DataSynapse

Virtualization for the Real-Time Enterprise

Centralized Command and Control of Enterprise Applications

FabricServer™ Product Overview

Table of Contents

1 Introduction

FabricServer™ virtualizes applications to run and scale across utility compute resources. While there are many virtualization technologies today that focus on system resources, FabricServer is unique because it provisions, activates, and manages applications on remote servers. These servers can be heterogeneous collections of legacy hardware or commodity-based utility data centers. By centralizing the command and control of application deployment and execution, FabricServer increases operational efficiency whilst reducing cost and complexity.

1.1 Application Virtualization

Application Virtualization frees applications from the confines of their fixed operating environments. Static host-specific configuration dependencies are removed from the application driving automation of software configuration, deployment, execution, monitoring, and management of individual applications, or large numbers of applications — all based on policy. Not only does FabricServer eliminate the need to install and configure applications on individual machines, it also enables the controlling a pool of compute resources in a highly scalable manner. Lastly, application virtualization requires the ability to dynamically expand or contract application capacity — based on schedule or demand. This last capability is often referred to as closed-loop provisioning — the environment can automatically respond to changing business needs. The figure below illustrates the three fundamental aspects to application virtualization.

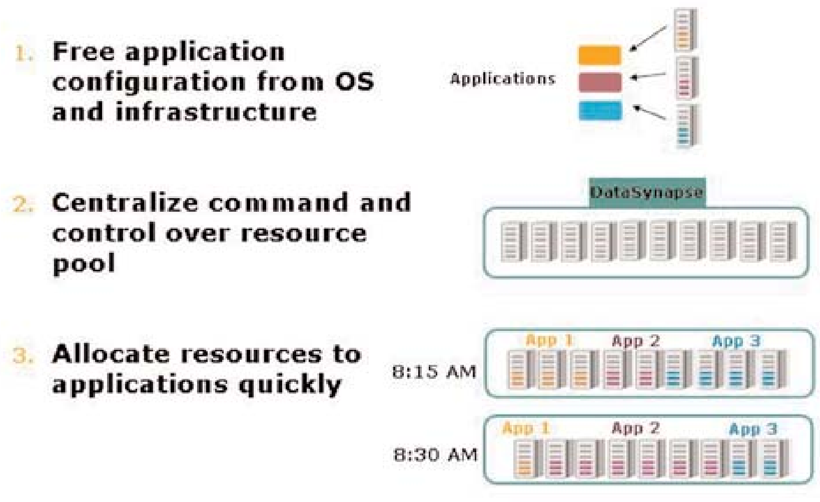

1. Free application configuration from OS and infrastructure
Applications
2. Centralize command and control over resource pool
DataSynapse
3. Allocate resources to applications quickly
8:15 AM
App 1
App 2
App 3
8:30 AM
App 1
App 2
App 3

Figure 1 - Three main concepts behind application virtualization

1.2 Uses and Benefits

Application virtualization is a transforming technology that opens up a number of use cases beyond just incremental automation improvement. FabricServer does automate manual tasks in a new radically more efficient paradigm, transforming the way application services are delivered to the business. The list below calls out some of the new capabilities that are made possible by using application virtualization:

Capacity-On-Demand

Application capacity can grow or shrink based on demand. Thresholds for activating new application instances can be made based on results from system or application level monitoring. Unused applications will automatically shrink down to their minimum required level of capacity, freeing up capacity for other applications where it is needed.

Standardization and Commoditization
Centralizing applications, application servers, and runtime distributions helps to bring order and discipline into an IT organization —a key driver to standardization. Migrating applications to commodity hardware is made much more attractive with application virtualization, because you can scale the management of these commodity resources. Server sprawl is made possible by low-cost hardware, but the management impact is large without application virtualization.

Business Continuity
Provisioning policies can be created and stored making it easy to activate entirely new application architectures across hundreds of disaster recovery (DR) machines with a single button click or Web service call. If there is a hardware failure, resilience is built into FabricServer, allowing new servers to be provisioned automatically.

Resource optimization and consolidation
By pooling resources and quickly allocating resources on demand, data center managers can dramatically reduce the amount of over-provisioning in the data center. Over-provisioning causes low utilization, and FabricServer can increase utilization and reduce the total number of servers required. In reducing servers, FabricServer is also reducing the application license and administration overhead associated with the application stack.

1.3 Utility Operations

FabricServer creates a utility computing operation by centralizing all aspects of application provisioning, activation, monitoring, and administration. Different people within an organization are typically tasked with these different activities. A utility operating model must not only provide the different roles and entitlements, but it must also have significant transparency and reporting for creating diagnostic methodologies and charge back models.

