Deployment Considerations for Data Center Management Tools

By Edward Wustenhoff –Sun Professional Services

Sun BluePrints™ OnLine - May 2002

http://www.sun.com/blueprints

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95045 USA
650 960-1300

Part No.: 816-4939-10
Revision A May 2002

Please
Recycle

Deployment Considerations for Data Center Management Tools

This article describes some of the main aspects to consider when deploying a data center management tools infrastructure (DCMTI). It also includes considerations to keep in mind when complementing this environment with a process management tool to facilitate the integration with other external processes such as, but not limited to, a help desk function.

This article is a prelude to a follow-on article that will describe an actual implementation of such a management architecture in one of Sun's iForce[SM] Ready Center programs.

The topics in this article are:

- *Main Considerations*
- *Architecture*
- *Other Considerations*

The main considerations when designing and implementing a DCMTI are:

- Create visibility at all layers for all aspects.
  "FCAPS" on page 3 describes these aspects (fault, configuration, accounting, performance and security)
- Create a process management environment to facilitate interaction with other organizations and service request control.

Considering these aspects results in a management architecture that has five major components:

- Agents
- Management servers and consoles
- Correlation and framework server
- Consoles
- Process management tool

The physical distribution within the management architecture can vary based on specific requirements. However, the natural separation points are:

- Between the agents and the server
- Between the management servers and the Framework server
- Between the Framework server and the process server

We recommend a separate management network for performance, visibility and security reasons.

After reading this document, you should have a good understanding of some of the main aspects to consider when building a DCMTI, and you will be ready to begin deploying a DCMTI. A follow-on article will describe the details of a deployment that incorporates the suggestions in this article.

Main Considerations

A good DCMTI provides the information to support several different views into the managed environment. These views are often organized by layer—facilities, network, compute and storage, and application infrastructure—Lightweight Directory Access Protocol (LDAP), domain name service (DNS), relational database management system (RDBMS), Network Time Protocol (NTP) and so forth, and at the top, the business application.

In addition to these views, there should also be a Service Level Management (SLM) view. The main objective of this view is to show how the service provided measures against a predefined Service Level Agreement (SLA) and its associated Service Level Objectives (SLOs). The articles *Service Level Management in the Data Center* and *Building a Service Level Agreement in the Data Center* describe the main concepts of SLAs and SLOs, so no additional details are included herein.

The views by layer must provide information of all aspects that are deemed important by the operations staff to keep the systems up and running. The International Standards Organization (ISO) has defined five areas (FCAPS) that completely address this requirement.

FCAPS

The FCAPS aspects are:

- *Fault*
- *Configuration*
- *Accounting*
- *Performance*
- *Security*

Fault

This aspect looks at the status of the components and whether they are performing within set thresholds. It is event based. Broken disks and dead processes are examples of events.

Configuration

This aspect manages the configuration of the IT components. It tracks the parameters and values of the IT components. Preferably a history of configurations is maintained so a bad change is backed-out easily.

Accounting

This aspect is an older concept that stems from the mainframe world. It is the ability to track usage of system resources and relate that to business units and/or customers to enable billing. An interesting side note is that, with the emerging ASP business models, accounting has received renewed interest.

Performance

This aspect manages the challenging task of monitoring how fast or slow a system responds and processes transactions. A key process in this area is performance tuning and capacity planning, where historical data is submitted for analysis to discover trends or model anticipated changes in the environment.

Security

This aspect manages the complete infrastructure from an authentication, authorization and access perspective. Security is very pervasive and should be addressed early in the architecture design and deployment phases.

As mentioned earlier, all of these aspects should be managed at all layers in the infrastructure. TABLE 1 shows that concept. An advantage of this representation is that it enables a quick overview to assess and identify areas that are candidates to be addressed by the management infrastructure.

TABLE 1 FCAPS Overview

	Fault	Configuration	Accounting	Performance	Security
Business application	5	2	2	3	2
Application infrastructure (RDBMS, LDAP and so forth)	5	2	1	1	1
Compute and storage platform	5	3	1	3	2
Network	5	2	1	3	3
Facilities	5	2	2	1	3

The numbers in this example, indicate a level of compliance. Five means, “well covered” and zero means, “not covered.” The same table can be used to describe the requirements for a DCMTI. In that case, five could mean, “important requirement” and zero could mean “no requirement”.

