Adaptive Control System for Server Groups in Enterprise Data Centers

Sven Graupner, Jean-Marc Chevrot, Nigel Cook,
Ramesh Kavanappillil, Tilo Nitzsche
Internet Systems and Storage Laboratory
HP Laboratories Palo Alto
HPL-2003-273
December 19th , 2003*

E-mail: {sven.graupner, jean-marc.chevrot, nigel.cook, ramesh.kavanappillil, tilo.nitzsche}@hp.com

enterprise, grid, adaptive, control, Utility Data Center, server, management, Web services

An adaptive control system automatically performs a loop over monitoring, assessment, and corrective action following a goal. A goal, for example, can be to keep the operational parameters of a controlled system within defined ranges.

This paper presents how Hewlett-Packard's Utility Data Center (UDC) is extended by a control system for server group flexing for horizontally-scalable applications. Capacity of server groups is automatically adjusted during operation by acquiring or releasing servers.

In contrast to similar control systems that operate in clusters, the control system here operates in the heterogeneous environment of a commercial data center. Joining and releasing servers from its virtualized application environments is inherently more complex and requires securely crossing protection domains.

The control system is based on Grid's web services standard OGSI leveraging OGSI's built-in security and event models. Adopting OGSI in the enterprise data center environment also introduces the opportunity for creating Grids in commercial enterprises.

* Internal Accession Date Only
Approved for External Publication

Adaptive Control System for Server Groups in Enterprise Data Centers

Sven Graupner, Jean-Marc Chevrot, Nigel Cook, Ramesh Kavanappillil, Tilo Nitzsche
Hewlett-Packard, 1501 Page Mill Road, Palo Alto, CA 94304, USA
{sven.graupner, jean-marc.chevrot, nigel.cook, ramesh.kavanappillil, tilo.nitzsche}@hp.com

Abstract

An adaptive control system automatically performs a loop over monitoring, assessment, and corrective action following a goal. A goal, for example, can be to keep the operational parameters of a controlled system within defined ranges.

This paper presents how Hewlett-Packard's Utility Data Center (UDC) is extended by a control system for server group flexing for horizontally-scalable applications. Capacity of server groups is automatically adjusted during operation by acquiring or releasing servers.

In contrast to similar control systems that operate in clusters, the control system here operates in the heterogeneous environment of a commercial data center. Joining and releasing servers from its virtualized application environments is inherently more complex and requires securely crossing protection domains.

The control system is based on Grid's web services standard OGSI leveraging OGSI's built-in security and event models. Adopting OGSI in the enterprise data center environment also introduces the opportunity for creating Grids in commercial enterprises.

1 Introduction

"Grid" has posed a vision on the IT industry that expands beyond traditional supercomputing. Grid's key words of collaboration, services, and virtual organizations also represent trends in enterprise data center environments.

IT infrastructure must become more adaptive, easier to deploy and to operate in order to support, and not hinder, collaboration among partners inside or outside business organizations. Changes in business and business partners must be accommodated faster by IT systems.

Unifying resources and applications under the notion of services also simplifies the data center environment.

Crossing technical and organizational boundaries in secure ways is encompassed under Grid's notion of virtual organizations, which may only last for the duration of a joint project or may last over years, and must be established and managed easily and yet securely.

Besides shared goals and vision, the shared technology base between Grid and enterprise environments in form of web services is in common as well. Web services have been used in the enterprise IT domain for some time and have become a solid and mature base for creating, operating and managing IT systems. The Open Grid Services Infrastructure (OGSI) [1] and the Open Grid Services Architecture (OGSA) adopt web services.

The commonality in goals, vision, and technology has made IT vendors and service providers lined up towards the Grid, yet recognizing the differences between supercomputing where Grid technology has emerged and enterprise IT environments. As one result of adopting the Grid vision, IT vendors have made announcements to grid-enable their products and use open grid technology where reasonable. The Grid has the potential to solve real business problems by simplifying global operation and access to enterprise computing services.

This paper presents how OGSI technology is used in Hewlett-Packard's Utility Data Center (UDC) [2] solution for a specific purpose: to create an adaptive control system that performs automatic server flexing for horizontally-scalable enterprise applications.

Horizontally-scalable applications are simultaneously operated on a number of servers of same type. Examples are web servers, application servers or clustered databases. The number of servers is adjustable statically or dynamically according to demand or other reasons, a process which is also called *server flexing*.