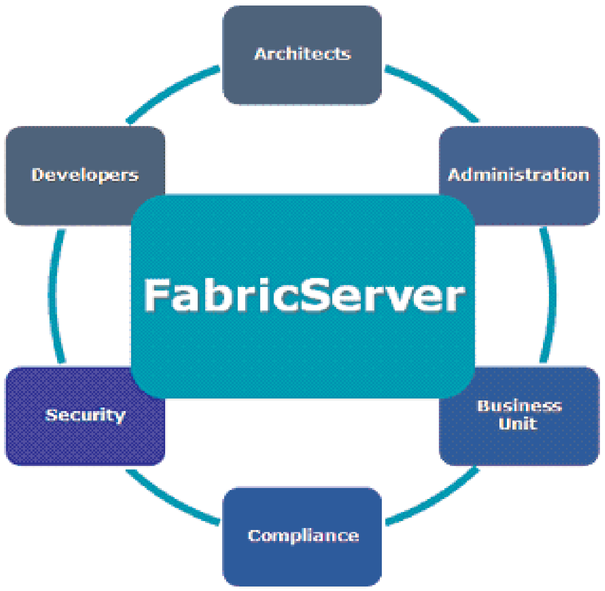


Figure 2 - Multiple users and roles require different entitlements

Configuration managers are involved with creating and standardizing distributions for runtimes, platforms, and application artifacts. Architects must understand the resource requirements and constraints for a given application and be able to create appropriate runtime policies. Administrators need to monitor applications and have transparency into the operations on remote machines. Therefore, uniform logging, auditing, and exception tracking are critical. Scheduling and coordination becomes an important capability and logistic, addressed by a utility computing model. Lastly, thorough reporting on usage is important for internal charge back and IT visibility.

2 FabricServer Version 2.0

FabricServer is a virtualization and provisioning platform that creates a highly adaptive utility computing infrastructure for applications. Applications can be running within standard application servers, middleware/integration infrastructure, web servers, independent software vendor applications, or standalone executables.

Applications and dependent runtimes are stored centrally, made assessable via a web-based repository. The central component is called the FabricBroker and serves a couple of purposes:

- Guarantees execution by deploying and activating applications on the machines in the compute pool.
- Allows operators to create, store, and modify specific policies for how and when the applications are activated on the utility resources.

2.1 System Overview

FabricBroker

FabricBroker is the central controller and coordinator for all activities related to applications. Application distributions, configuration, and policy are centrally stored. No manual interaction with compute nodes in the utility is required (other than the Broker). In addition, the application activation (start, stop, and clean-up) is based on schedule, rules, or directly by user. The FabricBroker captures statistical data for optimizing algorithms and providing historical reporting and dynamic provisioning.

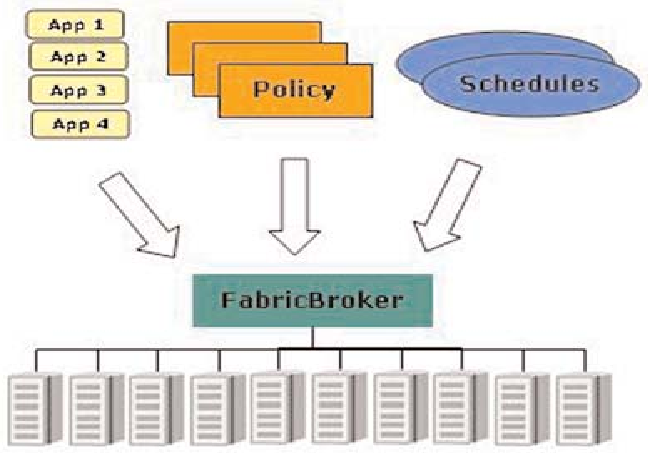


Figure 3 - Application and policy is centrally managed by FabricBroker

FabricServer Engine

FabricServer Engines are software processes (agents) running on the compute nodes in the utility. They are responsible for downloading the application files from their FabricBroker, controlling the application process lifecycle (start, stop, and clean-up), and collecting system and application monitoring statistics and sending the information to FabricBroker.

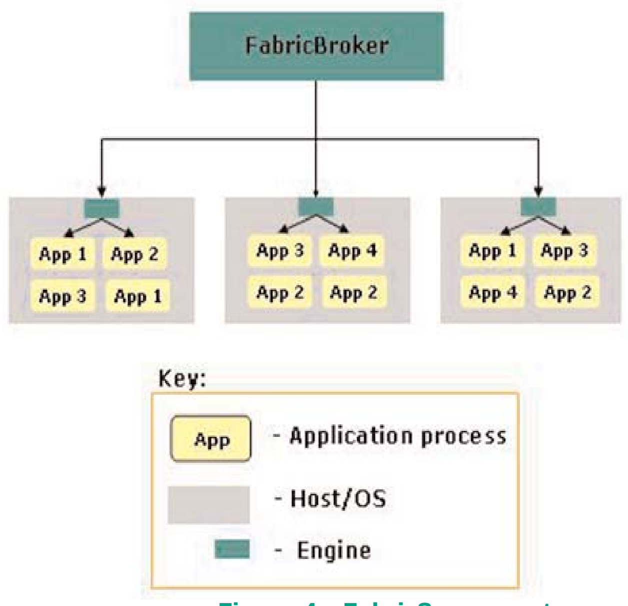


Figure 4 - FabricServer system components

FabricServer Web Admin
FabricServer has a rich, AJAX-enabled web user interface to support all functions around managing and operating the utility environment.

