


IT service management for high availability



R. Radhakrishnan
K. Mark
B. Powell

High-availability service management (HASM) is defined as information technology (IT) service management that is designed to meet the business demand for availability of critical IT and IT-enabled business services. HASM requires the use of the Six Sigma method and analytical tools applied to key service management processes and services; event and incident monitoring and management design; high-end and high-quality infrastructure and application configuration; high-availability (HA) architecture and design; and special solutions that implement HA patterns and associated technologies. In this paper, we examine HASM and discuss the process flow for designing and implementing HA technologies.

INTRODUCTION

High-availability (HA) infrastructure, applications, and service solutions are designed to achieve a specified level of availability. HA infrastructure and applications use HA architecture and design patterns, technologies, and standards such as clustering, fault tolerance, and fast recovery systems to achieve this goal. HA service solutions use techniques such as fault tree analysis (FTA),¹ component failure impact analysis (CFIA),¹ and HA configuration audits to reduce service outages and the impact of events and incidents on the delivery of agreed-upon service availability levels.

In this paper, we examine high-availability service management (HASM), which is IT service management that is designed to meet the business demand for availability of critical IT (information technology) and IT-enabled business services. HASM requires significant IT service management maturity and the use of the Six Sigma methods (such as

critical to quality [CTQ] characteristics and Six Sigma process capability) and analytical tools (such as failure mode effect analysis [FMEA]) applied to key service management processes and services. Six Sigma methods include those related to defining, measuring, analyzing, improving, and controlling IT processes. Six Sigma methods can also be applied to IT product quality.

HASM also requires the use of event and incident monitoring and management design, high-end and high-quality infrastructure and application configuration, HA architecture and design, and special solutions implementing HA patterns and associated technologies.

©Copyright 2008 by International Business Machines Corporation. Copying in printed form for private use is permitted without payment of royalty provided that (1) each reproduction is done without alteration and (2) the Journal reference and IBM copyright notice are included on the first page. The title and abstract, but no other portions, of this paper may be copied or distributed royalty free without further permission by computer-based and other information-service systems. Permission to republish any other portion of the paper must be obtained from the Editor. 0018-8670/08/\$5.00 © 2008 IBM

Quality infrastructure and application components and special solutions are both required and costly. However, investing in good management practices, architecture, and design provides significant returns, in terms of availability.

HA DEFINITION AND PROGRAM GOALS

In this paper, HA refers to application or service availability targets of 99.9 percent or higher availability. In contrast, the Service Availability Forum² defines HA applications or services as applications or services with an availability objective of “five nines,” i.e., 99.999 percent. For an application that can be accessed at any time, the former definition (99.9 percent) implies unscheduled downtime of 8.76 hours (525.6 minutes) and availability of 8751.24 hours per year (given 8760 hours in a non-leap year). This is equivalent to a few unscheduled outages in a year and a restoration of service in minutes. In order to provide this level of availability, an application or service requires a set of HA technologies, IT processes, and services supporting HA (the focus of this paper), as well as an IT organization that supports HA.

The first critical aspect of HA management is the understanding and documenting of customer requirements for availability. Understanding the business requirements clearly can help minimize overinvestment in areas that do not add needed value. Reaching this understanding can be a joint effort of the availability management, service level management and service financial management, requirements engineering, and architecture teams.

Most organizations use a combination of a number of models and analytical tools such as business impact analysis, EA (enterprise architecture) models, business models, service models, and service cost models to classify applications in terms of their level of criticality to the enterprise. Once an application is classified as mission- or business-critical, its availability objective is generally set in the HA range (i.e., 99.9 percent or higher).

Based on experience and industry trends, there are four key goals associated with HA: (1) maximizing or extending application or service uptime, i.e., mean time between service failures (MTBSF); (2) eliminating or minimizing the impact of service-related incidents by detecting and resolving component incidents before they impact application or