Interaction With Other Organizations

In addition to the views that represent the appropriate aspects organized by layer, a process management tool is a very important consideration.

A process management tool facilitates the transition of activities into other processes, and it facilitates the following main aspects:

Service Request Control

- Status update (new, latest event and so on)
- Progress enforcement (escalation, if needed)

- Qualification and routing (where next?)
- Closure (quality control surveys and so on)

Reporting

- Periodic reports
  - Management
  - Service Performance
- Exception reports

These functions are often provided by a *help desk* or *customer care desk*. However, in context of this document, the management infrastructure is assumed to be capable of generating requests based on predefined rules. The rules to determine when to create a request are implemented and enforced at the alert consolidation and correlation layer in the management infrastructure. “Architecture” on page 8 details this process.

Service Request Process

FIGURE 1 is a high-level process view of how the process management tool would handle a ticket. The intent is to highlight key steps that you must consider when building such a process and mapping it to the tools ticket.

It is important to realize that there are multiple sources for action requests in the IT management environment. Four sources are given here as an example; other sources exist, depending on specific situations. Before the request enters the process it should be prioritized, localized (in case of multiple locations of activities) and categorized. Based on that information it will be qualified and assigned.

Typically, this should be a generic name or group (not a person’s name) to avoid constant updating of the configuration files that link this information. Depending on priority, location and category, the ticket starts to follow a distinct process that tracks progress and key information for *service performance* and *management reporting* purposes.

Essential considerations for prioritizing and routing a request are:

- P—Priority
- S—Skills needed that determine the routing
- A—Action(s) to represent a distinct process