2.2 FabricServer Concepts

Application Domain
An Application Domain (or just "Domain" for short) is a collection of one or more running application processes that make up an application cluster or tier. If a Domain has two running instances, it means that there are two physical processes running (automatically activated on different machines if possible) that are load balanced by an external or internal load balancer. A Policy is set on a Domain to indicate how many processes should be running at a minimum (and maximum) at a given time. The FabricServer approach is a one-to-many configuration model. You only create one Domain image, but FabricServer dynamically changes the configuration files on the fly to allow the one image to be customized before launching on multiple machines. Things such as host names, ports, and work directories are changed in real-time by FabricServer Engines.

Distribution
A Distribution is a stored image (zip file) of the application server or platform that has been installed as a "golden image" (e.g. WebLogicServer 8.1 SP4 or Apache 2.0.). FabricServer uses this image as a baseline for provisioning the application platform. Necessary modifications to this are made in the container (described below).

Container
A Container includes wrapper code that starts and monitors the application and any configuration files in the distribution that need dynamic modification. Other media can also be included that overlays onto or overwrites the distribution (like third party applications or monitoring tools).

Policy
A Policy dictates how the system should be running at a specified interval of time. It is a set of application Domains with their corresponding minimum and maximum number of process instances, associated rules and constraints.

2.3 Functional Overview

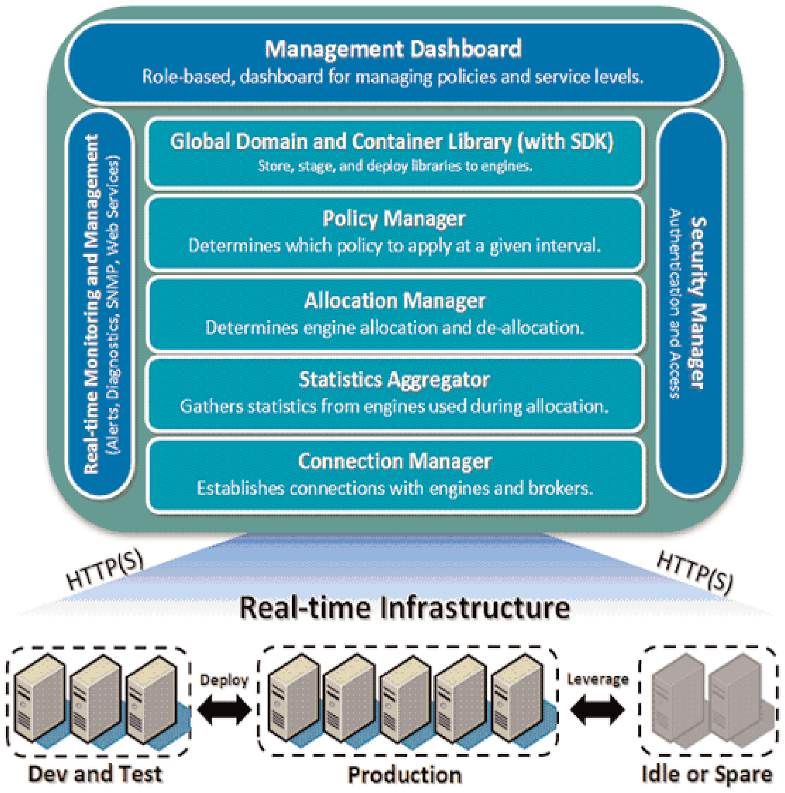

Management Dashboard
Role-based, dashboard for managing policies and service levels.
Global Domain and Container Library (with SDK)
Store, stage, and deploy libraries to engines.
Policy Manager
Determines which policy to apply at a given interval.
Allocation Manager
Determines engine allocation and de-allocation.
Statistics Aggregator
Gathers statistics from engines used during allocation.
Connection Manager
Establishes connections with engines and brokers.
Real-time Monitoring and Management
(Alerts, Diagnostics, SNMP, Web Services)
Security Manager
Authentication and Access
HTTP(S)
HTTP(S)
Real-time Infrastructure
Deploy
Leverage
Dev and Test
Production
Idle or Spare

Figure 5 - FabricServer functional components

Connection Managers
Connection Managers establish connections with Engines and other Brokers (for failover). The remote control of applications and statistics collection is made via the connection managers. The communication protocol is HTTP/HTTPS which allows for a highly scalable, secure, firewall-friendly management plane. Heartbeats flow between components via the connections in order to have robust operation with failover/failback capabilities.

Statistics Aggregator
The Statistics Aggregator gathers statistics from Engines for both historical reporting as well as dynamic provisioning of application resources. System statistics such as CPU utilization and memory footprint, are combined with application statistics such as threads, sessions, throughput, latency, etc. Together these statistics provide the overall demand for the application during a given time interval. Immediate history is kept in memory for efficient trend analysis and allocation decisions; a relational database is used for storing long term history and flexible reporting of usage and capacity requirements.