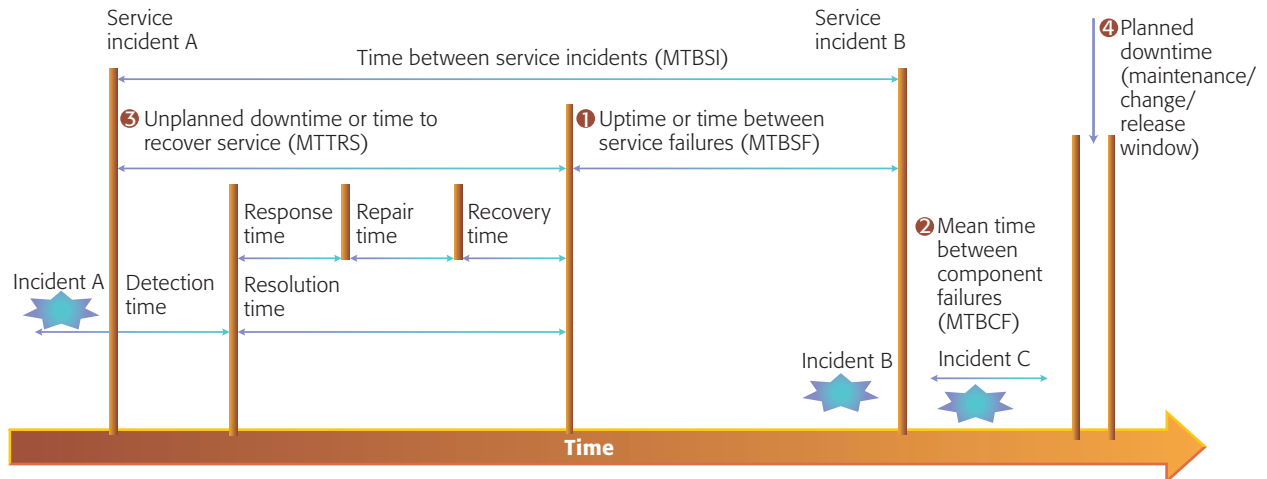
service availability; (3) minimizing unplanned or unscheduled downtime of applications or services, i.e., mean time to recover service (MTTRS); and (4) eliminating or minimizing planned or scheduled downtime (i.e., downtime for changes, releases, and maintenance work).

These four goals are related to *reliability* (as measured by MTBSF and mean time between component failures), *maintainability* (as measured by MTTRS and related to unscheduled or unplanned downtime (which may involve data or service restoration, component recovery, service or component failover, etc.) and *serviceability* (related to scheduled downtime). These goals are consistent with ITIL* (Information Technology Infrastructure Library) version 3 service design documentation.¹ Certain IT service availability literature relates these terms to continuous operations (CO) and continuous availability (HA + CO) (as in the case of the Service Availability Forum). The four key goal indicators (KGIs, a term introduced in the COBIT* [Control Objectives for Information Technology] framework³) and their associated IT processes are shown in *Figure 1*.

Table 1 lists the key performance indicators (KPIs) for the four KGIs associated with HA. For KGI 1 some examples of availability management improvements include removing each single point of failure (SPOF); building redundancy at all layers; automated application recovery (in milliseconds); implementing application and infrastructure HA technologies; and implementing best-practices HA configurations. For KGI 2 the root causes for these incidents can be related to change, release, overload, performance, SPOFs, human errors, known errors, or other factors.

AVAILABILITY MANAGEMENT FRAMEWORK

Once business goals and objectives related to HA have been established, the management framework to achieve the desired results must be put into place. In addition to the normal characteristics of any management framework, there are eight areas that are specifically impacted when managing HA solutions. Good governance (decision rights, accountability, clarity, and transparency) are required in any management system. Governance aspects are outside the scope of this paper since they are not unique to HA. The availability management frame-



Key goal indicator (KGI)	IT process associated with KGI
1 Maximize or extend service uptime (Application uptime—MTBSF)	Availability management, service level and service financial management, continuity management, application management (design and development), performance and capacity management, testing, evaluation, and validation
2 Eliminate or minimize service-related incidents (Detect and resolve component incidents before they impact service availability)	Event/incident management
3 Minimize unplanned downtime (MTTRS)	Incident/problem management (outage management), continuity management
4 Minimize planned downtime (Maintenance/change/release window)	Change management and release management

Figure 1
Four key goal indicators associated with HA

work discussed here is part of the IBM Process Reference Model for IT.⁴

To implement availability management successfully, there must be a clear understanding of business and customer availability requirements. Information must be integrated to enable proper analysis and planning. The customer must agree to invest in proactive and preventive measures. A system must be put in place for event and incident management and monitoring, and an HA power system (such as redundant, scalable, and fault-tolerant power supplies) must be installed and regularly tested.

HA risks must be proactively detected and addressed. Critical system and service components must be identified and continuously monitored. There must be high-quality components, smart redundancy (i.e., redundant systems with externally controlling hardware or software devices), and

tested contingency plans. There is also a need for continual improvement actions and updated availability plans based on service outage, system and service failure, component failure impact, fault trees, and other assessments and analyses.

Availability requirements analysis must be performed regularly. Service-level agreements (SLAs) must be used to define availability requirements clearly. The service incident escalation process must be clearly understood and based on a classification of availability incidents, and the business costs of interrupted service must be specified and quantified as a basis for understanding genuine availability requirements.

