Continuous Availability for Real-Time Services: 24x7x365 -- Systems, Data, and Applications

October 2005

Dirk Epperson, Kabira Technologies, Inc.

Lei Liu, Sun Microsystems, Inc.

Table of Contents

1 Executive Summary: An Integrated Continuous Availability Strategy Using Open Systems

To remain competitive with new global players in today's 24x7 global services economy, enterprises must have an IT strategy supporting immediate response times while improving cost efficiencies. These IT processes supporting real-time business transactions and services must also operate continuously, every day of the year.

Any solution must be able to reduce the costs of transactions, improve business efficiencies, support constantly changing and complex business processes, improve time-to-market for new services, and demonstrate ROI models supporting these business objectives.

In response to this trend, technology companies have introduced state-of-the-art open system products to reduce the costs of building, testing, and deploying these mission-critical systems. New simultaneous multi-processing (SMP) servers and blades, 64-bit multi-core processors, 64-bit multi-threaded operating systems like the Solaris™ OS and Linux, and high-speed network storage will change the landscape for competing in this 24x7 real-time environment.

The next challenge is for enterprises to develop an integrated approach providing "always-on" processing and applications services. First-generation high availability (HA) architectures were built using storage and cluster technologies to support data-driven applications during the 1980's and 1990's. Today, these HA systems need to support event-driven services, "always-on" service availability, centralized and distributed transaction processing, application scalability, and the flexibility to adapt to constantly changing business flows.

This is placing a new challenge on IT engineering teams to come up with a next-generation HA architecture supporting continuous availability. Continuous availability is the synergy of operating system, applications, servers, system management tools, and procedures that support real-time business requirements requiring low latency operations 24x7x365 days a year. Continuous availability services must be designed and implemented system-wide throughout the entire enterprise business system.

Sun Microsystems and Kabira Technologies have created a solution to deliver an integrated continuously available system with the processing speeds of a mainframe, on cost-effective, open system architectures. This degree of integrated high availability results in reducing transaction costs, improving agility of the IT systems and time-to-market for these mission-critical systems.

This article discusses the following key subjects:

1. How the evolving demands of real-time business models are driving technology to produce faster, more reliable, higher-capacity, extensible, and scalable open systems for transacting mission-critical business, all while fulfilling demands for lower total cost of system ownership.

2. How Sun partners with Kabira to fulfill an end-to-end approach to data and application protection that pushes beyond high availability, to system-wide continuous availability.

3. How the combined continuous availability functions built into the systems design and into the application-level infrastructure can lead to the fastest, most secure, low-latency system possible.

Fully integrated continuous availability is characterized by the following elements:

1. Automated operations
2. Automated failure detection and recovery
3. Application level HA
4. Security management
5. Application testing standards
6. Application-level system design standards
7. Service level agreement (SLA)
8. Change management

2 The Evolution of Technology to Real-Time High Availability Architectures

The networked services and financial industries are driving technology to meet an open-ended call for more flexible, open systems, and higher-capacity, real-time transaction processing at ultra-fast speeds. These industries are also behind the demand for high availability functionality that protects mission-critical applications and data.

The high availability evolution began when storage technology allowed the mirroring of data across disks and storage systems. Today, the real-time processing speeds and reliability presented by Kabira's memory-resident, multi-function software are redefining the standard of high availability.

Technology has evolved to enter a new era in operating system support for real-time flows. It is now possible for mission-critical businesses to move from disk-based high availability in a clustered environment to real-time, high-performance transactionality, without sacrificing reliability.

Sun Enterprise™ server configurations can meet the challenges of business' parallel quest for lower price/higher performance and reduced total cost of ownership, while maintaining availability.

High Availability and Real Time in Sun's Solaris 10 OS Architecture
The Solaris Operating System is designed and implemented with a real-time and highly available services architecture. The Solaris OS offers a compelling environment for real-time, multi-threaded applications, based on the collaboration of the following:

- Preemptive kernel
- Kernel interrupts as threads
- Fixed priority scheduling
- High-resolution timer
- Fine-grained processor control

The multi-threaded execution environment provided by the Solaris 10 OS enables high throughput and application scalability.

High Availability and Real-Time Transaction Flows Built in Kabira
The Kabira Transaction Switch's High Availability (HA) Component is a key element of an integrated services platform. Kabira software is designed to integrate highly available and fault-tolerant functionalities to provide resilience against hardware and software failures, without compromising performance and scalability.

Kabira's 64-bit, memory-resident transaction processing capabilities can achieve mainframe-class speed, reliability, and functionality on Sun's open platform systems, for a lower cost than maintaining clustered systems or proprietary mainframes.

The High Availability Component gives an enterprise business system "five nines" (99.999%) of availability, without reliance upon redundant clusterware, transaction monitors, or databases. Five nines is a carrier-class, high-speed, high-traffic system that has no more than five minutes of downtime per year. It is a low-latency system, where failovers are transparent to users. This prevents interruption of work or lost transactions, and backup and recovery functions occur with no degradation in performance.

2.1 Kabira's Competitive Advantage in the HA Marketplace

The High Availability Component is used for real-time network integration, messaging, and applications. Kabira provides application and transaction fault tolerance without the complexities of HA software or the constraints of proprietary hardware.

- All processes are engineered for deployment on highly flexible Sun Enterprise server hardware and open platform systems (with potential for much lower total cost of ownership).
- Mirroring data into the memory space of another computer with high-speed network connectivity gives better performance than disk-based systems can provide.
- The High Availability Component offers faster replication and faster recovery than can be provided by disk-based clustering solutions. Node failures are detected immediately, without the use of heartbeat mechanisms. The High Availability Component can be enabled for some or all software components in a solution.
- High availability functions hardened within the infrastructure level reduce the necessity for programmers to design, configure, monitor, and employ software architectures needed for real-time services.
- The High Availability Component gives customers building distributed blade server farms much higher levels of HA than would be possible to achieve with complex, disk-intensive clustered servers.
- The High Availability Component enables customers with legacy systems from other companies to migrate to Sun's lower total-cost-of-ownership open platform system, without sacrificing HA functionality.
- The High Availability Component supports legacy HA system constructs, such as heartbeat and transactional replication. Kabira leverages middleware technologies that bridge legacy and modern architectures, adding new, high-performance capabilities quickly and seamlessly, all within a single, unified architecture.
- The High Availability Component maintains application states, without hardware or database dependencies, and does not require special programming to achieve carrier-class HA on a scalable open platform.

- The High Availability Component adds high availability to existing Kabira Transaction Switch (KTS) deployments, retaining the competitive advantages of Kabira's high performance, scalability, and extreme agility.
- Kabira can provide persistence service with caching policies, but does not require that data be persistent.