Allocation Manager
The Allocation Manager determines which applications run on which machines — it has a pluggable algorithm that allows for priority-based optimization of applications across machines. The algorithm is driven by the current policies in effect and the current running conditions (statistics captured). Rules in the policy govern constraints (OS, memory requirements, etc.) as well as threshold data for adding more capability based on statistics. Blacklisting also ensures that applications that cannot be started on certain machines (for whatever reason) are not continually launched unsuccessfully, running successfully elsewhere.

Policy Manager
The Policy Manager allows users to create, modify, deploy, and activate application policy. Multiple Policies can be created that reflect running the applications on different resources or with different cluster sizes, etc. One Policy might be for a group of applications for disaster recovery (DR) or for Universal Acceptance Testing (UAT), while another set would be for production.

Domain and Container Repository
The Domain and Container Repository is the central location and management User Interface (UI) that stores all the libraries used for applications — including the distribution and runtimes (Java, Perl, etc.). The UI allows for staging and deploying these archives in controlled and roles-based fashion.

Reporting Database
The Reporting Database is used for storage of system and application statistics, alerts, user events, and other tracked data. The standard RDBMS vendors are supported and a simple web UI can be used to get simple canned reports or create and store custom queries.

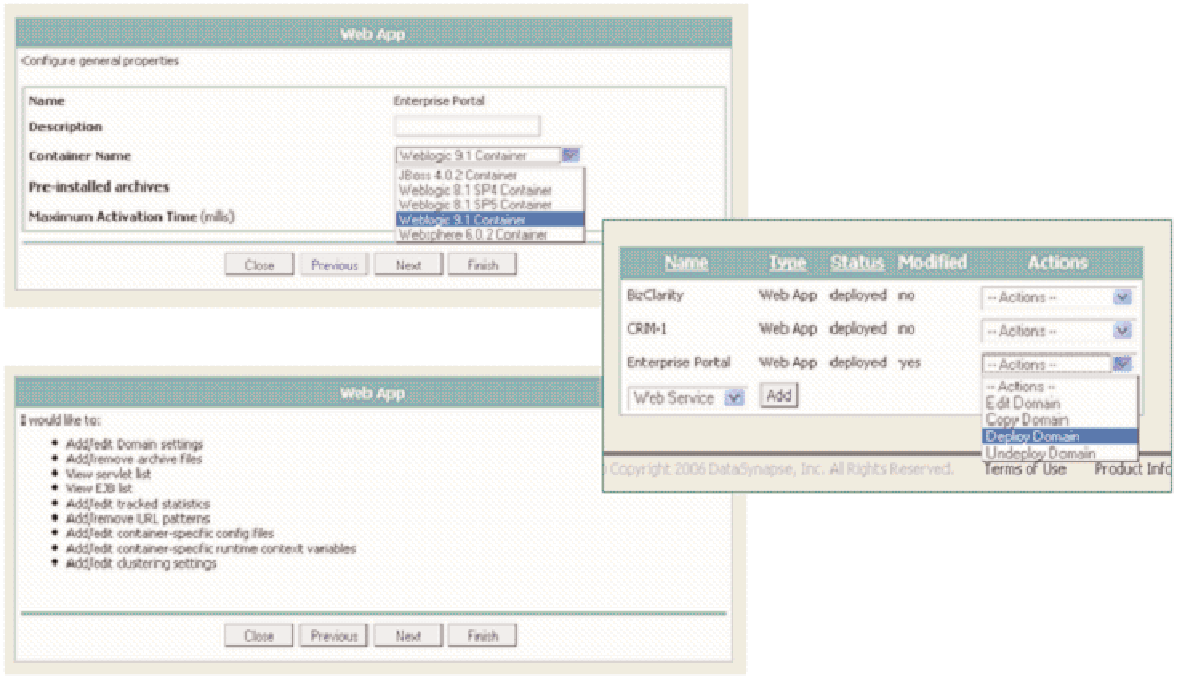
Monitoring and Management Dashboard
The FabricServer web console has a dashboard section with dynamic updates of quick view operational data — applications, resources, statistics, etc. Events are posted to the UI as well as to SNMP traps for capture by external tools and systems.

Web Services and SDK
Most activities that can be performed with the web console can be done via Web services (SOAP/HTTP), enabling other applications or management systems to control FabricServer. Events can be used to trigger scheduling or Policy changes within FabricServer.

3 Using FabricServer

3.1 Creating and Deploying Application Domains

FabricServer simplifies application management by storing all the information pertaining to the application servers, versions, customizations, and application code in one centralized location. There is no need to consider multiple configuration files for different computers or cluster members as FabricServer supports a one-to-many paradigm. Below are screen shots that demonstrate the creation and management of J2EE applications — including the selection of supported containers and runtimes.