Availability management should include the following management controls: a well-defined and established availability management framework; availability plans (enterprise-wide and per service);

Table 1. KPIs for the four KGIs associated with HA

KGI 1: Maximizing uptime	KGI 2: Minimizing impact of service- (or application-) related failures	KGI 3: Minimizing unplanned downtime	KGI 4: Minimizing planned downtime
Actual availability of service or application over 1 year or by time period.	Number of service-related or application-related incidents (by time period or by service)	Number of service-related or application-related incidents (by time period, by service, by incident type, or by root cause)	Amount of planned downtime by service, by application, or by time period, compared to agreed-upon planned downtime in SLAs
Number of SLA breaches (related to availability objectives) by time period or by service	Number of service-related or application-related incidents detected before impacting service or application availability	Percentage of service-related or application-related incidents by time period, by service, by incident type, or by root cause	Percentage of planned downtime by service, by application, or by time period, compared to agreed-upon planned downtime in SLAs
Number of availability management improvements by time period or by service	Number of service-related or application-related incidents detected and diagnosed before impacting service or application availability	Overall time to recover from incidents (by time period, by service, by incident type, or by root cause): mean time, median time, mode time	Amount of planned downtime by reason for planned downtime (change, release, maintenance, break fix, etc.)
Number or percentage of CI (configuration item) failures in production environment without impacting service availability	Number of service-related or application-related incidents detected and repaired before impacting service or application availability	Time to detect incidents (by time period, by service, by incident type, or by root cause): mean time, median time, mode time	Percentage of planned downtime by reason for planned downtime (e.g., change, release, maintenance, break fix)
Number or percentage of CI failures in production environment impacting service availability	Number of service-related or application-related incidents by root cause, by service, or by time period	Time to respond to incidents (by time period, by service, by incident type, or by root cause): mean time, median time, mode time	Number or percentage of changes performed live in production environment (without planned downtime)
Number or percentage of CI passing HA configuration audit by time period or by service	Percentage of service-related or application-related incidents by root cause, by service, or by time period	Time to diagnose incidents (by time period, by service, by incident type, or by root cause): mean time, median time, mode time	Number or percentage of upgrades or releases performed live in production environment (without planned downtime)
Number or percentage of live changes implemented successfully without impact to service availability by time period or by service		Time to repair incidents (by time period, by service, by incident type, or by root cause): mean time, median time, mode time	Number or percentage of maintenance performed live in production environment (without planned downtime)
Number or percentage of changes implemented successfully within change window and without impact to service availability by time period or by service		Time to recover from incidents (by time period, by service, by incident type, or by root cause): mean time, median time, mode time	Number or percentage of break fixes performed live in production environment (without planned downtime)

Table 1. Continued.

KGI 1: Maximizing uptime	KGI 2: Minimizing impact of service- (or application-) related failures	KGI 3: Minimizing unplanned downtime	KGI 4: Minimizing planned downtime
Number or percentage of releases implemented successfully within release window and without impact to service availability, by time period or by service			Number or percentage of changes or releases completed successfully within change/release window
			Number or percentage of changes or releases requiring extension of change/release window

the identification of critical IT resources; maintenance of the availability plans; a security policy; SLAs, operations-level agreements (OLAs) and underpinning contracts (UCs); an IT strategy; an IT plan; and a service catalog.

The following are metrics that can be used to measure the performance of an availability management system:

- Duration of time, human effort, costs, and skill levels required to perform process activities.
- Length of time since last enhancement to the availability management framework.
- Percentage of critical IT components and services within agreed-to availability requirements.
- Percentage of critical components and services with an up-to-date HA plan.
- Number and percentage of critical IT components and services with defined and agreed-to HA requirements.
- Number and percentage of critical IT components and services with agreed-to HA and recovery design criteria.
- Number and percentage of critical IT components and services with defined, agreed-to, and implemented HA targets and related measurements.
- Number and percentage of critical IT components and services monitored and reported.
- Percentage of breaches in agreed-to levels of HA investigated for the causes of unavailability (service level breaches as well as defined threshold “warning” breaches).
- Number and percent of availability plans approved, funded, and executed.

- Time since last availability management audit.
- Time since last availability management assessment.
- Time since last update to availability management strategy and design.

The following are metrics that can be used to evaluate the outcome of availability management:

- Percentage, duration, frequency, impact, and cost of availability incidents (based on SLAs).
- Percentage of availability incidents breaching agreed-to service level requirements.
- Percentage of critical IT components and services with MTBF within agreed-to service levels.
- Percentage of critical IT components and services with mean time between incidents (MTBI) within agreed-to service levels.
- Percentage of critical IT components and services with MTTR within agreed-to service levels.
- Percentage of HA SLAs without breaches.