In this paper we present a control system for dynamic adjustment (flexing) of servers provisioned for a horizontally-scalable application. The adaptive control interface as part of the control system is scheduled for the next UDC product release in February 2004.

This paper addresses three main aspects:

- Adaptive resource management in enterprise data centers by creating control systems for various purposes such as for automatic server flexing.
- Employing OGSI as platform for securely crossing protection domains in which the various components of the adaptive control system reside.
- Establishing the base for Enterprise Grids [3] which allow linking several data centers in an enterprise together using OGSA/OGSI open standards.

The work is based on previous work on introducing Grid technology into enterprise data centers [4], [5], [6], [7].

2 Resource Requirements for Commercial Applications

Commercial applications differ from job-oriented compute applications typically found in Grids in many ways. Commercial applications are inherently more heterogeneous in terms of application components requiring different operating systems, different server platforms, external devices such as firewalls and load balancers. Installation, configuration and management of commercial applications is complex since a multitude of hard- and software components is involved in an application environment. Workloads and use patterns are different as well. Commercial applications mainly serve transaction workloads and typically operate permanently.

Figure 1 shows a typical example of a commercial n-tier application with the variety of involved resources.

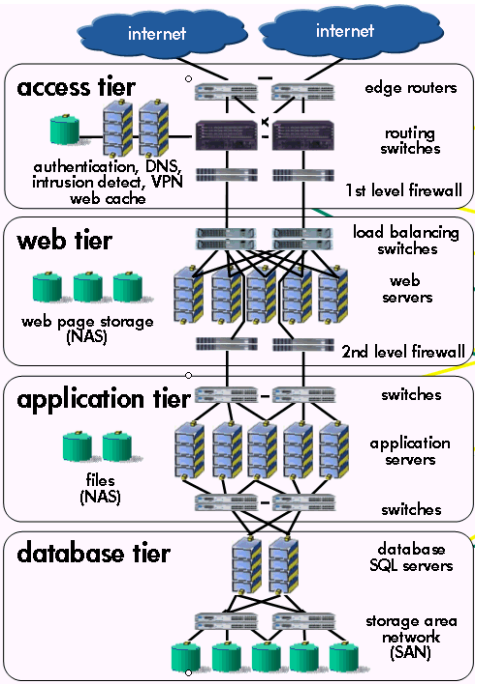

Figure 1: Typical multi-tier commercial application.

Since multiple application environments are operating in a shared infrastructure in a data center, commercial applications have strong requirements of shielding and isolation, typically achieved by virtual LANs (VLANs).

Resource requirements for commercial applications are:

- Commercial applications require heterogeneous resources and application components forming an *application environment*.
- The application environment requires configuring and managing the heterogeneity of resources and application components.
- Application environments co-exist in a data center and must be shielded from one another for fault and performance isolation and for security.

Complexity resulting from these conditions makes automation and adaptive control functions hard to achieve in commercial data centers.

Virtualization is a base technology to address some of these issues and also forms the base mechanism on which the adaptive control system relies.

3 Resource Virtualization

Virtualization may occur at any layer, from applications to resources. Turning an application into a service and shielding clients from its implementation behind a well-defined interface and access point basically means that the services abstraction virtualizes applications. The service access point controls the indirection between service clients and the applications providing the service. Controlling the service access point can be used for a variety of purposes: shielding service clients from application migration, version upgrades, service capacity adjustments, etc.

Similar indirections are introduced at lower resource layers for creating virtualized resources. Indirection points linking virtually created resources to their physical underpinnings can be altered during operation with the effect that application components are decoupled from physical hardware resources.

Resource virtualization can occur at several levels:

- virtualization of machine resources such as virtual memory or virtual devices, typically provided by the operating system,
- virtualization of “software resources” such as file systems that can be mapped to different file servers, or web services,
- virtualization of entire machines as layer between a native machine and virtual machine instances with operating systems and applications, and
- virtualization of an entire resource environment for larger, multi-machine application deployments in enterprise data centers.

4 Resource Virtualization in the Utility Data Center

The Utility Data Center is a data center solution which virtualizes the resource environment in a data center for entire application environments by controlling the indirection points provided by switches in VLAN and storage area networks (SAN). The UDC Controller

maintains all control points that are shown as little blue squares in Figure 2. The UDC Controller maintains storage in external storage arrays and attaches exposed logical volumes to servers by programming the SAN switch. It also configures devices such as hardware firewalls or load balancers.