Web App

Configure general properties

Name	Enterprise Portal
Description	
Container Name	Weblogic 9.1 Container
Pre-installed archives	
Maximum Activation Time (mills)	

JBoss 4.0.2 Container
Weblogic 8.1 SP4 Container
Weblogic 8.1 SP5 Container
Weblogic 9.1 Container
Websphere 6.0.2 Container
Close
Previous
Next
Finish

Web App

I would like to:

- Add/edit Domain settings
- Add/remove archive files
- View servlet list
- View EJB list
- Add/edit tracked statistics
- Add/remove URL patterns
- Add/edit container-specific config files
- Add/edit container-specific runtime context variables
- Add/edit clustering settings

Name	Type	Status	Modified	Actions
BizClarity	Web App	deployed	no	-- Actions --
CRM-1	Web App	deployed	no	-- Actions --
Enterprise Portal	Web App	deployed	yes	-- Actions --

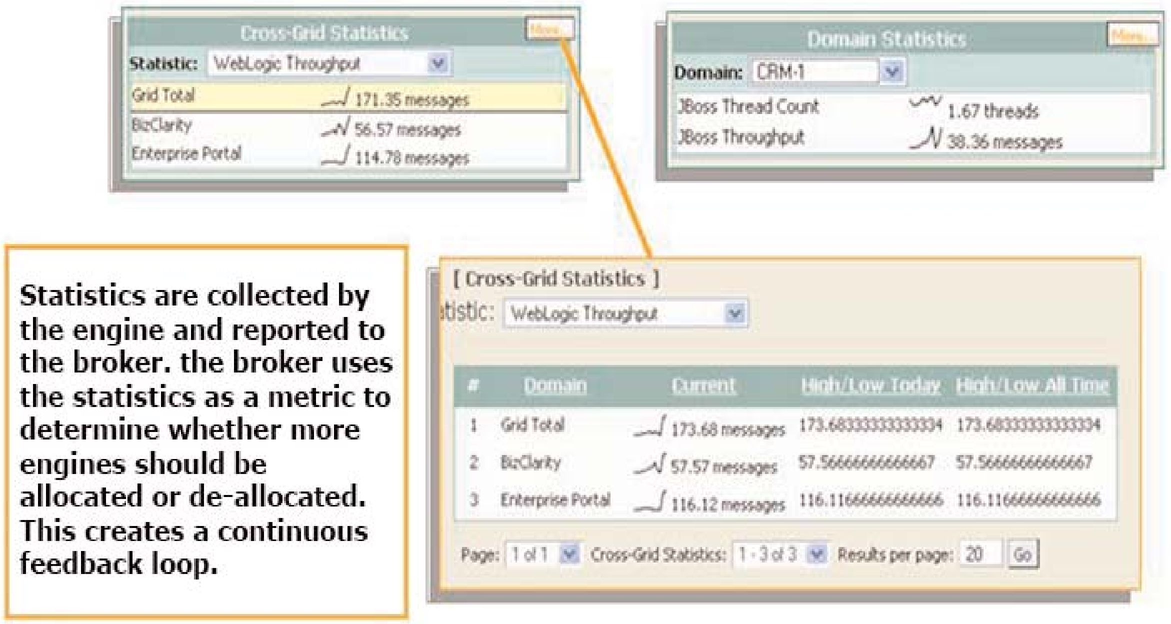
Copyright 2006 DataSynapse, Inc. All Rights Reserved.

Figure 6a — Creating and deploying applications is made simple

Once an application is created and configured, it can be deployed out to the utility with a single mouse click. In addition, it's easy to modify and re-deploy applications on demand making for highly scalable application management.

3.2 Statistics Gathering

FabricServer monitors key statistics to use for IT intelligence and dynamically adjust the allocation of resources to applications. When using Java applications, JMX can be used to capture any relevant MBean attributes. Measuring the right JMX attributes to determine load and system performance is under user control as it is critical for enforcing and meeting SLAs. For example, the load on a Java application server — and the commensurate user response time — is not usually best assessed through CPU utilization. Instead, employing JMX attributes such as queue depth, service throughput, or thread count are better alternatives.



Cross-Grid Statistics

Statistic: WebLogic Throughput

Grid Total	171.35 messages
BizClarity	56.57 messages
Enterprise Portal	114.78 messages

Domain Statistics

Domain: CRM-1

JBoss Thread Count	1.67 threads
JBoss Throughput	38.36 messages

Statistics are collected by the engine and reported to the broker. the broker uses the statistics as a metric to determine whether more engines should be allocated or de-allocated. This creates a continuous feedback loop.