Implementing a management architecture requires several technical components, at a minimum. These include monitoring tools for all infrastructure and applications, a configuration management system, and an SLA or service-level monitoring system. Among the tools that are required are tools for service monitoring, service diagnostics, system automation, service testing, performance testing, component and service downtime data capture, diagnostic and root cause analysis, load and stress testing, and vulnerability scanning. In addition, there must be documentation or schematics de-

scribing configuration and logical connections, and a database of known errors.

A number of models must be used in availability management, including general and service availability models, general and service availability cost models, and general and service availability measurement models. Required methods include techniques for business impact assessment, risk assessment, risk analysis management, component failure impact analysis, and FTA. Among the required data is security monitoring data, incident information, configuration information, change information, problem information, and operational monitoring data. These data sets can be used for event correlation to predict availability incidents and perform diagnosis and root cause analysis. Outputs include change requests, availability plans, service availability improvement plans, availability reports, service and solution availability requirements (for use by service-level management in SLAs), and critical IT component and service availability, reliability, and maintainability reports.

When creating an availability management system, designers must consider the following questions relating to organizational capabilities: Is the number of technical and managerial staff adequate? Are there adequate skill sets (i.e., technical, process, and managerial skill sets)? Is training and continuous learning available for both technical and managerial staff? Have skill sets and training been mapped to IT-enabled services and processes? Have they been mapped to HA services and processes? Have organizational resources been mapped to HA applications services? Has organizational performance management been modeled after IT service management and performance (measured in terms of availability and other qualities)? And have the solution development life cycle teams been integrated with HA management inter-domain teams and matrix organizations?

SERVICE LIFE CYCLE STAGES AND HA

ITIL version 3 identifies five stages of the services life cycle: (1) the service strategy management process (including service portfolio and pipeline management, service requirements management, and service financial management); (2) service design processes (including the service-level management process, the service catalog management process, the availability/capacity/continuity man-

agement process, information security management, and supplier management); (3) service transition processes (including change, asset, configuration, and knowledge management, transition planning and support, release and deployment management, and service testing, validation, and evaluation); (4) service operations processes (including event, incident, and problem management, request fulfillment and access management, and the service improvement process); and (5) service improvement processes.

Each of these five stages and the IT processes within each stage of the service life cycle impacts one or more of the four key HA goals. *Figure 2* depicts the relationship between the service life cycle and HA processes and services.

Service strategy

Service strategy helps in defining availability requirements and rationalizing expenditures for improving service availability by detailing the relationship between service, IT, functional, and business strategy. As an element in service strategy, *service portfolios* can be grouped into service tiers, with each tier having its own set of service-level objectives (SLOs) and service-level requirements (SLRs). The service targets for each SLO may vary by service tier. Service tiers can in turn include availability tiers, with key differences in their availability objectives. Several enterprises are engaged in developing and refining these service-level and service-availability tiers for such reasons as alignment between business needs and IT capabilities and efficiency in the use of patterns, technologies, products, and building blocks associated with each tier. The service pipeline process that includes identifying new services has the potential to benefit from service-level and availability tiers, since new services can be rationalized and classified into one or more tiers.

Key SLOs associated with service availability can help with gathering and documenting service availability requirements. This requirement-gathering process can result in classifying services into different service tiers. Service financial management can help with rationalizing expenditures for improving service availability by detailing the relationship between services, IT, functional, and business strategy. The cost of service unavailability and the cost of improving service availability are

