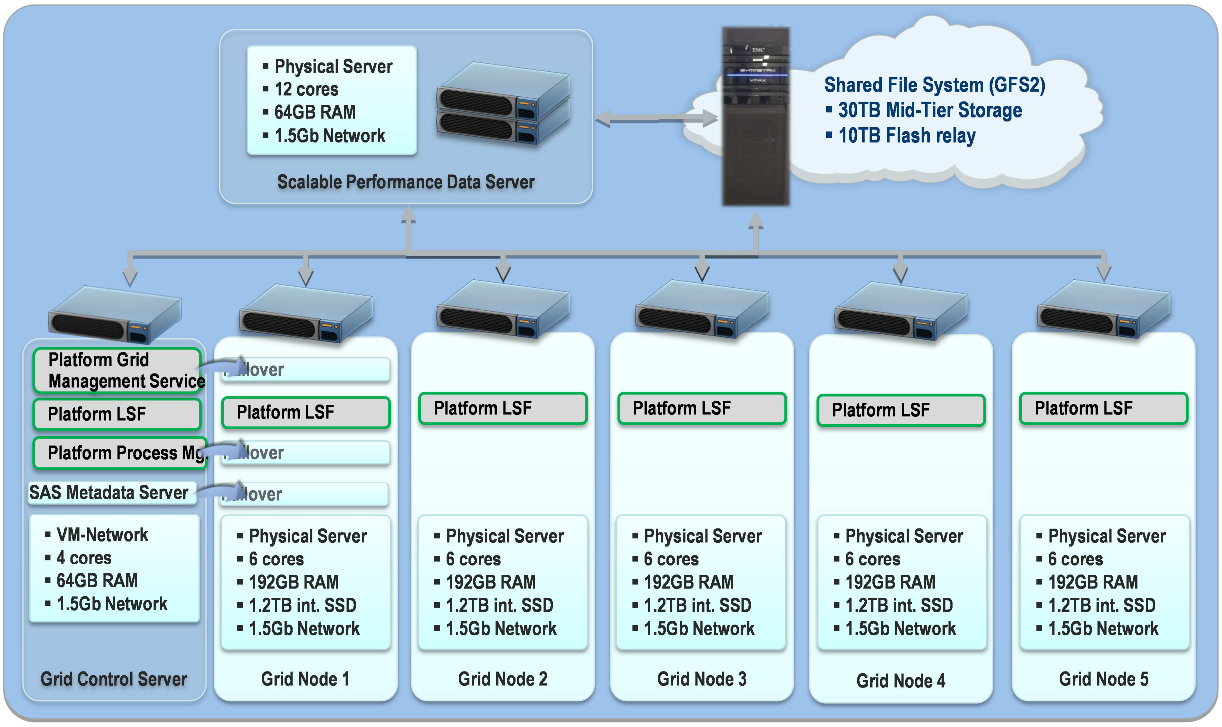

Figure 2. Technical architecture of the new platform

CAN THE ALREADY EXISTING SCHEDULER BE USED INSTEAD OF LSF?

Depending on a project’s needs, this question can be religious rather than technical. It is important to be aware of what is possible and what is not at the beginning of a planning phase.

Like most banks, ERSTE GROUP has an established scheduling system for planning and executing job-flows. Due to the fact, that this scheduler is used by all systems prior to and after the SAS solutions, we planned the use of this scheduler within the new SAS® Grid Manager platform in the same way it was done before on the existing SAS platform.

While integration with non-LSF scheduling systems is technically possible, the effort needed to expose jobs and flows to such a system can be high. The SAS Grid Submission Utility (SASGSUB) can start jobs in the grid system but building flows around such a vehicle can take time.

Unfortunately, the only other option (= achieving seamless integration of a scheduler to the SAS solution running on grid) was to rebuild all job-networks that were previously managed by ERSTE's scheduler – a previously unexpected effort that took a toll on the project plan.

Not only was the actual work necessary, there were organizational consequences as well because the internal department responsible for operating and monitoring jobs was not prepared to handle another scheduling solution.

As the SAS world suddenly became a black box to this department, there was the need to a) somehow communicate with the existing job-networks around SAS processing and b) find a replacement for the previous ticketing system to notify administrators and developers in case of failure.

The latter issue was solved by LSF's feature of sending messages (Email, SMS) in case of certain events. Integration into the processing landscape was achieved by using file-trigger mechanisms of both scheduling systems.

This means that SAS is being started by files written to a specified location and then "returning the favor" by writing triggers for the enterprise scheduler.

Even though it works nicely, this was not the preferred implementation, but it allows us to link the internal job scheduler to the now used LSF Scheduler within SAS® Grid.

WHICH FEATURES CANNOT BE USED BECAUSE OF COMPANY POLICIES?

Deep integration of Platform LSF into SAS is the main characteristic of SAS® Grid Manager. Integration with other scheduling systems has already been covered and other projects have in fact used a construction that combined multiple systems this way. Besides scheduling, there are other potential mismatches to company's IT policies.