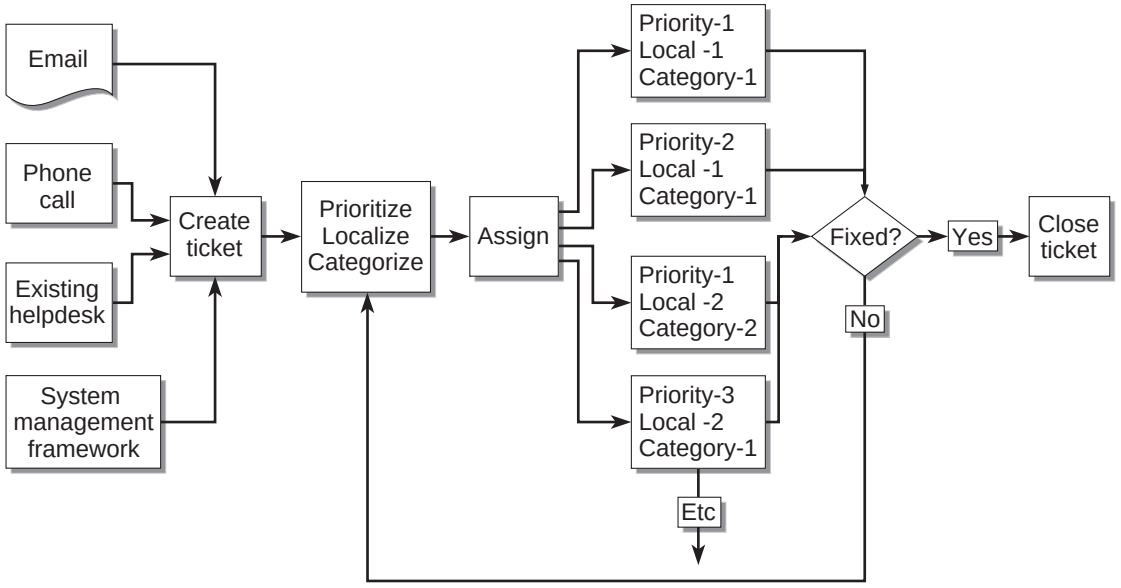

FIGURE 1 Sample Request Process View of Ticket Handling

TABLE 2 Service and Management Reporting

Function	Driver	Examples
Priority ->	**Cost of downtime**	
	No. of users affected	P1–More than 10 users affected and / or business critical system is down during production hours
	System function (business critical)	P2–Less than 10 users affected and / or not a business critical system during any time of the day
	Time of day	P3–Request for enhancement. Not business critical. No time pressure.
	Service Level Agreement	P4–Specific rules as per the agreement
		... and so on.
Routing ->	**Skills needed**	
	What type pf technology?	S1–Computer Sun hardware disk fault **P1**
	What type pf alert (FCAPS)?	S2–Computer IBM operating kernel performance system **P2**

TABLE 2 Service and Management Reporting *(Continued)*

Function	Driver	Examples	
	What priority?	S3–Network Cisco hardware router configuration	**P3**
Process ->	**Action needed**		
	Resolution time	A1–Must be resolved ASAP	**S3 P1**
	Skills needed	A2–Should be resolved within 4 hours	**S1 P2**
	Priority of request	A3–Should be resolved within 2 hours	**S2 P4**

When all three functions have been defined, you can create a matrix that relates the priorities of a request, based on the skills needed to the appropriate process. This typically identifies which group is assigned. Based on the preceding table, TABLE 3 shows this priority request matrix.

TABLE 3 Priority Request Matrix

	P1	P2	P3	P4
S1	A1	A1	A6	A10
S2	A2	A4	A7	A10
S3	A3	A5	A8	A10
S4	A3	A5	A9	A10

It is important to realize that *service request priorities do not influence the priorities or criticality at the system agent layer.* The health of a system is independent of its impact on the business. The former is addressed in the DCMTI, the latter in process management.

Each specific process has a rule to allow for escalation and re-assignment. When all goes well, the request is fulfilled and the ticket is closed. The closing process can include activities like informing users, updating databases, and sometimes even initiating clearing of alarms in the DCMTI.

FIGURE 2 shows some key aspects to consider in the specialized resolution process of a trouble ticket. It illustrates the preceding considerations with more detail.

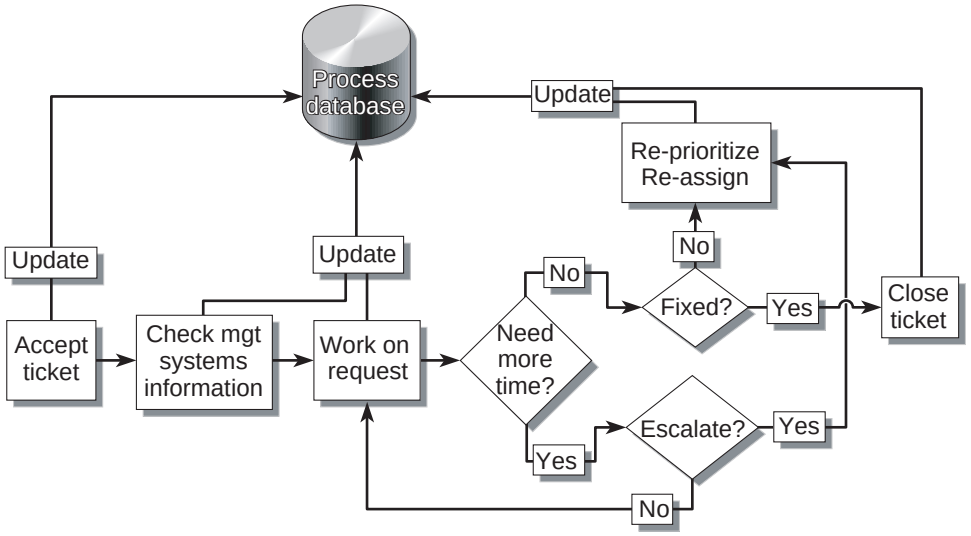

FIGURE 2 Sample Trouble Ticket Resolution Process

Most notable is the update of the *process database* at key steps in the process. Also, in the decision tree towards the end, there is an interesting example of how escalation can be achieved. Generally, an automated approach to escalation is not recommended because it would automatically reassign a ticket. The most common approach is to run daily reports or create alerts for supervisors who make the best decision for the next step, and generate ad-hoc reports (email, text page and so on) for high priority events that require immediate attention.