An application environment is called a *farm* in the UDC. A farm spans across all tiers and includes all its resources.

Resource virtualization achieves a variety of effects:

- Farms are shielded from each other by placing them into separate virtual LAN environments.
- VLANs provide applications with private IP address spaces avoiding reconfiguration of application components when they migrate between servers.
- Application components are untied from server machines by maintaining their entire persistent state in logical volumes in the external storage arrays. SAN switches control attaching logical volumes to the server machines assigned to a farm.
- Device configuration allows to assign and to configure devices such as load balancers or firewalls to farms.

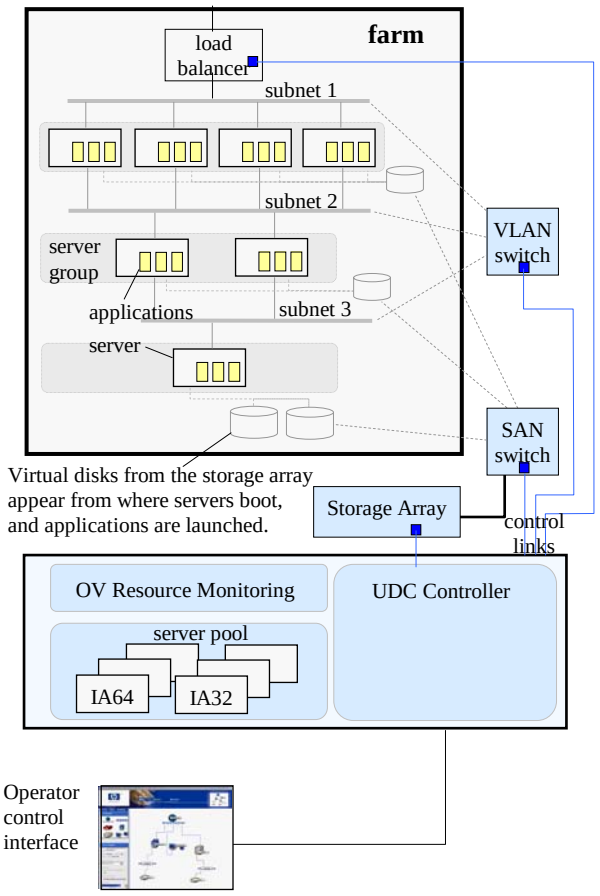

Figure 2: Virtualizing resources for a farm in the UDC.

Figure 2 shows how storage and network resources are virtualized in the Utility Data Center based on controlling VLAN and SAN switches.

Since applications become untied from machines and the physical network-configuration, applications can be migrated to other servers by terminating them and saving their persistent states to the associated volumes in the storage arrays. Reattaching those volumes to a new server machine, rebooting from the contained images, and reconfiguring the virtual network such that the new server machine appears under the same virtual IP address as the previous machine, allows to bring applications up on a new servers without awareness of the application.

The control software (UDC Controller) maintains exclusive control over VLAN and SAN switches, which also eliminates the risk of missconfiguration by operators.

The UDC Controller maintains a pool of server resources that can be assigned to application environments on demand. Server resources are of heterogeneous types such as IA32, IA64, or PA-RISC. Servers are selected from the resource pool based on descriptions provided by specific application environments.

Within the farm shown in Figure 2, the three tiers are separated by three subnets 1, 2, and 3. Subnets are also established by the UDC Controller by programming the VLAN switches accordingly.

Horizontally-scalable applications run instances on each server in a tier. Those servers are called a *server group*.

Server groups may have a fixed or a varying number of servers. Adjusting that number of servers is called *server flexing* and is the essential capability of the adaptive control system described in this paper.

HP OpenView is used in the UDC for collecting basic resource monitoring data. The adaptive control system can use this data as input for assessment.

5 Server Group Flexing

As shown in Figure 2, a server group consists of a number of servers of same type, running the same version of the operating system and application, both originating from the same disk image. Life-cycle scripts are executed when the operating system is booting for the final configuration (Figure 7). Each server uses its own copy of a master image. The process of creating that copy is in the range of minutes limiting the periodicity of the control loop to 10's of minutes or hours, which is adequate to accommodate longer-term daily, weekly or monthly patterns in commercial workloads.