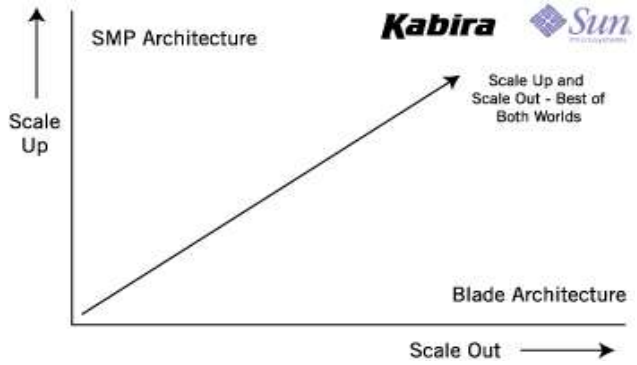

Figure 1: KTS Scales Up and Scales Out

2.2 High Availability Through the Software

A highly available system creates the most stable environment for an enterprise, protecting mission-critical applications from severe impacts associated with planned and unplanned service outages.

High availability minimizes outage frequency and duration, maximizes timely recovery of all data, and makes failover transparent to end users. The solution from Sun and Kabira encompasses these attributes of a highly available system: fault tolerance, rapid recovery from failures, workload balancing, and disaster recovery functionality.

- Sun Enterprise servers provide high-speed, mission-critical server interconnect architecture. The extensive memory capacity of the Sun Enterprise server's SMP server is optimized for its SPARC® IV processor.

  All components of the system are built for interoperability. At the same time, each one is designed to work independently as a function of isolation. As an added safety feature, a failure in one component of a system does not bring down other components of the system.

- The High Availability Component supplies high availability at the application level by incorporating high availability elements within its core architecture.

Applications built on KTS automatically inherit these services:

- Distribution, thread-management, query access, state management
- Caching, indexing, concurrency control, memory management
- Automatic application recovery, memory-resident transaction management
- Logging, queuing, persistence, deadlock detection and correction
- On-the-fly swapping of application logic with no loss of transactions

KTS Delivers Level-4 High Availability
Installing and configuring the High Availability Component adds HA functionality to the application level that achieves Level 4 on the IDC High Availability Spectrum. Optimize system performance by choosing HA for only the services that are to be highly available.

The HA Component utilizes a reliable, asynchronous-capability, event-driven architecture to provide a robust and high-performance solution. Kabira can also use synchronous communications to achieve HA.

- Memory-resident speed and storage capacity allow data to be duplicated instantly across multiple nodes in active/active or active/passive modes.
- Applications can be developed and deployed on Kabira to achieve 99.999% availability (less than five minutes of downtime per year).
- Programmers need not write security-aware and high availability-aware code into the core of their applications.
- During recovery, all transactions should be completed, and no data should be lost.
- Kabira supports asynchronous and synchronous communication services.
- Sun's Enterprise servers and Kabira technology are designed to produce a computing environment with continuous availability, that is, free from outages, 24-hours a day, seven days a week.

2.3 Beyond High Availability to Continuous Availability

The integrated solution is calculated to minimize the frequency, scope and duration of any service outage, either planned or unplanned, and maintain overall continuous availability during system component outages. If a failure occurs in any component during processing, the Kabira High Availability Component will continue to process transactions seamlessly on a backup system in real-time, with failover being transparent to users.

3 Elements of Continuous Availability

Sun Enterprise servers and the Kabira High Availability Component integrate the following elements of continuous availability within the hardware, systems management, and application-level transaction processing software:

- Automated operations
- Automated failure detection and recovery
- Incorporating HA into an application
- Security management
- Application testing
- Application-level system design standards
- Service Level Agreement (SLA) management
- Change management

Each element of continuous availability functions to reduce:

- Frequency
- Duration
- Scope
- Recurrence of service outages

Continuous Availability Element	*Sun Enterprise Server Components*	*Kabira Components*
Automated Operations	• Sun Management Center • Enterprise Management Systems • Core Voltage Telemetry Monitor • CPU Diagnostic Monitor • Sun Fire™ Error Checking and Correction (ECC) • Sun N1™ Grid Engine • N1 System Manager • N1 Service Provisioning System • System Management Agent	KTS
Automated Failure Detection and Recovery	• Solaris Service Management Framework • Solaris Predictive Fault Management Architecture • Solaris Containers • Solaris Resource Management • Solaris DTrace • Solaris Process Right Management • Sun Fire ECC • Solaris Operating System variable monitoring and tracking	HA Component
Security Management	• Solaris System Security • Security Hardening Services • Solaris Zones Partitioning • Service Management Facility • Solaris Fingerprint Database • Basic Audit and Reporting Tool • Sun Managed Services • Sun Java™ Enterprise System Identity Management Suite	Secure Services Layer Kabira Identity Management Services (IMS)
SLA Management	• Sun Management Center • Solaris Management Console • Sun System Manager • Sun Management Center Service Availability Manager • IP Quality of Service (IPQoS)	SNMP Agent Kabira Event Logging and Statistics Component
Change Management	• Sun Management Center • Solaris Service Management Framework • Control Point • Solaris Extended Accounting	HA Component KTS Online Versioning

Table 1: Continuous Availability Platform Components

Note: Based on expertise building real-time services on the Solaris platform, Kabira offers KTS System Design Standard and Application Testing Design Standard best practices on the Solaris platform.

3.1 Automated Operations

The most effective systems maximize the number of automated controls. Automated systems recognize and react to events more quickly than a human can, deploying predictable and consistent reactions to defined events every time.

Automated operations also lend themselves to creating fault avoidance in the enterprise system. Fault avoidance encompasses error-free operations, preventative actions, and management of system alerts/messages.

Designing for automation of a continuously available system augments high availability:

- To ensure non-disruptive changes during scalability changes, application rollouts, and scheduled maintenance
- To recognize and route systems messages to the appropriate components
- To trigger necessary backup, copy, recovery or retrieval functions.

Levels of continuous availability must be identical for all interrelated components. For example, if the maximum level of tolerable outage for your business processes is five minutes, then the applications and data that support that business process must also be designed to recover from a failure in five minutes or less. Continuous availability takes into account the interdependencies between business needs, applications, and data.

3.1.1 Automated Operations in Systems Management

Sun's systems management software manages the following critical areas of the enterprise business system:

- Security
- Operations and administration
- Deployment
- Availability
- Network recovery

Automated operations software can perform the following functions:

- Testing
- Isolation
- Controlled restarts
- Retries
- Performing change management tasks concurrently with regular systems processing duties
- Message sending/event handling

The Sun Management Center provides comprehensive instrumentation and administrative knowledge for monitoring and managing Sun environments. It also features open interfaces that enable information to be shared with other management platforms. Automated corrective actions can be taken at the agent level, minimizing service impact and

reducing the need for manual corrective actions. Combine it with the Sun enterprise management systems to provide a unified management infrastructure.