[Cross-Grid Statistics]

Statistic: WebLogic Throughput

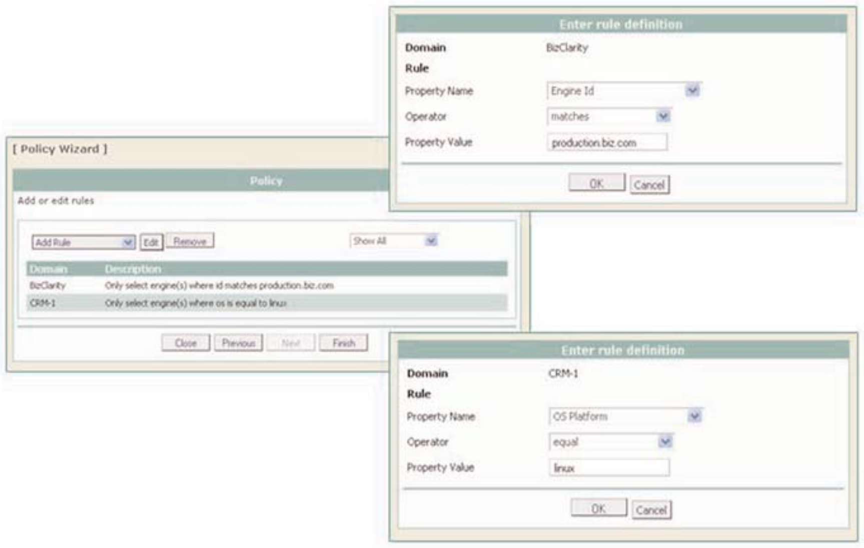
#	Domain	Current	High/Low Today	High/Low All Time
1	Grid Total	173.68 messages	173.68333333333334	173.68333333333334
2	BizClarity	57.57 messages	57.56666666666667	57.56666666666667
3	Enterprise Portal	116.12 messages	116.11666666666666	116.11666666666666

Page: 1 of 1 Cross-Grid Statistics: 1 - 3 of 3 Results per page: 20

Figure 6b - Select application and system statistics are captured and displayed

3.3 Creating and Managing Policy

Policy Wizards allow for the simple creation and management of application policy. Applications added to the policy must be assigned a minimum and maximum number of application server instances. Applications are also given priority weights, which communicate to FabricServer the proportion of server instances to be split — if there is contention for resources. For example, if there are not enough resources to meet the minimum or desired level for level server instances, the policy will be used to allocate scarce resources to the identified priority applications. Once the minimum and maximum numbers of application instances are set, FabricServer will provision servers to the minimum level, and it will choose to allocate more resources to that Domain if there is a rule that can be used to track the load on the application. For example, WebLogic Queue Size can be used to understand the current load on the cluster. If the average queue size (sampled over time, averaged over all server instances) increases above a certain level, then another server will be started automatically. Likewise, if this average quantity falls below a certain level, a resource will relinquished to the pool.



Enter rule definition

Domain	BizClarity
Rule	
Property Name	Engine Id
Operator	matches
Property Value	production.biz.com

[Policy Wizard]

Policy

Add or edit rules

Domain	Description
BizClarity	Only select engine(s) where id matches production.biz.com
CRM-1	Only select engine(s) where os is equal to linux

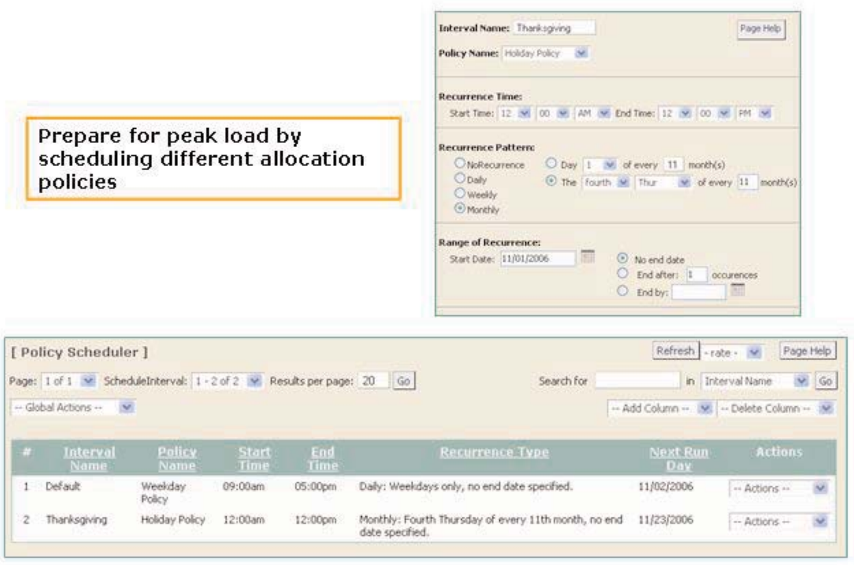
Enter rule definition

Domain	CRM-1
Rule	
Property Name	OS Platform
Operator	equal
Property Value	linux

Figure 7 - UI for creating and managing application policy

3.4 Scheduling

FabricServer will schedule policies to be in effect over specified time intervals. Much like MS Outlook event scheduling, FabricServer allows for re-occurring scenarios running appropriate application policy reliably and automatically. The resulting schedule can be viewed and validation processes will alert the user if the schedules are conflicting and allow for precedence assertion.