Reasons for flexing the server number up might be that application load is increasing and servers are approaching saturation levels. Reasons for releasing servers from

server groups might be that servers are needed for other applications. Servers operated under a pay-per-use regime provide incentives to farms and their owners to free resources when they are not needed.

Server flexing can occur:

- Statically, by stopping the entire application environment, performing all necessary reconfigurations for joining or releasing servers to or from a server group, and restarting the application environment with the new configuration. Static flexing is typically performed manually today.
- Dynamically, by joining in new servers into a server group of an operating farm and gracefully releasing them from an operating farm. Dynamic flexing relies on automatable reconfiguration capabilities supported by virtualized resource environments such as UDC.

5.1 Server Flex Control Loop

In a server flex control loop, flex operations are initiated by a control system that continuously oversees the conditions in the farm (based on load, use patterns, failure conditions, etc.) and evaluates conditions in order to come to a conclusion to increase or decrease the number of servers in a server group. Figure 3 shows a control loop for server flexing. The control loop consists of three main stages monitoring, assessment, and adjustment.

Monitoring. – Monitoring is typically based on elementary resource metrics such as CPU utilization, process queue lengths, used network bandwidth, memory utilization and swap rate. These measures are collected in the Utility Data Center by OpenView. If application instrumentation is provided, application-level metrics can be taken into account as well such as transaction rates, response times, numbers of simultaneous sessions, etc.

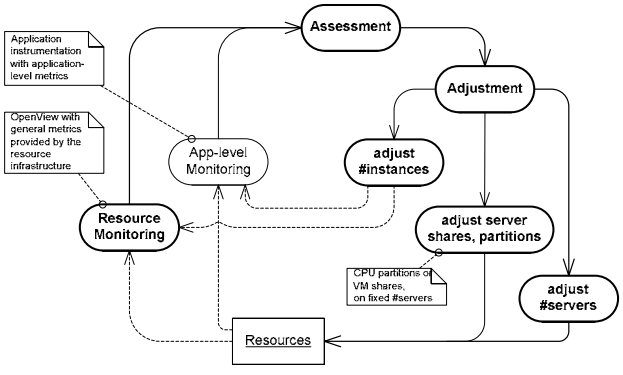

Figure 3: Server flex control loop.

Assessment. – Given measurements as input, the conditions in a server group can be assessed whether the system is considered operating within defined bounds or not. Automating assessment can be simple such as observing thresholds or it can be complex. The behavior of an assessment system is defined as policy. Policies must be customizable.

Adjustment. – As a result of assessment, a decision about a corrective action is derived, which leads to an adjustment in the system. Various means for adjustments exist for horizontally-scalable applications:

1. The number of instances of an application can be increased or decreased.
2. On certain hardware, CPUs or CPU partitions can be added or removed for an application in order to adjust processing capacity. Virtual machines allow adjusting CPU shares between virtual machines.
3. The number of physical machines can be adjusted (which remained constant in the first two cases).

Virtualizing resource infrastructures such as UDC have the capability to dynamically (during the operation of an application) add or remove physical servers from an application environment. It has the capability to configure servers into that environment, or release and unconfigure them from the environment.

The outer-most control loop in Figure 3 encompasses the acquisition and release of entire physical servers from the resource infrastructure.

The three dimensions of adjustments also reflect different levels of granularity in terms of resources (CPU cycles, CPU partitions or CPUs, or entire servers) and time scales of shorter-, mid-, or longer-term control loops, which may co-exist. When a finer-grained loop has reached its limits, the control loop of the next level can expand its reach.

6 Adaptive Control System for Server Group Flexing

The purpose of the adaptive control system for server group flexing is to provide horizontally scalable applications the ability to acquire or release physical server resources using the capabilities of the Utility Data Center to configure and unconfigure servers from the virtualized resource environment of a farm.

6.1 Requirements

A number of requirements have been considered for the definition of the adaptive control system:

1. *Automation* driven by policy. – The control system operates without involvement of the operator after the operator has set the control policy.
2. *Application focus*. – The control system, specifically the assessment and the decision-making part, must be customizable for specific applications and operating regimes defined for a data center.