Sun's enterprise-class Sun Fire™ server platforms (V1280-E25K) and the Solaris OS have predictive fault monitoring features that allow a system to track and monitor variables and analyze or compare the data to set thresholds. Results can be used to warn system operators of an incipient failure.

Other systems management functionalities on Sun include:

- Core Voltage Telemetry Monitor (CVTM) functionality to provide a warning mechanism for processors that show a gradual increase in Vcore voltage
- The CPU Diagnostic Monitor (CDM), an online processor diagnostic program
- Error Checking and Correcting (ECC) logic on Sun Fire architecture that allows for both the detection and correction of single-bit data error disturbances
- A Soft Error Rate Discrimination (SERD) algorithm to detect when a specified number of distinct ECC events have occurred on the same processor in a 24-hour period
- The System Management Agent, the Sun Microsystems implementation of the open source Net-SNMP agent, which supports the AgentX, USM and VACM protocols
- The Sun N1™ System Manager infrastructure lifecycle management software, which helps customers provision, monitor, patch, and manage Sun Fire™ servers built on x64 architecture and workgroup servers based on SPARC technology
- The Sun N1 Service Provisioning System, which simplifies application lifecycle management by rapidly provisioning business services that span multiple tiers (J2EE™ technology-based application servers, web servers, and databases) across heterogeneous environments; offers organizations a standard method of deploying business services and tracking changes throughout the deployment process; and includes an audit trail to help customers meet regulatory compliance

3.1.2 Automated Operations in Application-Level Management

The High Availability Component helps users manage Kabira applications seamlessly by providing the hooks to current systems management software. The automated systems management feature writes log files and sends alerts to a third-party management platform.

KTS applications inherit a service management framework with facilities for managing the following key categories:

- Fault
- Configuration
- Accounting
- Performance
- Security

Table 2 describes the capabilities of each Kabira facility and the service management category in which it is embedded.

Management Category	*Capabilities and Operations*	*KTS Service Management Facilities*
Fault	Fault detection, isolation,correction and recovery; error logging and error statistics; alarm generation and alarm aggregation	• Event service • SNMP channel
Configuration	Initial data loading, resource shutdown, change management, pre-provisioning, software upgrade and migration	• Configuration framework • Switchadmin Interface
Accounting	Service and resource accounting,activity logging, revenue assurance, event correction	• Event Service • Appstats package • Logging package • Security framework
Performance	Performance data and statistics collection, performance reporting	• Event Service • Appstats package • SNMP channel
Security	Access controls, data privacy, alarm generation, security audit trails, user and role-based security	• Security framework • Event Service • Switchadmin interface

Table 2: Kabira Service Management Categories

3.2 Automated Failure Detection and Recovery

A continuously available system operates at the same processing capacity whether the system is running smoothly or has experienced a failure in its hardware or software components. It also masks hardware or software recovery processes from the user, so that repair actions are automatic, fast, and concurrent with the system's continued use and access.

3.2.1 Automated Failure Detection and Recovery in Systems Management

Sun Enterprise servers' automated recovery tasks include active and passive monitoring systems. These systems identify and repair problems before they become apparent to end users.

Sun Enterprise servers automate these recovery tasks:

- Activation
- Deactivation
- Initial program load (IPL)
- System restart
- Network recovery
- Session recovery
- Transfer of workload to a backup system
- Concurrent maintenance and repair

Solaris Service Management Framework (SMF) provides service objects with automatic restart of failed services.

Solaris Predictive Fault Management Architecture (FMA) provides fault diagnosis of system environmentals to warn of an incipient failure.

Solaris Containers technology provides isolation and partitioning between software applications or services with full resource containment and control to produce more predictable service levels.

Solaris resource management provides system resource control, notification, and monitoring with resource isolation and allocation. DTrace is a comprehensive dynamic tracing facility that can be used by administrators and developers on live production systems to examine the behavior of both user programs and of the operating system itself.

Sun Fire servers' ECC logic allows for the detection and correction of single-bit data errors. A Sun Fire server and the Solaris OS allow a system to track and monitor variables, such as correctable or uncorrectable error rates and locations, or system environmentals. The system can then compare the data to set thresholds, or analyze it with complex decision-making algorithms. Results are used to warn system operators of an incipient failure or, in some cases, to remove the offending Field Replaceable Unit (FRU) from the system configuration before it can cause a system failure. The Solaris OS offers fault management architecture that focuses on fault diagnosis rather than performing traditional ad-hoc error handling.

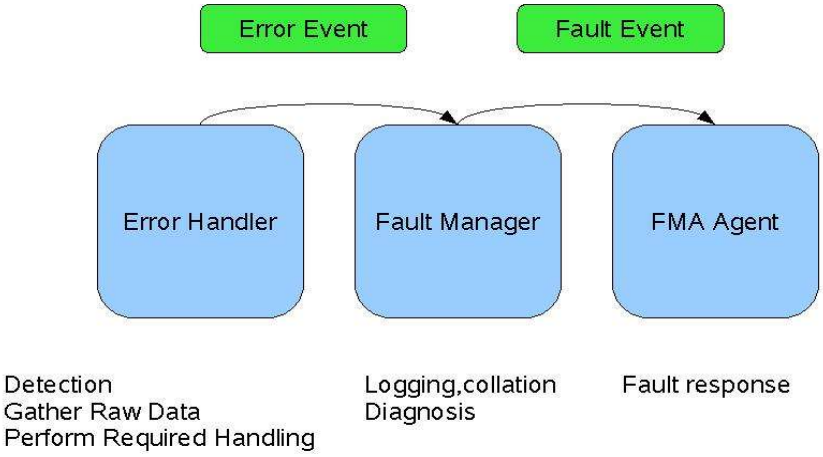

Figure 2: Sun Fault Management Architecture

As illustrated in Figure 2, an error event is an unexpected or anomalous result or condition. An example is a correctable memory error from ECC-protected memory. A fault event is a defect that may produce an error. Not all errors are necessarily the result of a defect.

Error handlers respond to the detection or observation of an error. The error handler takes any required action to properly handle the error and then (if you wish it to report events for the fault manager) it collects error data and prepares an error report for transmission to the Fault Manager.

Some error situations include:

- When a kernel CPU error trap generates a hardware exception in response to some hardware detection of an error
- When a SCSI driver detects an unexpected bus reset
  When an application performs an illegal access and gets a SIGSEGV message (and has installed a signal handler to handle it)
- When `svc.startd`(1M) decides that a service is restarting too frequently

The Fault Manager is responsible for fault diagnosis through pluggable diagnosis engines; they play a central role in the architecture and may be arranged hierarchically with events flowing between them. For example, one fault manager instance might run within each Solaris instance, and another fault manager instance might run on the system controller for the platform. FMA agents subscribe to fault events produced by the diagnosis engines and may take appropriate remedial or mitigating action.