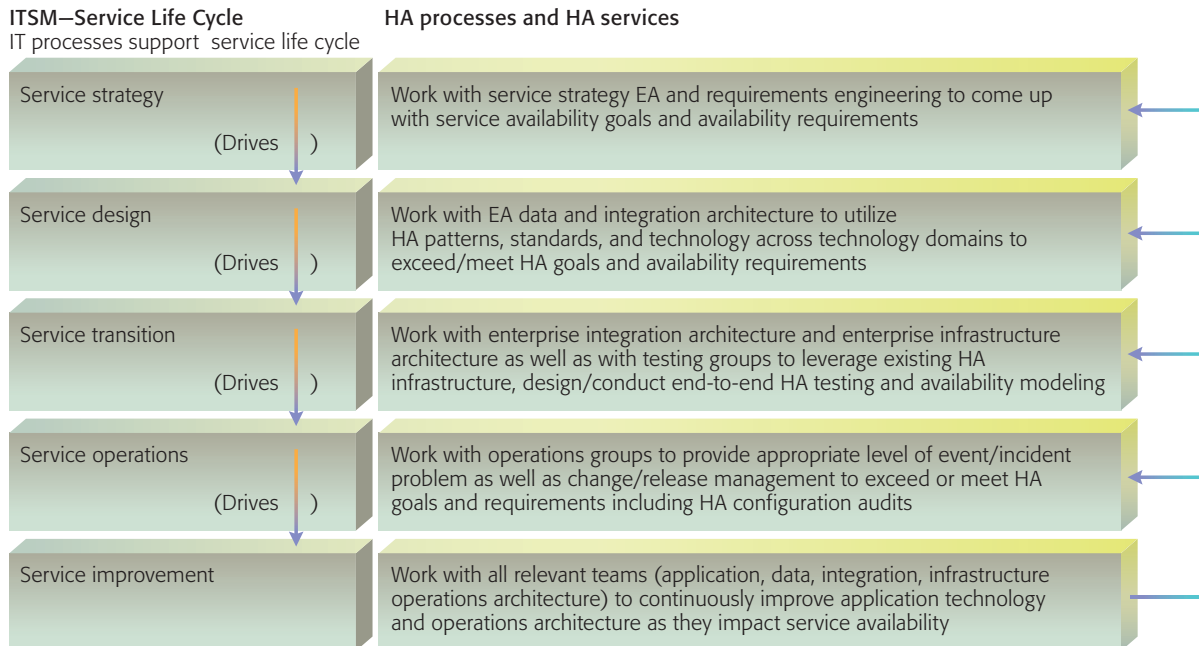


Figure 2
Stages of the services life cycle

two critical inputs for better NPV (net present value) and ROI (return on investment) analysis for expenditures on service availability.

Service design

Service design involves determining and documenting service requirements (including those related to availability) and designing services to meet or exceed a set of functional and nonfunctional requirements. Availability management is an IT process that is part of the service design stage of the service life cycle. Service design is directly responsible for using availability architecture patterns, technologies, and standards,² in both processes both in the application design and technology infrastructure design processes. The ITIL version 3 service design concept is a critical change from ITIL version 2. In version 2, availability management was part of service delivery. By moving it to service design, ITIL version 3 makes it clear that waiting until service delivery to plan service levels, availability levels, capacity levels, continuity plans, security plans, and financial plans will not result in an efficient design.

In many IT organizations, only the technology solution is designed. ITIL version 3 calls this the design of the *service utility*. It is insufficient for the

service utility or the functional technology solution alone to be designed and delivered to operations personnel. The core concept of ITIL version 3 is that a *service package* should be delivered. The service package is made up of the service utility as well as the service warranty. The *service warranty* is made up of the sum total of the service plans for the solution. These include the SLA, the availability plan, the capacity plan, the continuity plan, the security plan, and the financial plan.

The integration of service management best practices throughout the life cycle is the core idea motivating the ITIL version 3 service life cycle. Integrating service management best practices into the development life cycle requires life cycle teams with checklists and policies to guide decision making through each lifecycle stage. Service product management is required to monitor the life cycle activities. The next section will give a more-detailed discussion on availability architecture patterns, standards, and technologies as they impact service design and the service development life cycle.

To clarify how service design processes are impacted by patterns, technology standards, and related technologies, we give two examples, one at the application/data level and one at the IT level. At the

application level, the J2EE* (Java* 2 Enterprise Edition) standard JSR 88⁵ and the associated APIs are used for live application upgrades. The J2EE standard is used for application-level clustering. The J2EE standard JSR 77⁶ and compatible tools are used for state maintenance and session management. There are several application clustering and database clustering tools on the market. The J2EE standard can be used for application-level clustering.

At the infrastructure technology layer, a consolidated enterprise-wide SAN-based replication solution is used when possible (application-specific replication can be used as an exception, for special business requirements). Enterprise-wide server, storage, and network virtualization technologies are used to enable application component isolation, on-demand scalability, and quick recovery on standby virtual machines.

Service transition

Service transition moves the service package into operational mode and involves the development of the base configuration information and knowledge management related to the service. This includes documentation of the service architecture, service-related operational procedures, and other service-specific documentation. Service transition also involves the testing, evaluation, and validation of the service in a preproduction environment including testing, evaluation, and validation of HA technologies and capabilities. Change, release, and transition planning must also be performed, including operational readiness and final production deployment.

Processes employed in service transition include asset configuration and knowledge management, change management, transition planning and support, release and deployment management, and service testing, validation, and evaluation processes.

Asset management includes assets associated with HA technologies and tools. Configuration management includes identifying and tracking HA technology related configuration items and HA configuration audits, among others. Change management includes changes associated with HA technologies and availability improvement plans. Availability modeling and availability testing are also part of the service transition phase.

Service operations

Service operations in production involve operational activities such as

- Event and incident management, which is critical for MTBSF, MTBCF, and MTTRS.
- Post-deployment configuration audits, including HA configuration audits.
- Post-deployment operational audits, such as change audits and maintenance audits.
- Advanced change management capabilities and change models for HA services and applications.
- Advanced release management capabilities and release models for HA services and applications.

Post-deployment operational work also involves day-to-day maintenance activities, both reactive and proactive, operational, infrastructural, and minor code-related changes, and major releases.