3. *Open-standard's-based interface* and ability to integrate with application management and control systems. – Applications are built more and more for utility modes of operation. It means that they incorporate interfaces and components for their management and control. Application vendors enrich their applications with such management systems. One major requirement of the adaptive control system here was being able to integrate with those application management systems providing them the link, based on open standards, into the resource infrastructure where they can acquire or release configured servers.
4. *"Plug-and-Play"*. – Application integrators or data center operators must be able to define control loops easily. A "pluggable" component design based on open standards should form the foundation.
5. *Trust and Security*. – Control functions in data centers are hard to automate not only because of technical complexity. Automating tasks relies on trust which operators must develop in order to delegate functions they perform manually today to automated systems. Security breaches and malfunction are main concerns that must be prevented by automated systems.
6. *Integration into higher-ordered control systems*. – The control system discussed here addresses the rather limited scope of flexing individual server groups for horizontally-scalable applications. Higher-ordered control systems can expand the scope to entire farms, farm groups, data centers, or even across data centers. Those scenarios have been considered for the design, but are not discussed here.

6.2 Design Choices

The following design choices are guided by requirements:

- Split of the control loop into a fixed infrastructure part embedded in the resource infrastructure providing server configuration, basic resource monitoring, and a customizable *Control Plug-in* containing the assessment and adjustment functions.
- Defining an open-standard's-based interface between the Control Plug-in and the resource infrastructure.
- Selecting OGSI as interface and interconnect technology mainly because of openness, web services standards, and built-in security and event models.
- Simple interaction patterns between the Control Plug-in and the underlying resource infrastructure based on service invocations and events.

The following figure shows the split of the control loop into Control Plug-in and resource infrastructure.

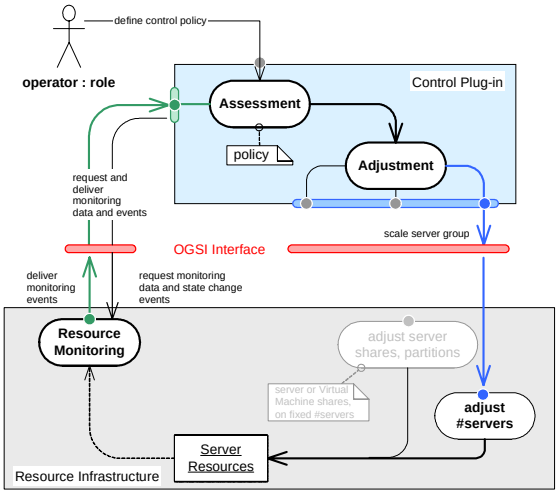

Figure 4: Component split of the control loop.

The OGSI interface between Control Plug-in and resource infrastructure primarily has two main functions:

- Request and delivery of basic resource monitoring data (provided by OpenView in the UDC) and delivery of state change events such as a change in the number of servers currently participating in a server group.
- Request to scale a server group up or down to a desired target number of servers.

Control policy is defined by the operator or the application administrator, in the current system in form of simple thresholds that are parameterizable for Control Plug-ins, which are implemented as Java OGSI clients.

6.3 Design of OGSI Components

The split of the control loop into a part that contains assessment and adjustment (Control Plug-in) and the resource infrastructure delivering monitoring events and providing the capability to actuate server group adjustments leads to the following design of components that are implemented as OGSI services:

- *Control Plug-in* – implements assessment and making adjustment decisions, contains control policy, and is customizable by application administrators or data center operators.
- *Interface Instances* – counterpart for one Control Plug-in in the resource infrastructure. It provides port types for actuating flex decisions and delivery of monitoring and state change events.
- *Interface Factory* – provides a port type for creating Interface Instances. A Control Plug-in initially contacts the Interface Factory in order to create an Interface Instance, which it then uses for all subsequent interactions.

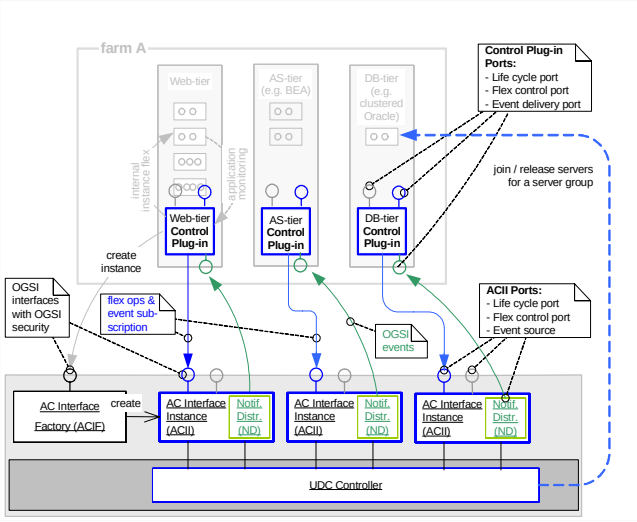

Figure 5: OGSI component design and basic flows.