3.2.2 Automated Failure Detection and Recovery at the Application Level

Kabira's High Availability Component eliminates the need for complex data replication strategies and mechanisms. This helps reduce the costs and performance degradation associated with bolting backup and recovery middleware onto a system. Every transaction is guaranteed, and failover is transparent to end users. Recovery consists not only of all data associated with an application, but also the current processing state at time of failure; thus, applications restart from the last successful processing step.

Inherent Recovery Services
Kabira's inherent process recovery functionality supplements the High Availability Component to produce robustness on a single machine. Transaction states are preserved in shared memory. For single-node recovery, in the event of a process failure, the Kabira Process Coordinator will restart the process and roll back all transactions in shared memory, then restart those transactions.

3.2.3 Advantages of Kabira High Availability Component Over Clustered Systems

Cluster solutions are reliant upon disks for recovery. The traditional disk-oriented solutions have negative effects on performance and operating conditions. It can take minutes to recover from failure because of the need for the disks to restore the file system on the backup server. Kabira software is designed to restore services within seconds.

Because Kabira is designed to run on legacy solutions as well, it can take advantage of existing systems while increasing the performance, availability and flexibility of existing solutions.

3.2.3.1 Application Recoverability
The High Availability Component provides transparent recovery support for all applications built on the system. The recovery logic is built into the application infrastructure, relieving application designers from having to build recovery-processing logic into each application. This saves money and programming resources and alleviates the problems associated with making changes to applications after deployment.

Kabira's native recovery logic also reduces the complexity of application modeling, since the recovery logic is not encoded within the application models. Application designers need only concern themselves with handling application-related problems. They will not need to concern themselves with failures of hardware or of most external servers. For example, if a Kabira application is using an external database and the database fails, the system

continues to retry the database connection until the database is restarted. This retry logic is provided by the system, not encoded in the application.

Kabira's inherent HA services can also increase the productivity of senior IT staff, who can switch from HA-aware coding in their applications to a simpler, configuration-driven deployment of their application, messaging, and connectivity solutions.

3.2.3.2 Controlling Data Loss Exposure

Traditional HA services had to be hard-coded into each application, but Kabira's High Availability Component separates the application logic from the HA services. The High Availability Component can be configured for multiple types of data. Users can choose to trade-off potential exposure to data loss with desired performance levels by using synchronous, asynchronous, or scheduled (batch) backup:

- **Synchronous replication** provides the lowest exposure to data loss. Data is processed on the local node and the transaction is backed up to the remote node before the business transaction is "complete." This uses a low latency, high-speed network infrastructure such as Infiniband (which has latency on the order of 50 microseconds).

- **Asynchronous replication** improves performance because it features half the network traffic and much lower latency than the synchronous backup method. With this method, a transaction is processed and sends a message of completion to the client before a mirror is propagated to the backup node.

  This method is ideal for geographic distribution or disaster recovery, where distances between sites may exceed 1000 miles and result in very high transaction latency. The amount of queued data is controlled by the time interval of the backup. Smaller intervals minimize the possible data loss. This method allows for a tiny window of failure, so if your company cannot afford any data loss, then you should use synchronous backup.

- **Scheduled replication** is useful for handling large numbers of very small value transactions, where limiting exposure of lost transactions is most important. For example, in a system charging for wireless messages, there may be a high volume of high-speed messages, but each message might carry a value of only a few cents. It is not cost-effective to ensure that each single transaction charge is never lost, but an accumulator can be set up to mirror a defined group of data on the backup node at timed intervals. For example, you could configure the system to direct that a batch of updates be propagated to the backup when charges total $100. Batching is the most performance-efficient backup method, because there is no "go to backup" after each transaction; however, it is the method with the largest window of possible failure.

On multi-node systems, the High Availability Component router detects failures immediately upon attempting to route the transaction and forces a reroute to a backup node. Each node is configured with routers to provide full redundancy. The application platform infrastructure also contains a configurable component restart function for intra-node recovery.

3.2.3.3 Complete Data Protection

A continuously available system requires its applications to be designed, coded and tested with high availability standards. That is, applications must be able to reduce or eliminate the duration and frequency of planned and unplanned service outages, and to limit their scope. The High Availability Component ensures process recovery and complete data recovery protection. Kabira uses high-speed interconnect, such as Infiniband, to replicate data across

multiple nodes in active/active or active/passive mode. In the event that one node fails, the backup node will recover and complete the transaction on the backup node.

The High Availability Component is based upon two unique functions:

- Continuous mirroring of application data to a backup node (which may also be processing transactions of its own, and whose location need not be constrained by distance to the primary server)
- A routing feature that continually directs messages to the appropriate node that is available to provide a service; this is designed to ensure the completion of every transaction as well as no data loss, even upon failure

These High Availability Component features provide complete data recovery:

- Object replication
- Object partitioning routing and load balancing
- Automatic failover
- Fault tolerance
- Network distribution
- Multi-phase commit across heterogeneous platforms
- Non-repudiation, once-and-only-once delivery

Features of the KTS High Availability Component	***Description***
HA Node	A node is a group of one or more CPUs accessing shared memory. Any KTS node can be configured for HA service. Each node notifies other system nodes of its availability to perform HA services. Multiple HA nodes may run on the same machine to facilitate development and testing, then be deployed on multiple servers. Separate nodes are connected by high-speed inter-processor bus (IPB), such as Infiniband or Gigabit Ethernet. Each node can be configured as "active", as "backup" or as both at the same time. Nodes can be distributed over wide geographic areas while functioning as backups.
Failover Detection and Resolution	Detects failure through asynchronous testing with transactions and activates backup partition(s), enabling recovery in seconds. Compare this with recovery taking minutes on traditional, disk-based systems.
Mirrored Objects	Any object that is necessary to preserve can be configured for HA:dynamic read, write and update triggers are installed to "mirror" the object on one node to a designated backup node in real-time. Upon failure of the primary node, the backup "mirror" of the object on the designated backup node will become the master and resume transactions. Minimize performance impact by replicating only critical stateful application objects.
Object Partitions	These are groups of mirrored application objects that share a backup strategy. Partitions are used for load balancing the work of the applications across one or more nodes, and administrating backup strategies. Each mirrored object is assigned a partition ID to determine which partition the object belongs to. Mirrored objects can be moved from one object partition to another at runtime. The data partitioning scheme ensure data integrity by having a single master copy of each instance and avoiding distributed locks.
Message Router	Tracks status of object partitions. Kabira applications process messages in flows. The flow routes the message to various consume objects for processing. If the target object is a mirrored object, the router directs incoming messages to the appropriate node; this helps ensure the transaction cannot be lost, even if destination fails. Upon failure in one node, the transaction is instantly directed to a node with an active copy of the object required for the transaction.
Object Location Transparency	Applications do not need to know whether the objects they are accessing are located on local or remote machine.