Operational activities associated with operational-level agreements and vendor contracts, such as vendor-specific preventive maintenance, configuration, and HA configuration audits, hardware refresh, and hardware and software release activities (as part of vendor contracts). Other activities include service-specific preventive maintenance, configuration and HA configuration audits, hardware refresh, and hardware and software release activities (as part of OLAs). All of these are critical activities for meeting or exceeding service availability objectives.

Service improvement

Service improvement includes the development and implementation of service, application, and infrastructure availability improvement plans. Availability improvement plans can be based on thorough availability architecture analysis (i.e., identifying gaps between current availability capabilities and target availability architecture) or on the ad hoc development and implementation of service, application, infrastructure, and operational architectural improvements as they relate to and impact service availability.

Figure 3 depicts the relationship between HA processes and services and IT service management (ITSM) tools. All of these HA-related processes and services can provide critical input for service availability improvement plans.



Figure 3

HA processes and services and IT service management tools (EA MDR: enterprise architecture metadata repository; SDLC: systems development life cycle; SCM: software configuration management; CCMS: change and configuration management system)

AVAILABILITY PATTERNS, STANDARDS, AND TECHNOLOGIES

In the context of this paper, the term *patterns* refers to the definition of a specific approach to solving a problem. We have briefly discussed architecture and design patterns that are specific to making applications, their data, and the associated infrastructure highly available. Patterns are foundational, reusable, and are typically implemented on multiple technology domains, and do not change in the short or medium term. Technology implementations of patterns change more frequently than the patterns themselves. An example of a pattern is the clustering of system resources.

We use the term *technique* to refer to a rigorous set of steps that have to be followed in order to document and define the scope of a problem.

Examples include component-failure impact analyses and SPOF surveys.

The standards relevant to the topics discussed in this paper include those developed by industry associations such as SNIA (Storage Network Industry Association) and DMTF (Distributed Management Task Force), government bodies such as NIST (National Institute of Standards and Technology) in the U.S. and OGC (Office of Governance and Commerce) in the U.K. for ITIL, and nongovernmental bodies such as ISO (International Standards Organization) or IEEE and organizations such as the .NET Framework community or the J2EE community. There may be one or more standards associated with a pattern. Technologies are also associated with patterns, and may be standards-compliant or noncompliant.

In the following section, we discuss some patterns that are useful in developing availability architectures.

Availability architecture patterns

Redundancy patterns are those which introduce redundancy into the hardware or software of a system in order to increase system availability and minimize downtime. These patterns utilize technologies such as the use of redundant and parallel application components; redundant arrays, or *farms*, of physical or logical servers; RAID (redundant arrays of inexpensive drives); and Internet protocol load balancing or network load balancing.

Technologies related to clustering and grid patterns include failover clusters, active clusters, application-aware and intelligent clusters, network grids, compute grids, and grid storage. For availability and event-monitoring patterns, related technologies include application monitoring, resource monitoring, event monitoring, incident monitoring, and event correlation and availability prediction.

Fault and error management patterns are those related to the handling of faults or errors in ways which minimize the effects of these failures. This includes making systems and subsystems fault-tolerant, utilizing automated restart following an error condition (which may involve “watchdogs” and agent-based or agent-less restarts), and fault, error, and failure monitoring.

Other patterns include *replication patterns* (which replicate data, synchronously or asynchronously, to minimize downtime, in an application-aware or application-unaware manner), *virtualization patterns* (which abstract one or more physical resource to one or more logical resource), *maintaining state patterns* (intended to maintain server or session state), *concurrent maintenance patterns* (for live or “rolling” changes or upgrades, where one system supporting a service is taken offline for changes and upgrades), *stand-in processing patterns* (involving the preparation and availability of a temporary, typically downgraded replacement or stand-in service, database, or system in case of a failure), and *process manager patterns* (related to change management, configuration management, and event, incident, or problem management tools).

Each of these availability patterns may or may not have associated standards. Typically, there is a one-

to-many relationship, with multiple standards associated with a pattern. As an example, one of the technology standards associated with live changes is JSR 88, a J2EE standard for live application upgrades and multiple J2EE vendor implementations. This pattern and the use of the standards and technologies associated with this pattern (i.e., the live changes pattern) are critical for KGI 4.

A high-level view of the impact of availability architecture patterns on service and technology architecture and implementation is shown in *Figure 4*. The architecture patterns shown can be part of the architecture management process, including EA development and solution architecture development.