OGSI's event and life cycle mechanisms are used for OGSI components. Each has a number of port types:

Control Plug-in:

- Event Delivery Port – primary port for event delivery from the associated Interface Instance or other event sources such as application-level instrumentation.
- Flex Control Port – allows a higher-ordered control systems to connect to and instruct a Control Plug-in.
- Life Cycle Port – for managing the life cycle of the control loop (start, stop, suspend, resume, etc.) and the Control Plug-in.

Interface Instance:

- Flex Control Port – primary port for receiving server flex operations from the associated Control Plug-in.
- Life Cycle Port – port for life cycle operations on Interface Instances.
- Event Distribution Port – port where Control Plug-ins can subscribe for monitoring and state change events.

Interface Factory:

- Factory Port – initial contact point for a Control Plug-in that allows the creation of one Interface Instance after its authenticity and authorization has been validated.

Figure 5 shows the OGSI components that are used to implement the control system. The figure shows a farm with three server groups (tiers) of web servers, application servers, and a clustered database in the upper part. Each server group has a Control Plug-in associated that is connected via OGSI to its Interface Instance counterpart in the resource infrastructure.

One Control Plug-in is connected to at most one Interface Instance. An Interface Instance receives flex operations from its associated Control Plug-in and translates and delegates them into corresponding operations in the resource infrastructure, and finally into the UDC Controller. Monitoring and state change events are delivered in the reverse direction from an Interface Instance to its associated Control Plug-in.

6.4 Crossing Protection Domains

Since the UDC Controller is performing critical resource management functions in the Utility Data Center, it is located in a protected domain that cannot be reached from applications outside.

Since Control Plug-ins are located outside the UDC Controller in order to meet the customization and integration requirements, the interaction between Control Plug-ins and Interface Instances must cross the protection domain boundary in a secure way (in analogy to system calls in operating systems).

OGSI offers a built-in security model that is used for securely crossing protection domains between the Control Plug-in residing outside and the Interface Instance residing inside the protected UDC Controller domain.

OGSI Security is applied at two stages:

1. Control Plug-ins are authenticated. A Control Plug-in's certificate is validated by the Interface Factory at initial contact. Only Control Plug-ins with registered certificates can create Interface Instances and are able to subsequently interact with the UDC Controller.
2. Establishing an encrypted communication channel between the Control Plug-in and the Interface Instance ensuring for both sides that the exchanged information is authentic and unaltered.

6.5 Server Flex Control Cycle

Unassigned servers are maintained in UDC's resource pool. When the Control Plug-in initiates scaling a server group up or down, selected server resources transition through the stages shown in Figure 6.

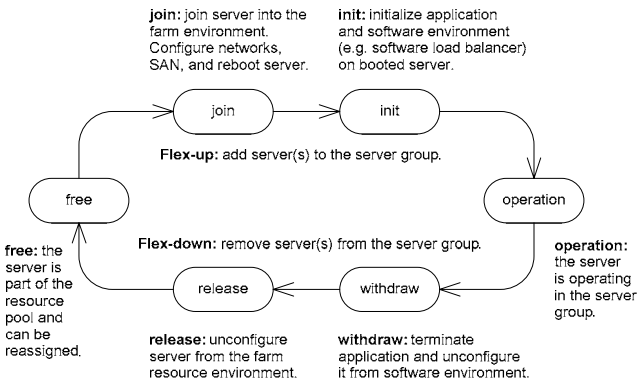

Figure 6: State diagram for server flex operations.

Flex up. – A server (or a number of servers) matching the type of the server group is selected from the resource pool and set into the join state. During that stage, the farm resource environment is reconfigured to include the new server(s). This means that the VLAN is reconfigured, virtual IP addresses are assigned to the server, DNS is reconfigured, and the logical disks from the storage array with copies of the server group images are created and mapped onto the joining servers. After that, the joining servers reboot and launch applications from those disks. The join state is finished when new server(s) have booted and applications are launched. Launching applications is initiated by start-up scripts configured on images.