Prepare for peak load by scheduling different allocation policies

Interval Name: Thanksgiving
Policy Name: Holiday Policy

Recurrence Time:
Start Time: 12 : 00 AM End Time: 12 : 00 PM

Recurrence Pattern:
( ) NoRecurrence ( ) Daily ( ) Weekly [✓ Monthly]
( ) Day 1 of every 11 month(s)
[✓ The] fourth Thur of every 11 month(s)

Range of Recurrence:
Start Date: 11/01/2006
[✓ No end date] ( ) End after: 1 occurences ( ) End by:

[Policy Scheduler]

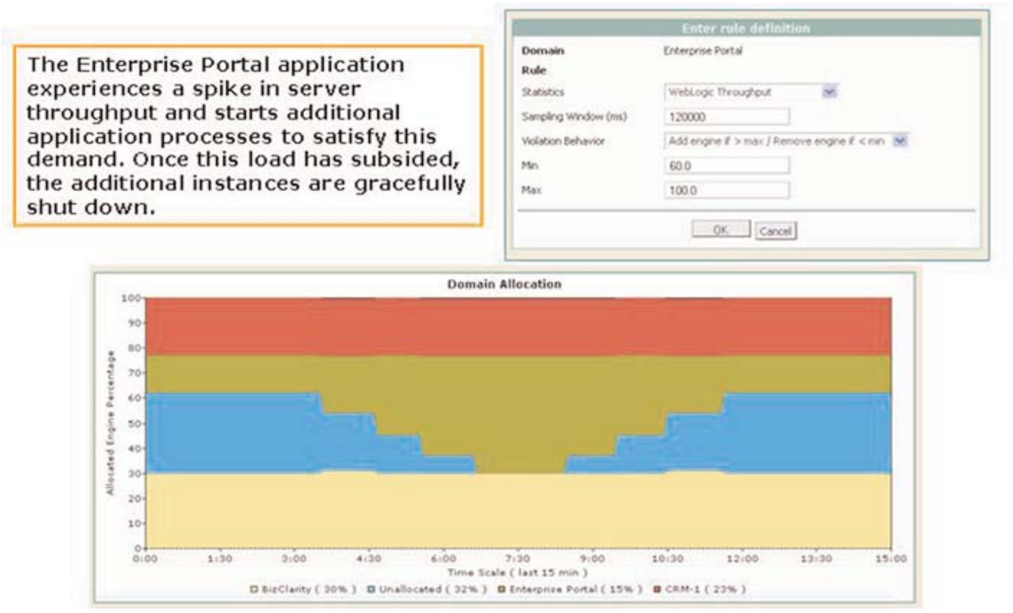
Page: 1 of 1 ScheduleInterval: 1 - 2 of 2 Results per page: 20
Search for in Interval Name

#	Interval Name	Policy Name	Start Time	End Time	Recurrence Type	Next Run Day	Actions
1	Default	Weekday Policy	09:00am	05:00pm	Daily: Weekdays only, no end date specified.	11/02/2006	-- Actions --
2	Thanksgiving	Holiday Policy	12:00am	12:00pm	Monthly: Fourth Thursday of every 11th month, no end date specified.	11/23/2006	-- Actions --

Figure 8 - Flexible application policy scheduling

3.5 Real-Time Provisioning

Current application environments are statically sized and managed in silos. This usually means that a business critical application will be allotted a fixed amount of resources in order to meet peak demand. With FabricServer, applications can be resized dynamically, assuring high availability (HA), SLA fulfillment and efficient resource utilization. Applications are no longer fixed to a specific host; instead, any application can run on any resource, at any time.



The Enterprise Portal application experiences a spike in server throughput and starts additional application processes to satisfy this demand. Once this load has subsided, the additional instances are gracefully shut down.

Enter rule definition

Domain	Enterprise Portal
Rule	
Statistics	WebLogic Throughput
Sampling Window (ms)	120000
Violation Behavior	Add engine if > max / Remove engine if < min
Min	60.0
Max	100.0

Figure 9 - Dynamic allocation of resources based on demand

3.6 Virtual Router

Existing load balancers where in place will be used in many circumstances with FabricServer. By default, dynamic load balancing capabilities are also included within FabricBroker and packaged separately as a JEE web application. The built-in load balancing is extremely useful for test and UAT environments, as well as new application deployments that don't have load balancing strategies already place. The FabricServer Virtual Router is constantly updated with the current application cluster configurations, so that no load balancer configuration is required. FabricServer supports multiple Virtual Routers for redundancy and load sharing and continues to run in the event of failover.

4 Integration and Interoperability

4.1 Load Balancers

FabricServer has an Event API that allows custom code to receive Container activation events and notify downstream systems. This plug-in allows for the simple integration with both hardware and software load balancers. Whether the plug-in calls hardware balancers via SOAP or continually updates software balancing files, this capability is flexible and easy to implement.