Architecture

Having described the main considerations to include in a DCMTI, the following diagram shows the layout of the management architecture; the following sections describe each component in detail.

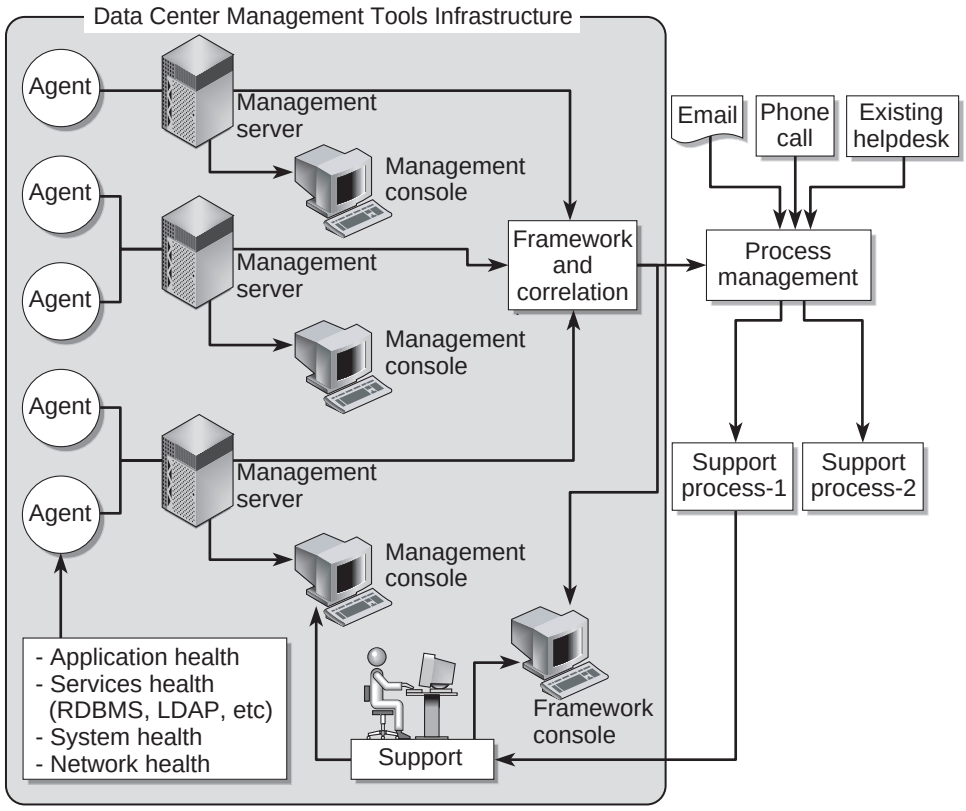

FIGURE 3 Management Architecture Layout

A key point this drawing makes is the separation of process from the DCMTI. The agents collect only data to determine system status and health at all layers (facilities, network, compute and storage, application infrastructure (LDAP, RDBMS, NTP, DNS and so forth) and the business application for all aspects (FCAPS). The management servers and framework/correlation server provide filtering and automatic resolution. Only when human interaction is required (passive or active) is the information forwarded to the process management system.

Agents

The agents collect the matrix to support the views. They focus mainly on health and status. The thresholds set here are only as they relate to the system monitored. They do not include the decisions about business severity, or whether or not they conform to an SLA. That is done in the process management layer.

For example, a disk fails and, as a result, triggers a severe alarm because it should be fixed as soon as possible. However, the trouble ticket (in the process management tool) might have a medium priority because the failed disk was mirrored and no service interruption occurred. The agent triggers the first alert; the process management tool sets the ticket priority.

Management Servers

These are the specialized tools that accept the alerts from specific agents. A Sun™ Management Center software agent talks to the Sun Management Center software management server and determines whether the error can be fixed or should be forwarded to the framework server. A BMC Knowledge Module (KM) will do the same when monitoring the Oracle database.