Table 3: Data Protection Features of the Kabira High Availability Component

3.3 Incorporating HA Into an Application Using the High Availability Component

Any node configured to implement the High Availability Component can actively process work, as well as actively receive mirror backup copies of another node's transaction processes, producing very fast, efficient "active/passive" and "active/active" high availability service.

3.3.1.1 Failover

Configuration of primary and backup responsibilities is handled at the level of granularity of "object partitions," which are groups of mirrored objects. Each object partition has a primary and a backup node. These nodes are not bound by geographical or distance requirements and can be hundreds or thousands of miles apart from one another. When the primary node for an object partition fails, the backup node takes over the work of that partition. All pending and future transactions will be routed to the backup node.

The Kabira High Availability Component provides support for mirroring the application data to a backup node and for routing requests to specific nodes that are currently providing a service. This combination of mirrored application data and routing of request messages allows the backup node to take over the application service with virtually no downtime and no data loss.

3.4 Configuring for Continuous Availability: Four Models Using KTS

The following diagrams illustrate how Kabira's memory-resident processing and storage functions can be configured to interoperate between servers to instantly replicate data and applications.

3.4.1.1 Active/Active Configuration

Both servers are active; each can process transactions and receive continually-written, real-time copies of the other's data and applications in a separate section of its own RAM.

This method is inherently secure and provides unlimited backup and storage space and real-time replication facilities, due to large memory space in SMP storage servers. Both machines have takeover capability.

Data can be partitioned by type or instance across multiple servers. Transactions are routed to the correct server to be processed. As the volume of data or transactions grows, the workload can also be re-partitioned dynamically with no downtime. There are always at least two servers capable of taking over transaction processing from exactly the point of failure, and completing those transactions.

Upon failure, either machine assumes the role of the other active machine and finishes any transactions that may have been started in another node; these tasks are accomplished without data loss and transparently to users.

In Figure 3, Object A is mastered on Server 1. Object B is mastered on Server 2. Each object is backed up on the other server. In the event of a failure of Server 1, Objects A and B will be active on Server 2. Transactions for objects A and B will be routed to Server 2.

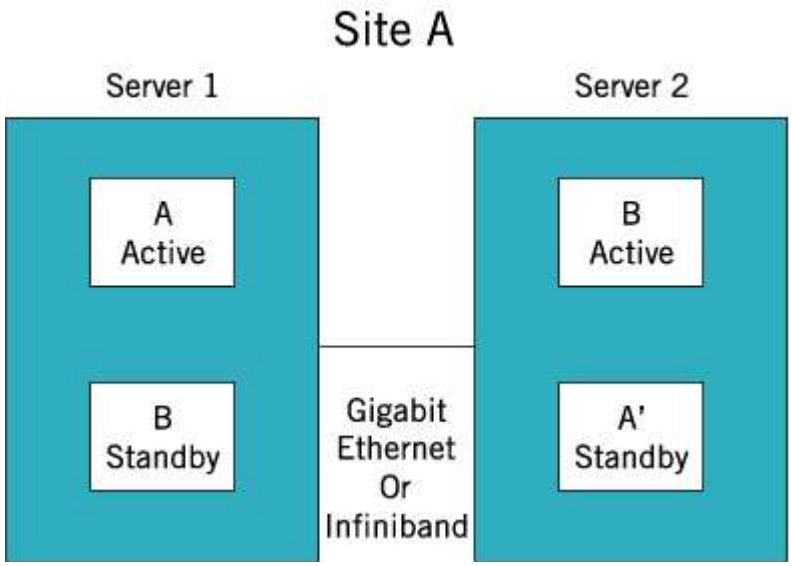

Figure 3: Active/Active Server Configuration

3.4.1.2 Active/Standby Configuration

One server is active as a transaction-processing server with memory space devoted to serving as a real-time receiver of data and applications. The other server is passive, acting as a "hot standby."

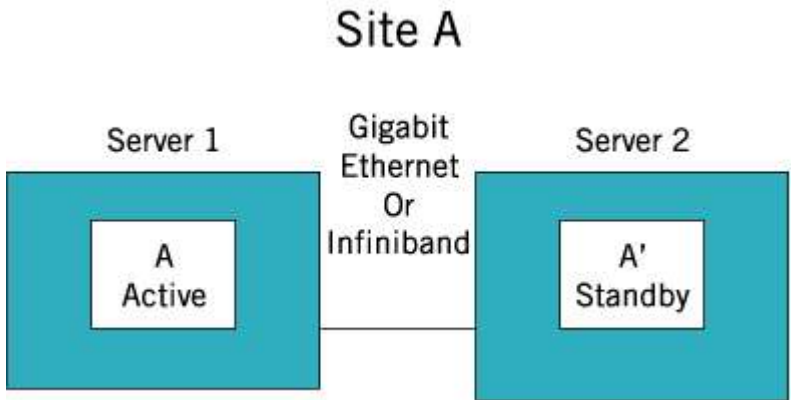

Figure 4: Active/Standby Server Configuration

As each transaction is processed, the hot standby server receives real-time data updates of all key application data. The replication supports high-speed, memory-to-memory copying over Ethernet or Infiniband. Any number of backup servers can be configured for maximum redundancy. The standby server can be locally attached for high-speed, low-latency updates, or geographically distributed for disaster recovery. There is no restriction on the distance

between servers. In the event of a failure, Server 2 will become active and take over processing all transactions for Server 1.

3.4.1.3 Memory and Non-Volatile Backup (Persist to Data Warehouse)

In this backup/storage configuration, one server is an active transaction-processing server containing data and applications in memory. A data warehouse provides transactional backup on disk in the event that all in-memory systems go down. Data is replicated to the second server for recovery.

The data warehouse provides non-volatile storage in the unlikely event that both servers fail. Data can be written to the data warehouse or file logging system synchronously or asynchronously. Through configuration, the system can be optimized for performance and reliability. The active, in-memory copy of the data is the authoritative source, not the data warehouse.

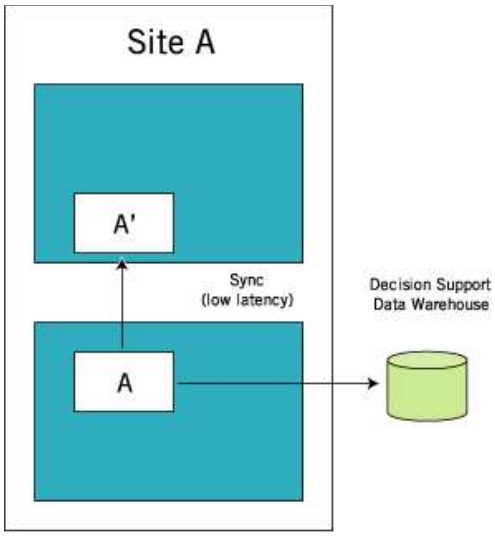

Figure 5: Memory and Non-Volatile Backup Configuration

3.4.1.4 High-Performance, Scalable Disaster Recovery Solution

Putting all the backup capabilities together results in an extremely high-performance, continuously available, geographically distributed grid-computing solution with disk backup. Data is backed up to memory and disk, both locally and remotely, for complete disaster recovery.