The following init state allows executing further life cycle operations on joined servers (shown as ALCD – Application Life Cycle Daemon in the interaction diagram in Figure 7). This allows reconfiguration in the software environment, for instance, to start directing workload to joined servers. At the end of the init state, the new servers are fully operational as part of the server group.

Flex down. – The flex down cycle is also initiated by the Control Plug-in by selecting a number of servers to be released. In reverse order to flex-up, life cycle scripts on servers are executed first during withdraw, unconfiguring applications of affected servers from their environment. A load balancer may need to be reconfigured to not send further workload to affected servers, active sessions have to be finished, etc. At the end of the withdraw state, the local application instances are no longer part of the application and can finally be terminated.

During the release state, the servers to be released are unconfigured from the farm's resource environment including the VLAN. Disks (logical volumes) are detached by reprogramming the SAN and cleared. No state of application instances is kept for server groups.

At the end of the release state, all state associated with affected servers has been removed from the software and hardware environment including the servers themselves. Servers now transition to the free state and return to the server pool for new assignment.

6.6 Server Flex Protocol

The server flex protocol encompasses two parts. Monitoring and state change events must be delivered to the Control Plug-in and eventually other subscribers. And flex instructions (scale server group requests) must be delivered in opposite direction.

Event delivery. – Event delivery is based on OGSI events, which are based on changes in Service Data Elements (SDE) [1]. SDEs are maintained in Interface Instances. Two SDEs have been defined:

SDE SN – the current number of servers (SN – Server Number) in a server group. This SDE triggers events informing the associated Control Plug-in and other subscribers when either the farm administrator has reconfigured the server group through a console, or the server number has changed in effect of a previous flex operation. If the SN reported in the SDE equals to the target SN previously issued as flex request, the Control Plug-in concludes that the previous operation has succeeded. SN change events are trigged at state changes *init→operation* and *release→free* (Figure 6).

SDE LL – represents the current Load Level (LL) in a server group. This is an aggregate number taking various OpenView base measurements into account: CPU load, memory usage, IO and SWAP activity. Aggregation is based on a model that for any server *i* in a server group LL*i* = max(CPU, MEM, IO, SWAP), with each of the base measurements normalized to a scale of 0-100. The LL of the server group then is LL= Σ LL*i*. This simple model assumes that load between servers in the server group is evenly distributed.

The rate of LL SDE change events is controlled by OGSI's event flow rate control mechanism with *min_interval*=30sec and *max_interval*=300sec.

Based on SN and LL change events, which are also time stamped at the source, the Control Plug-in can obtain a load and utilization profile over time and assess changes in the server group that may lead to flex decisions. Policy parameters in the Control Plug-in must be tuned to avoid thrashing effects.

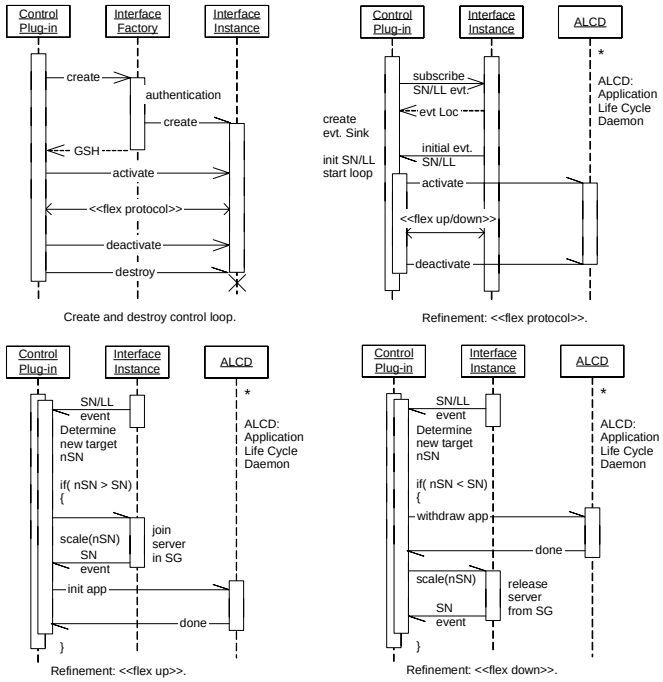

Figure 7: Server flex interaction diagrams.