HA AND ITSM MATURITY LEVEL

A minimum *maturity level*, as defined by the EA processes groups and ITSM process groups of an organization, is required for HA service management.⁷ HASM is associated with the highest levels (levels 4 and 5) of ITSM maturity (see Reference 8 for an example of one of the many ITSM maturity models) and some level of EA maturity (for example, see the four stages of architecture maturity⁹ proposed by the MIT Center for Information Systems Research). HASM requires an IT environment that is managed, optimized (for availability), and dynamic (i.e., capable of proactively and quickly responding to availability-related events and incidents), supporting business process and service availability.

The key dimensions of HASM maturity include dedicated internal organization for HA program and availability management and strong leadership and executive sponsorship; organizational “buy in” with an integrated view of availability architecture and availability management processes; and availability patterns, standards, products, and technologies that are built into the systems development life cycle (SDLC, for both software and infrastructure development) and configuration management (including software configuration management [SCM]).

Maturity models

Maturity models indicate how well a service or process is performed. Typically, the maturity levels scale from no activity, to managing a service or process in a basic way, to managing it optimally.



Figure 4
Availability architecture patterns and service and technology architecture

Maturity models typically have four or five levels of maturity and it is necessary to perform activities at the lower levels and build capabilities before achieving higher levels of maturity. Generally speaking, maturity models are very useful for diagnosing and then planning management improvements.

A maturity assessment is very useful as a means to perform a detailed diagnostic of the capability of a management system. The IBM service management maturity model defines each of the five levels of maturity for each of the 46 management processes in the IBM Process Reference Model for IT.⁴ Each process is broken down into different categories to be assessed. Within each category, there may be 5 to

20 different line items. The categories are: management framework; performing process activities; interfaces; organization (roles, teams, and functions); technology components and architecture; information; and management controls and measurement (including COBIT management guidelines).

Maturity models are concerned with performance; they do not indicate why an activity should be performed, what business outcome is to be achieved, or what specifically is being managed. Availability management, like all other management practices, can be applied in more than one context. This is where adoption models come into play.

Adoption models provide a business outcome context for the management practice.

Adoption models

Adoption models differ from maturity models and are based on what is being managed, rather than how well it is being done. Adoption models focus on outcomes rather than performance. The IBM service management adoption model describes five levels of adoption of service management best practices. Each describes what is being managed in the order in which these management practices evolve. Service management best practices, as with availability management or capacity management, are usually adopted in specific areas, usually at different times. The same management practices, and often the same underlying management tools and architecture, can be applied to increasingly valuable business outcomes.

A common way to describe the evolving adoption of service management best practices posits three levels of adoption: enterprise systems management (for infrastructure and applications), IT service management (IT services, which are configurations of infrastructure and applications), and business service management (business processes, business services, and supply chain management, which are also configurations of infrastructure and applications).

In practice, moving among these levels is complex, challenging, and often difficult. We have found that it is helpful to conceptualize five levels of adoption: (1) managing IT infrastructure, usually in discrete technology-oriented silos; (2) managing applications, end-to-end across the infrastructure, which usually requires greater integration between development and operations and therefore requires governance changes; (3) managing IT services, which requires complete integration inside IT, between development, operations, and other functions focused on IT business alignment, and requires additional changes to governance; (4) managing business processes and business services, which requires integration between all of IT and the business; and (5) managing supply chains, partner “ecosystems,” and dynamic collaboration.

When considering HASM, it is thus critical to establish what is being made highly available: is it infrastructure, applications, IT services, business processes, or business services? Within the context

of what is being made highly available, a maturity model is useful to help diagnose and improve the management capability at the level of maturity required to achieve the desired business outcomes. By recognizing the distinct concepts of maturity and adoption, customers can make management decisions that target the exact areas of management capability that should be changed based on business requirements and specific desired business outcomes.

Put another way, one should avoid increasing management maturity merely for the sake of maturity. A specific overall level of maturity is not a valid business objective. However, improving specific management capabilities for managing specific qualities can be very critical to achieving business objectives and controlling costs.

Enterprises in several industries are at different stages of design and implementation of IT service management programs and HA programs. These enterprises are thus at different levels of service management maturity and HASM maturity.

CONCLUSION

This paper took a holistic view of HASM and discussed some of the key aspects associated with HA and HASM from the perspectives of availability objectives and requirements, an availability management framework, the service life cycle (i.e., service strategy, design, transition, operation, and improvement), IT processes, ITSM maturity, and HA patterns, technologies, and standards.

We have discussed the need to have an integrated view of EA architecture and HASM, as HASM is associated with the highest levels (levels 4 and 5) of ITSM maturity and some level of EA maturity. HASM requires a managed, optimized, and dynamic IT environment supporting business process and business service availability.

*Trademark, service mark, or registered trademark of the Office of Government Commerce (UK), the Information Systems Audit and Control Association and the IT Governance Institute, or Sun Microsystems, Inc., in the United States, other countries, or both.