SAS® Grid Manager comes with a wide range of high availability features that are powered by the enterprise grid orchestrator (EGO) which lets you script many contingency actions to move processes and start services. Since this is a scripting framework, it could potentially provide a wholesome solution for high availability where upon failure a node could not only take over services, but also IP-addresses and DNS-aliases. However, dynamic changes of network configuration and name servers by a third party tool is something that IT departments are rarely comfortable with. To ensure uninterrupted service, a decision was made to not use EGO (although it has been tested successfully during the POC phase) but have a "grid master node" that runs inside a highly available virtual machine.

Another issue that was discussed but ultimately dismissed was the technical foundation underneath the monitoring and configuration tool RTM. It is based on open source components, which means a MySQL database engine for which there was no operations expertise available. Since RTM can be run out of a pre-configured image that does not require deep knowledge about the database, its' use for the project was ultimately not a problem.

WILL EXISTING CODE RUN WITHOUT MODIFICATION?

The simple answer to this question is, yes it will! However, that is only half the truth…

As in many cases, the response to this question will differ depending on it coming from sales people or technical consultants. *"Of course it will!"* can very quickly turn into the typical *"Yes, but …"*.

The truth is that every existing code will run on a SAS® Grid system, but if you want to achieve the maximum possibilities of the technology, you sometimes have to adapt your sources, or - more specifically - the flow of the sources. An inexperienced user will think, that all the sources which are able to run in parallel, will run fully automatic in parallel only by sending them to the SAS® Grid system, however this kind of automatism is not the reality. You have to redesign your job flows within SAS® Enterprise Guide® as well as adapt your SAS® DI Studio® flows to get the full benefit out of the distribution technology. Within the SAS® DI Studio® there is an internal Grid analyzer, which can help you migrate your jobs to run in parallel. Within SAS® Enterprise Guide® projects, you have to take care of how you are implementing your flows. As an example, the following statements, written within one single program within a SAS® Enterprise Guide® project, will not run in parallel even though they are fully independent from each other:

```
/* Step 1 */
data work.table_one;
  set input.table_a;
run;
```

```
/* Step 2 */
data work.table_two;
  set input table_b;
run;

/* Step 3 */
proc sql;
  create table work.table_three as
  select * from input.table_c;
quit;
```

To achieve a parallel run of all three steps, and to get faster results due to parallel processing instead of one after the other, you have to put each of them into a separate program file and then implement a processing flow within the SAS® Enterprise Guide® project designer.

As you can see, the act of licensing SAS® Grid Manager and installing it on your hardware is just the beginning.

WHAT NEEDS TO BE DONE IN ORDER TO FULLY BENEFIT FROM A SAS GRID TECHNOLOGY PLATFORM?

The previous chapters of this paper have already given a short overview of what has to be done in order to fully benefit from a SAS® Grid Manager platform. The most important thing all SAS® Grid prospects have to do before they begin planning and starting their migration project is analyzing their processes and programs to get an idea of the efforts needed to adapt sources and flows so that they can fully benefit from SAS® Grid technology.

PART 1 – ANALYZE AND SEPARATE YOUR PROCESSES!

We did this in a pre-project phase to get exact estimations for the migration project and to be able to plan it properly. Of course, we could have only migrated the existing sources and flows to the new SAS® Grid system, but then we would not have gained even a fractional amount of the possible benefits. First, you should start identifying sources, which can be separated or split into several independent parts. This adaption gives you the most benefits because then the SAS® Grid system is able to distribute them throughout all available nodes. It is also important to split long processes into more parts if possible to get smaller packages. It is always easier and makes your system much more flexible to distribute 100 small jobs than 3 big ones. Within our company we first decided to stop processing in packages (e.g. more bank institutes within one single run) to a fully tenant processing level. This was the first and most important change within our process-flow because due to that, the smaller tenants can be calculated separately and complete independent of each other in a much faster way. The next thing is to adapt your existing code to be able to run in parallel. Extract parts out of existing flows and redesigning the flows. During that phase, you should always be aware that not every single step will give you increased performance. So start with the “long-runners” and maybe spawn processes with big input data into several equal parts and merge them after every process has finished – be aware, that the split of the data and also the merge of the results will take additional processing time which may cut the gained performance.

PART 2 – DESIGN A PROPER QUEUING MECHANISM / QUEUING RULES!

The next step is to design proper queuing rules and mechanisms. To avoid, that a low-priority process puts the brakes on one of our mission critical processes we had to design queues and appropriate rules for those. First, we collected all possibilities of processes and flows within our SAS® system. Afterwards we weighted / rated them with their importance to the company. The last step to get the different queues and the corresponding rules is to categorize the collected processes into queuing groups and afterwards define the rules for and between those queues. Due to the fact, that our SAS® Grid system provides all of our environments (developing, testing and also production) we also have to consider the maximum allowance of system resource usage of each queue and the behavior of the queues between themselves. Meaning, which queue has to pause processing if the “high-priority” queue requests the maximum of its potential system resources while other processes from lower prior queues are using them.