Instance-level data partitioning allows the addition of more servers and data centers to horizontally scale the solution. The solution can scale to support extremely large telecom and financial networks using geographic partitioning of data across nodes.

This system is engineered to solve the conflicting goals of data consistency, processing speed, and reliability. Routing all transactions to the single, authoritative source of the data enables data consistency. Transactions are replicated synchronously to a locally attached backup server for transactional reliability. Transactions are logged to a data warehouse asynchronously, providing a non-volatile backup without a significant effect on performance. This results in real-time disaster recovery without impacting performance.

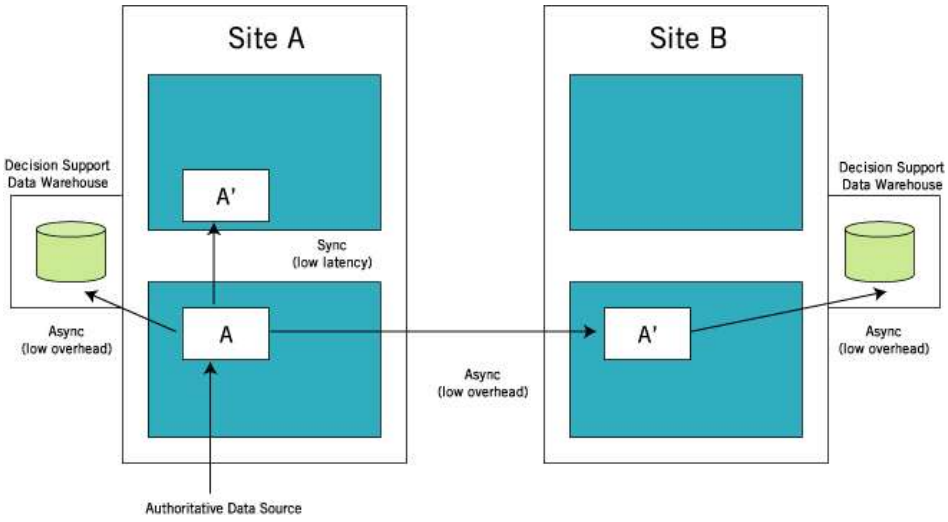

Figure 6: High-Performance, Scalable Disaster Recovery Configuration

3.5 Security Management

Continuous availability depends upon having a system that is secure and cannot be compromised, where all modifications are made only by authorized users. Security management is tightly integrated at the systems management and application development level. Sun and Kabira technologies combine to present a system of proven and ahead-of-the-curve security within every level of your enterprise's computer infrastructure, including:

- Hardware level
- Operating system level
- Application level
- Storage level

3.5.1 Systems Level Security Management

Solaris system security is designed to prevent intrusion and protect machine resources and devices from misuse. (See *Solaris 10 System Administrator Collection: System Security*, http://docs.sun.com/app/docs/doc/816-4557/6maosrjbb?a=view.) Security features also protect files from malicious modification or unintentional modification by users or intruders. Sun's security management elements include:

- **Hardware access:** Features that limit access to the PROM (programmable read-only memory chip), and restrict who can boot the system. See *SPARC: Controlling Access to System Hardware (Task Map)*: http://docs.sun.com/app/docs/doc/816-4557/6maosrjcq?a=view.

- **Resource access:** These are tools and strategies for maximizing the appropriate use of machine resources while minimizing the misuse of those resources. See *Controlling Access to Machine Resources*: http://docs.sun.com/app/docs/doc/816-4557/6maosrjbo?a=view.

- **Role-based access control (RBAC):** An architecture for creating special, restricted user accounts that permit users to perform specific administrative tasks. See *Role-Based Access Control (Overview)*: http://docs.sun.com/app/docs/doc/816-4557/6maosrjfi?a=view.

- **Privileges:** Discrete rights on processes to perform operations. These process rights are enforced in the kernel. See *Privileges (Overview):* http://docs.sun.com/app/docs/doc/816-4557/6maosrjfj?a=view.

- **Device management:** Provides additional protection for devices that are already protected by UNIX permissions. Device allocation controls access to peripheral devices, such as a microphone or CD-ROM drive. Upon deallocation, device-clean scripts can erase any data from the device. See *Controlling Access to Devices:* http://docs.sun.com/app/docs/doc/816-4557/6maosrjbn?a=view.

- **Basic Audit Reporting Tool (BART):** Provides a snapshot, called a manifest, of the file attributes of files on a system. Changes to files can be monitored to reduce security risks by comparing the manifests across systems or over time on one system. See *Using the Basic Audit Reporting Tool (Tasks):* http://docs.sun.com/app/docs/doc/816-4557/6maosrjds?a=view.

- **File permissions:** These attributes of a file or directory restrict the users and groups that are permitted to read, write, or execute a file, or search a directory. See *Controlling Access to Files (Tasks)*: http://docs.sun.com/app/docs/doc/816-4557/6maosrjea?a=view.

- **Security enhancement scripts:** Many system files and parameters can be adjusted to reduce security risks through the use of scripts. See *Using the Automated Security Enhancement Tool (Tasks)*: http://docs.sun.com/app/docs/doc/816-4557/6maosrjf4?a=view.

- **Solaris Zones partitioning technology:** Provides operating system-level virtualization services to isolate running applications and conduct workload resource allocation, notification, and monitoring.

- **Service Management Facility (SMF):** Provides options for booting the system, auto restarting services, and managing service objects, as well as for backup, restore, and rollback.

- **Solaris Fault Management Architecture:** Provides fault diagnosis rather than traditional ad-hoc error handling.

- **Authentication service implementations:** These include Solaris secure RPC, PAM, SASL, Secure Shell, Kerberos Service, IPsec, and smart card.