These are the main tools for the respective support specialists.

Correlation and Framework Server

All relevant alerts are forwarded to this server by the management servers to allow correlation and a single view into the health of the infrastructure. Here the decision is made to automatically create a ticket and start the appropriate resolution process.

The correlation server can also be a management server for its own specific purpose. For example, Tivoli has some configuration management agents that can report to the same Tivoli server that acts as the framework server.

Consoles

Almost all management servers require a proprietary console to allow management of the management server. Consoles can be started on different systems to provide specific views into the infrastructure. They enable the support specialists to “drill-down” to find more information regarding an open ticket.

Consoles are often easily distributed and more than one console can often connect to the same server to provide multiple views for different purposes.

Process Management Server

This is the server that takes the information and alerts from the DCMTI and makes decisions about priority, routing, and which process to use. It is often a Helpdesk-oriented tool like Remedy or ClearCase that has features to meet the previous described requirements.

The main objective for a process management tool is to streamline the processes and assure a closed loop to keep requests from “falling into the cracks.”

With these components in place almost all IT management processes can be supported by providing timely, integrated and consolidated information, streamlining process steps, and creating good visibility in the main aspects of the managed IT environment.

Other Considerations

This section covers the following topics:

- *Distribution of Components*
- *Management Network*
  - *Performance*
  - *Visibility*
  - *Security*

Distribution of Components

During the deployment of a DCMTI you must decide how to distribute the previously described components. Most often this becomes a discussion about control and ownership, which is beyond the scope of this document. However, there are three natural separation points.

1. **Between the agents and the server.** This separation point should be considered when there are few servers to be managed and the network between the management servers and the agents is robust and not very expensive. Most often this can be done in a campus environment with a high-speed backbone.
2. **Between the management servers and the framework server.** This approach is practical when certain expertise or management capability is only necessary at a local level.

3. **Between the framework server and the process server.** This approach is preferred when a high level of autonomy is required. This requirement typically happens when sites are geographically dispersed. An additional central framework server might be considered to create a *world view* of the IT environment.

Another way to redistribute the components in the DCMTI is by collapsing components. This can be done by combining more than one management server into one physical computer system or by including the process management tool in the framework server.

The main considerations in this case are performance and manageability. The latter becomes more critical as one computer becomes a lot more complex due to the need to combine two different servers that typically assume sole control over their resources. However, in smaller deployments the cost of hardware sometimes outweighs the challenges of complexity.

Management Network

Building a separate management network (also referred to as an out-of-band network) to support the aforementioned management infrastructure is recommended highly. The following sections describe the main reasons for this recommendation.

Performance

Depending on the implementation, the management network traffic can be significant. Without a separate management network, all traffic from the management servers and clients, and from production activities compete for network bandwidth. This situation can create a problem. When there is a busy network connection, the management traffic cannot reach the management servers and alerts can not reach the process management server.

Visibility

An extended version of the preceding scenario is that the network may fail, in which case there is no route to the management server and the failed network alert might not reach the management servers. Even when redundant networks are in place, you must still consider the possibility that a severe outage results in management information loss.

Security

Although good progress has been made in Simple Network Management Protocol (SNMP) v2 and v3, SNMP v1 is inherently insecure. By having it share the same network as the production systems, these systems can be more vulnerable to security violations. You can achieve higher levels of security by separating the traffic.

Author's Biography

EDWARD WUSTENHOFF
Chief IT Consultant
Sun Professional Services

Edward has a total of 16 years experience in networked computer systems and data center management, including the latest internet technologies. The past seven years were at Sun where he became familiar with most Sun products and technologies.

Edward is currently a Chief IT Consultant in Sun Professional Services of the America's at Sun Microsystems, Inc. In one of his previous roles at Sun, he managed the Enterprise Management Practice where he advised Sun's customers about best practices, tools selection, and deployment strategies.

Previous projects included architecture and design of networks and computer systems, in addition to assessing customer environments and suggesting improvements.

Currently, Edward is very active in the IDC management space.