PART 3 – BE SURE THAT YOUR HARDWARE CAN HANDLE THE NEEDED PERFORMANCE!

To be sure, that the hardware can handle the needed performance of the software, you have to test this during a bulk-test where high resource intense processes are running in parallel. With this test, you can figure out possible bottlenecks like network-connections, file-system limitations or all other limitations of the system. If you survive this test without any performance issue, you will know that your sizing was done well!

WHICH TECHNIQUES LEAD TO THE BEST PARALLELIZATION OF CODE?

As mentioned in the previous section there are many strategies for making big, long-running jobs small and agile. SAS offers several automatisms that make you immediately benefit from a grid-infrastructure. Tools like SAS® Enterprise Miner or SAS® Enterprise Guide are grid-aware and can start independent parts of projects simultaneously.

In addition to this, there is PROC SCAPROC, which analyzes code and proposes blocks that could run in parallel. While in theory, this is a powerful mechanism, the heuristics used can never replace the mind of an able SAS programmer, especially when it comes to complex macro code. The human component cannot be understated, especially when it comes to actual improvements by splitting code. There are cases where the overhead generated by splitting and merging completely expunge performance benefits while also making the code harder to maintain. Proper training for programmers is therefore a must in every SAS® Grid Manager implementation.

HOW DO WE SET UP QUEUING?

Workload management is one of the most important features that SAS® Grid Manager offers but at the same time, it can be a major obstacle in an implementation project. The general ideal for designing queues is “as complicated as necessary, but as easy as possible”. Queues will evolve over time and in many cases, it does not make sense to start with the most elaborate concept because it can lead to unexpected behavior in the early stages of a project.

LSF knows a wide variety of variables that can be used to define rules for job execution like timeslots, load and priority. Queues are then enabled on hosts and can have attributes like automatic restart in case of failure or preemption to move lower priority jobs on hold.

For our four environments running mainly batch-jobs, a simple setup with using different priority values and spanning across all grid-nodes is a starting point. As soon as interactive (EG) users start using the system, they will be equipped with their own queue, where it then becomes important to divide them from active batch runs, so that system usage remains high across the grid.

HOW IS A SAS GRID ADMINISTRATED AND WHO WILL DO IT?

Aside from the usual administration tasks that are common across all SAS® 9 Intelligence Platform environments, SAS® Grid Manager brings additional work that needs to be assigned to people. Technical implementation and requirements engineering on the end user’s side go hand in hand which means that at the time of introducing a grid platform it is also a good idea to think about establishing a SAS competence center inside an organization. Luckily, this has already happened before this project so that resources were in place and roles were easy to assign.

Technical administration can be done in a variety of ways that differ in the breadth of functionality they offer. SAS Management Console offers a plugin that can perform general operations like opening and closing hosts and show load information at a high level. Platform RTM can do many things and in most projects is the go-to tool for administering a grid system. Graphical representation of information as well as deep configuration is possible within a web interface. Configuration files and command-line tools from Platform Computing are the most comprehensive way of configuring and monitoring a grid system.

To ensure continuity in the way the grid system works, it is important that only a select few administrators can change queue-definitions and other configuration files. Monitoring jobs in RTM and Process Manager can be done by power-users who are able to restart crashed jobs but they should not be given permissions to re-assign jobs to different queues. In general, business should define priorities between different workloads that IT can then deliver in the form of fitting queues.

CONCLUSION

There are some obstacles in establishing every entirely new architecture and grid is no exception. Careful planning will minimize the pain immensely especially if future requirements are considered from the get go. This paper has shown some of the common decisions and criteria behind them to give you, the reader, some guidance even though your environment will very likely look a lot different from ERSTE GROUP’s.

ACKNOWLEDGMENTS

A big THANK YOU goes to all the people inside s IT Solutions who were willing to actively take part in the transition from a well-established system to a future-ready new infrastructure. It was not always an easy time but the results we produced speak for themselves.

RECOMMENDED READING

- *Using SAS® 9 and Red Hat's Global File System2 (GFS2)*
- *A Survey of Shared File Systems (updated August 2013)*

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Name: Manuel Nitschinger
Organization: s IT Solutions AT Spardat GmbH
Address: Geiselbergstrasse 21-25
City, State ZIP: A-1110 Vienna
Work Phone: +43 50 100 - 15982
Fax: +43 50 100 - 15982
Email: manuel.nitschinger@s-itsolutions.at
Web: http://www.s-itsolutions.at

Name: Phillip Manschek
Organization: SAS Institute Software GmbH
Address: Mariahilfer Strasse 116
City, State ZIP: A-1070 Vienna
Work Phone: +43 664 10 62 521
Fax: +43 1 25 242 590
Email: phillip.manschek@sas.com
Web: http://www.sas.com