Flex request. – Once the Control Plug-in has come to the conclusion to flex a server group, it issues a scale server request and waits for its completion indicated by a SN change event. Since execution of flex operations is in the range of 10+x minutes, flex requests are asynchronous. The Control Plug-in can observe the time within which it expects a flex operations to complete and take action when this time passes without receiving a server number change event.

Flex requests are issued with the total target number of servers rather then incremental changes in order to make flex requests idempotent. Failure or incompletion within the expected time frame simply may cause the Control Plug-in to repeat the request leading to the same result.

Figure 7 shows the server flex protocol including event delivery and server scale requests.

Figure 8 shows the effect of the server flex control system with the absolute server group load level decreasing over time from about 300 to 0. Accordingly, the Control Plug-

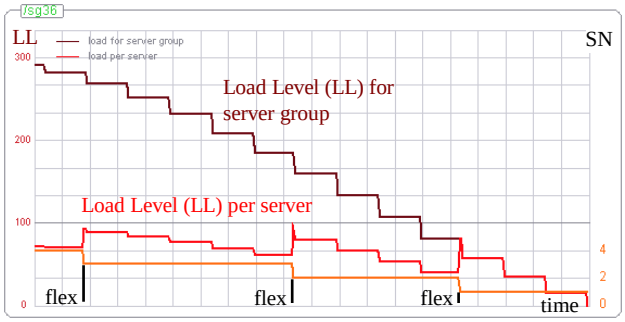

Figure 8: Effect of flexing on server load level (LL).

in reduces the number of servers in the server group from 4 to 1, leading to the flat server utilization curve shown in the diagram.

7 Summary

An adaptive control system for flexing server groups for horizontally-scalable applications in the Utility Data Center has been presented in this paper. The control system is based on Grid's open OGSI standard.

In contrast to similar control systems in clusters [9], [10], [11] the control system here operates in the heterogeneous environment of a commercial data center. Joining and releasing servers from its virtualized application environments (farms) is inherently more complex and requires securely crossing protection domains.

The capabilities of the Utility Data Center are expanded by a control interface to which external Control Plug-ins can connect for establishing control loops. The control interface is secured based on OGSI's security model.

Since OGSI is based on web services, HP's Web Services Management Framework (WSMF) [8] was used to manage OGSI components of the control system. WSMF has been proposed to OASIS/WSDM for standardization.

The adaptive control system has been built using Globus Toolkit 3 and is becoming part of the UDC product.

References

[1] Global Grid Forum: *Open Grid Services Infrastructure v1.0*, April 2003.

[2] Hewlett-Packard: *Utility Data Center*, http://www.hp.com/solutions1/infrastructure/solutions/utilitydata, 2003.

[3] Hewlett-Packard: *Enterprise Grid*, http://www.hp.com/techservers/grid/index.html, 2003.

[4] Sahai, A., Graupner, S., Machiraju, V., van Moorsel, A.: *Specifying and Monitoring Guarantees in Commercial Grids through SLA*, Proc. of CCGrid 2003, May 2003.

[5] Graupner, S., Pruyne, J., Singhal, S.: *Making the Utility Data Center A Power Station for the Enterprise Grid*, HP Labs Technical Report HPL-2003-53, http://www.hpl.hp.com/techreports/2003/, March 2003.

[6] Andrzejak, A., Kotov, V., Graupner, S., Trinks, H.: *Service-Centric Organization of Globally Distributed Computing*, IEEE Internet Computing, Special Issue on Grid Computing, July/August 2003.

[7] Graupner, S., Machiraju, V., Sahai, A., van Moorsel, A.: *Management += Grid*, DSOM'2003, October 2003.

[8] Hewlett-Packard: *Web Services Management Framework (WSMF) v2.0*, Specification, 2003.

[9] IBM: *xCAT, Extreme Cluster Administration Toolkit*, http://www.alphaworks.ibm.com/tech/xCAT, 2003.

[10] Ahmed, K., Priddy, K., Mashayekhi, V., Fang, Y.-C.: *The Cluster as Server*, Platform Computing, Inc., 2003.

[11] Sun Microsystems: Sun Grid Engine, http://wwws.sun.com/software/gridware, 2003.