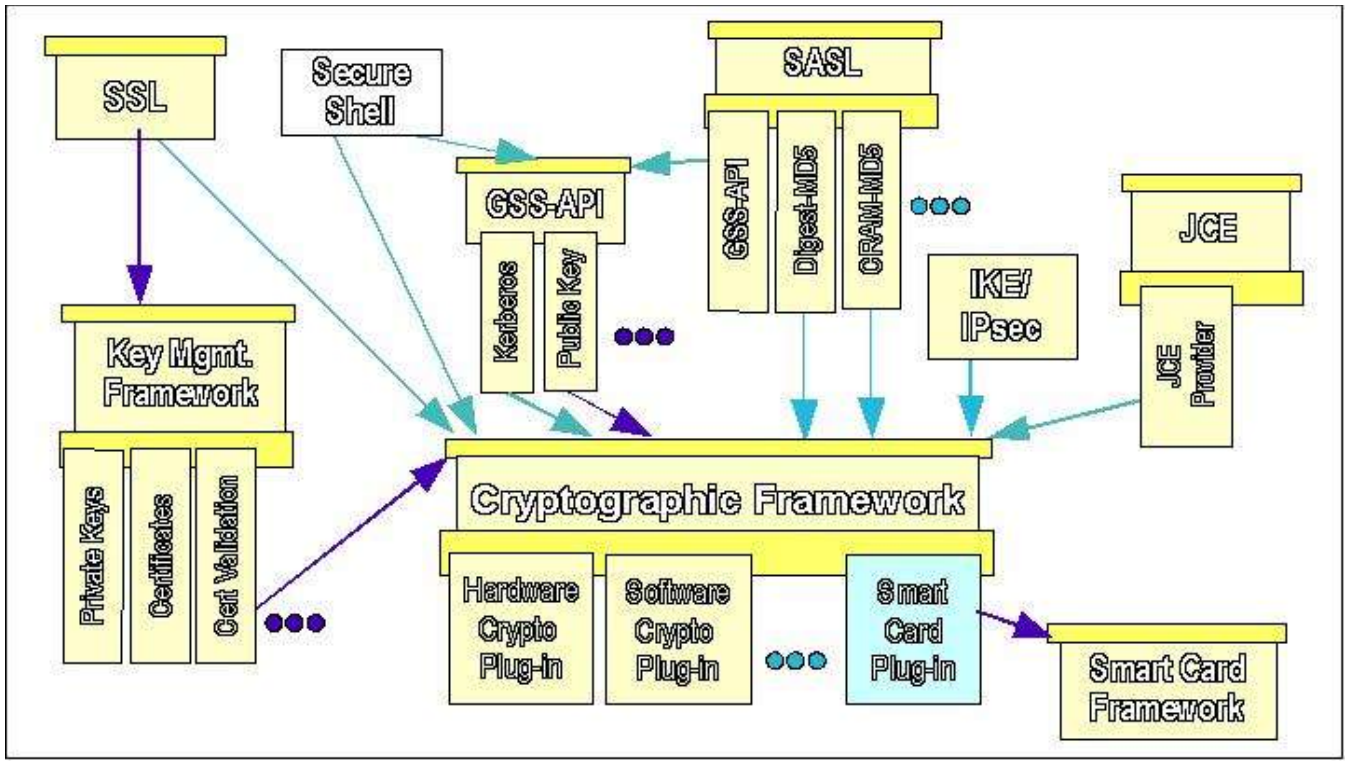

Figure 7: Solaris Security Architecture

3.5.1.1 Sun Managed Services

Sun Managed Security Services provide proactive security management and incident prevention through leading event correlation and analysis tools.

Sun's end-to-end managed security services include:

- Sun-Managed Firewall Services: These services focus on addressing your security requirements and business needs, and help to ensure that your firewall is in concert with those needs.
- Sun-Managed VPN Services: These services provide a design and deployment plan driven by your requirements. Technical staff employs industry-standard authentication and encryption technology.
- Sun-Managed Intrusion Detection Services: These services manage host and network-based intrusion detection systems (IDS) architectures. Alerts are issued in the event of a perceived intrusion. Alerts are actively monitored and tracked by certified security engineers who follow client-defined protocols for response, notification, and mitigation.

Key features of Sun Managed Security Services include:

- *Highly automated services*
- *Remotely delivered services*
- *Fixed price*
- *Monitoring and management of security devices*
- *Ongoing vulnerability assessments*
- *Incident response and remediation*

3.5.2 Security Management at the Application Level

Kabira provides complete platform and data security. The Switchadmin framework enforces security and access controls to prevent unauthorized configuration changes at the application level. The data is secure while in motion and at rest.

Kabira's in-memory architecture is designed to ensure that no temporary data is written to disk as data passes through. Data on the wire can be encrypted as required by application protocols. Data can also be encrypted when it is written to disk. Three-tier, database-oriented systems compromise data security by writing the data to disk.

Kabira's security infrastructure provides:

- Secure node administration (support for securing administrative access to nodes)
- Secure distributed applications (enabling programmers to develop and deploy secure distributed applications)

The Switchadmin function provides the interface for configuring, monitoring, and maintaining all applications developed and deployed on the application platform. The Switchadmin framework can enforce security and access controls on any operation. Audit logging is provided for all operator actions.

The Kabira Security Services Level (KSSL) core technology provides access to multiple, industry-standard security mechanisms. Users can develop applications for secure deployment in environments that may utilize a range of security technologies.

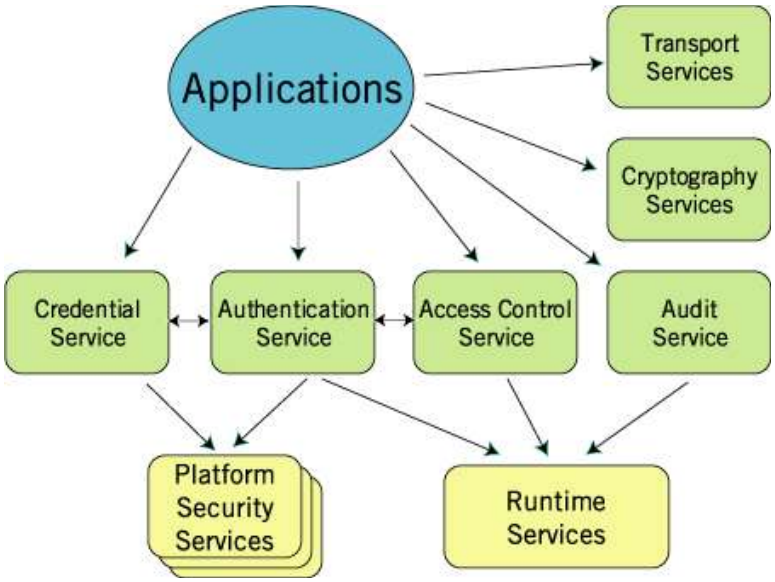

Figure 8: Kabira Security

KSSL provides the following services that enable development and deployment of secure applications on Kabira technology:

- Authentication services (token-based, username/password pairs, Public-Key-based authentication)
- Access control services
- Confidentiality (securing dialog between authenticated users)
- Security audit services
- Credential services
- Secure network communications
- Cryptography services
- Non-repudiation services (ensuring that a sender may not deny that a given message was sent)
- Replay detection (ensuring messages are not intercepted and replayed by an eavesdropper)
- Delegation (enabling the system to carry out tasks on behalf of a given principal, with assurance that the authentication will stay in effect as the work occurs within other parts of the system)

Every application developed and deployed on Kabira meets the same rigorous security requirements that have typically been demanded only of mainframes and networking systems. Kabira applications are reported to be more robust and run much more efficiently, since their security elements are integrated into the operations infrastructure level.