4.2 GridServer®

Many components, principles, and concepts that are provided by GridServer are also present in FabricServer. Don't know what GridServer is? Visit our website, www.datasynapse.com, for more information.

The following list covers the broadest overlapping capabilities:

1. *Application Deployment*
   Grid Libraries are a cross-platform, application component packaging format, which makes it possible to deploy cross-platform, cross-language, versioned binaries and environment settings. Both FabricServer and GridServer have an updating facility component for rolling out new code to distributed resources. This enterprise feature allows for many applications and many versions of the same application to run in the same distributed environment simultaneously.

2. *Agent Technology*
   Background processes that monitor resources and activate and control dependent processes that host service instances are called Agents. GridServer and FabricServer have the same Engine Daemons that act as agents.

3. *Web Management*
   GridServer and FabricServer have the same integrated framework for exposing management functionality via web (HMTL), API or SOAP access mechanisms. The Ajax-based dashboards are also the same infrastructure and cross-product dashboards will be present in future versions of the product.

4. *Communications Infrastructure*
   Both FabricServer and GridServer are built on the same highly scalable management plane. The micro-kernel plug-in architecture makes it possible to add new capabilities in a loosely coupled fashion.

Since FabricServer and GridServer utilize the same agent and Engine, they are completely interoperable. Engines reporting to GridServer can be redirected to FabricServer (and vice versa), where they will automatically reconfigure into FabricServer Engines.

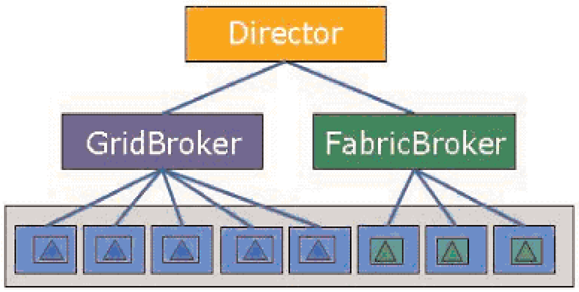


Figure 10 - FabricServer and GridServer Engines use the agent bootstrapping technology and can be utility across products.

4.3 Other Virtualization Technologies

FabricServer interoperates with other virtualization, provisioning, and monitoring systems in three ways:

1. When FabricServer needs more servers in its shared compute pool, it can notify external systems of this need. This case may arise if a certain application has very specific resource requirements that aren't currently available in the pool of resources managed by FabricServer. It can also arise if there is a peak capacity requirement across many applications at the same time and the current pool of resources is not adequate to fully service the load.

2. FabricServer's Policy and Scheduling Engine can activate other systems to dynamically create or provision server instances. In this case, FabricServer becomes a Policy Engine for not only the applications on FabricServer, but for other provisioning and virtualization systems, controlling server images outside of FabricServer.

3. FabricServer's event mechanism can be used to notify monitoring and management systems of activation events, exceptions, etc. via SNMP or Web services calls.

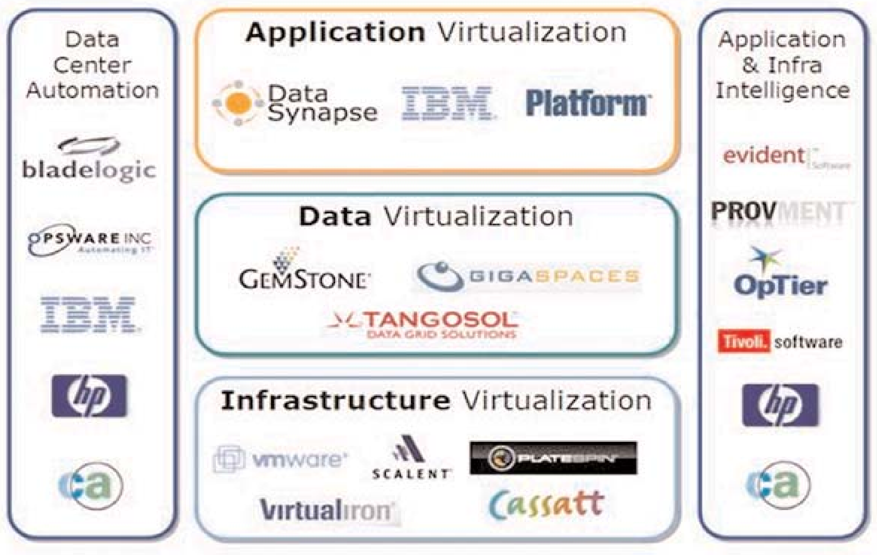


Figure 11 - FabricServer can interoperate with all layers of the stack

About DataSynapse

DataSynapse, Inc. is a global provider of application virtualization software. The company's flagship products, GridServer and FabricServer™, virtualize business-critical applications and adaptively provision them across a real-time infrastructure. DataSynapse drives business agility through shared services, helping clients reduce the cost and complexity of their IT infrastructure. The company is headquartered in New York City.