CITED REFERENCES

1. *Service Management—ITIL Version 3*, Office of Government and Commerce (OGC), U.K., <http://www.ogc.gov.uk>.

best-management-practice.com/Portfolio-Library/
IT-Service-Management-ITIL/ITIL-Version-3/
?trackid=002202&DI=582733.

2. *SA Forum Educational Material*, Service Availability Forum, <http://www.saforum.org/education/>.
3. *COBIT*, Governance Institute (ITGI), <http://www.isaca.org/cobit/>.
4. *The IBM Process Reference Model for IT (PRM-IT)*, IBM Corporation, http://www.ibm.com/software/tivoli/governance/servicemanagement/welcome/process_reference.html.
5. *Java Specification Request JSR 88: Java EE Application Deployment*, Java Community Process (2006), <http://jcp.org/en/jsr/detail?id=88>.
6. *Java Specification Request JSR 77: J2EE Management*, Java Community Process (2005), <http://jcp.org/en/jsr/detail?id=77>.
7. R. Radhakrishnan, *Enterprise Architecture and IT Service Management*, White Paper W078, The Open Group (2008), <http://www.opengroup.org/architecture/wp/>.
8. M. C. Paulk, C. B. Weber, B. Curtis, and M. B. Chrissis, *The Capability Maturity Model: Guidelines for Improving the Software Process*, Addison-Wesley Professional, Boston (1995).
9. J. W. Ross, P. Weill, and D. C. Robertson, *Enterprise Architecture as Strategy: Creating a Foundation for Business Execution*, Harvard Business School Press, Cambridge, MA (2006).

Accepted for publication July 21, 2008.

Published online November 4, 2008.

Rajesh Radhakrishnan

IBM Global Technology Services, 12174 Chancery Station Circle, Reston, VA 20190 (rajesh.radhakrishnan@us.ibm.com).

Mr. Radhakrishnan is a senior managing consultant and senior IT architect for IBM Global Technology Services (GTS). His current client engagements involve global ITIL® program management, service-level management, and service availability/continuity management. Prior to joining IBM, he was an independent IT architect at Zygous, LLC, for two years and a senior technical consultant at Sun Microsystems, Professional Services, for nearly five years. He was also a senior technical staff member at Metamor Worldwide for five years and consulted for numerous Fortune 500 companies. He specializes in enterprise architecture and service management consulting, business continuance, high-end systems, high availability, storage technologies, and technical project management. He has an M.B.A. degree from Old Dominion University and an M.S. degree from the University of Virginia, McIntire School of Commerce. He was awarded the VITS (Virginia IT Scholar) for 2003. He is certified as a Solaris Systems Administrator, storage architect, TOGAF (The Open Group Architecture Framework) practitioner, ITIL-ITSM consultant (manager's level), IBM Senior IT Architect, and as an Open Group Master Architect.

Kari Mark

IBM Global Technology Services, 320 Macalester Street, Saint Paul, MN 55105 (kamark@us.ibm.com).

Ms. Mark is a certified senior managing consultant and delivery manager with IBM Global Technology Services in the business continuity and resiliency services practice. With more than twenty-five years of experience in the IT industry, she has extensive experience in all aspects of designing, implementing, and operating high-availability (HA) systems, for a variety of computer platforms, in a variety of client

environments. Most recently, she helped develop an availability architecture framework for the assessment and design of HA systems to ensure that the resiliency characteristics of the underlying infrastructure align with client business goals. In addition, she has extensive experience with designing and implementing dual data center strategies to achieve business resiliency objectives. Prior to joining IBM, she was the technical support director for a large, multi-data-center customer environment. She has a Bachelor of Computer Science degree from the University of Minnesota Institute of Technology, and is ITIL Foundations and PMP Certified.

Bill Powell

IBM Global Technology Services, 7045 Glenwood Avenue, Golden Valley, MN 55427 (wdpowel@us.ibm.com).

Mr. Powell has 20 years of diverse experience in a variety of IT environments. He has served in a variety of roles, including IT services manager, vice president of technology services, product manager, service product manager, IT management consultant, and education developer. He has spoken at many service management events globally and is prominent in the service management profession. He has been a member of the OGC ITIL advisory group, the itSMF USA® Management Advisory Board, and the ANSI Technical Advisory Group for Service Management and Governance Standards, and has been the U.S. representative to the International Standards Organization (ISO) for positions on international standards related to service management and governance. He is a member of the ISO/IEC working groups for governance, the board of directors for the Institute of Certified Service Managers, and the leader of the IBM ITIL interest group, as well as a member of the IBM IT Service Management core team. He has been recognized for outstanding management and service delivery performance, as well as unique and creative perspectives on problem resolution. He has received numerous professional performance awards, including recognition for leadership, planning, and technical ability. ■