3.6 Application Testing

Kabira's design methodology helps avoid application faults by building technologies into the applications design platform that assist programmers in the definition, control, and management of a sound development process. Kabira's extreme programming methodology and automated nightly regression testing help developers program software quality into the application development process.

Extreme programming methodology dictates that the tests should be developed at the same time as the code. The earlier a bug is identified in the lifecycle, the lower the schedule impact and cost of fixing the bug. Kabira tools automatically perform the testing and measurement chores of the extreme programming methodology.

These easy-to-apply tools include unit-level white box testing, system-level function point testing, and performance testing, fully automating the testing lifecycle.

The biggest benefit of the methodology is realized when making changes to an existing system. Automated regression testing reduces the impact of minor changes made to a system. Faults are identified immediately. Developers can move to a rolling release cycle, with the system test phase being reduced from three months to two weeks. Automated testing early in the development lifecycle is much better than relying on a test team to perform the repetitive, difficult, and exhaustive checking required for error-detection in complex, high-performance systems.

3.6.1 Unique Characteristics of the Kabira Development Process Methodology

Developing applications on the Kabira platform includes the following processes, which are not typical of other development methodologies:

- **Exhaustive testing and measurement**: All test frameworks are script-driven and include frameworks for unit testing, code coverage measurement, leak testing, isolated testing of consumers, flows and channels, performance measurement, and regression tests.
- **Integrated documentation**: Component developers can use the Docsys facility to document components as they develop them. Docsys is similar to the Javadoc™ tool and is written in-line with the source code and installed with the binary code of the component package.
- **Service management**: In addition to application design, Kabira methodology includes design of administration interfaces and configuration data. This helps eliminate potential problems with these facilities, which can often pose larger and more complex problems than the applications themselves.
- **Traceability**: The system offers requirement tracking and a source control system for requirements traceability and test cases.

3.6.2 Recurrent Code Testing

The infrastructure platform is configured to test for regressions every night. This rigorous testing process includes building, deploying, and testing all components from the lowest level to the application level. The impact of a change at the lowest level can be immediately recognized at the application level. All outcomes are reported to the responsible developers.

Every line of code developed on the application platform is executed (both positive and negative execution paths, including the handling of fault or "unexpected" paths). According to Kabira, 100 percent code coverage on the code

base is achieved *every night*. Consider this in light of the fact that most IT projects exercise only 40 to 70 percent of their software *through their entire development lifecycle*.

Code-based tests include abort-handling verification, resource leak detection, and performance measurement.

3.6.3 Unit Testing and Measurements

The unit test framework isolates individual components and packages, such as message consumers, rules and classifiers, configuration handlers, or event notifier handlers. Each component is measured for correctness and stability. The unit test framework includes leak detection, deadlock detection, and abort testing. A test failure indication is made if any of these mechanisms detect a problem.

3.7 Application-Level System Design Standards

Design standards help ensure that developed solutions meet system requirements and are built according to best practices. Kabira tools help developers realize the best practices defined in the Kabira coding standards. These standards include a development methodology with specification, coding, and documentation standards. Reviews are included at key stages of the development process to ensure developers are following the process.

Design standards also include directives to ensure that systems meet availability and performance objectives. Proven design elements at the system level map how systems will act in various circumstances. Designers can determine application performance characteristics before deployment, using a combination of application models and load estimates. Kabira's system tools enable developers to characterize and optimize systems before deployment; determining system performance this way can help eliminate the possibilities of catastrophic failures. Instead, systems are configured to fail "gracefully," and Kabira provides the tools to size them.

Kabira application models provide a complete representation of the processing that will be performed by an application. Because Kabira's execution characteristics are known (including the impact of object operations on processing, memory and disk), the application execution is predictable and avoids problems from deadlock, race conditions, and data integrity.

Kabira design standards and tools can greatly reduce the number of defects discovered in product systems, leading to improvement in overall system availability.

3.8 Service Level Agreement (SLA) Management

The success of business increasingly hinges on efficiently delivering predictable service levels to satisfy customer profitability. Thus, good resource management is essential to service level management. Availability management is achieved by defining, measuring, and managing standards of system performance as they relate to the system available time desired by users. The computer system must be able to measure, monitor, and control these activities to guarantee agreed-upon levels of service expected by the user.

Availability management also includes planning for changes and maintenance on the system, designing for acceptable standards of service delivery under both normal and undesirable conditions, and tracking and reporting of events.

The system can be engineered to create and maintain a certain level of service needed by end users. The agreement between a service provider and end users that defines the level of service delivery is a Service Level Agreement.

A system is defined as “available” if it is doing its designated work at the expected rate. System unavailability is divided into two main categories: scheduled downtime and unscheduled downtime. For both categories, availability management at the system level includes measuring uptime and downtime, identifying and categorizing planned and unplanned downtimes, and reporting system performance in comparison to targeted performance.

Each system platform has a set of available measurement tools to gather data across the system resource hierarchy. Collected information can be extracted into reports in data and graphical form to expose exceptions, trends, and details about selected resources.

Availability objectives are defined in terms of the following:

- Outage frequency, duration, and service hours
- Maximum number of outages allowable per time interval
- Minimum and maximum response times for each application
- Mean time to repair and recovery time

3.8.1 SLA Management in Systems Management

The resource management solution in the Solaris OS provides a flexible response to the varying workloads that are generated by different applications on a system, and it enables major system resources such as CPUs and VM. Resource management also facilitates customer resource utilization in order to achieve agreed-upon service level more effectively. In addition, Solaris bandwidth management enables network bandwidth usage allocation and monitoring. Both resource management and network bandwidth management help ensure that the Solaris OS can deliver more predictable service levels in the production environment.

Sun Service Availability Manager is optional software, which extends the functionality of Sun’s existing management solution by monitoring and measuring the availability and performance of all major network services. Sun Service Availability Manager enables customers to test the availability and quality of application service levels from the same common Solaris Management Console used to manage and monitor the systems themselves. These enhancements allow service providers and other customers to optimize performance of Sun systems, reducing systems management costs while increasing service levels.

The Solaris OS supports IP Quality of Service (IPQoS). IPQoS enables system administrators to provide different levels of network service to customers and to critical applications.

Sun also offers a complete set of managed services for solutions that can be custom-configured to help improve and manage a solution IT environment. With Sun, it is possible to choose what to manage, and for how long. Whether short- or long-term, service assessments are done on a function-by-function basis and goals are agreed upon up front, backing them with SLAs. Engagements encompass product, people, and process options, across security, storage, operations management, and more — from the smallest to the most complex solution.

3.8.2 SLA Management in the Application Infrastructure

Kabira's High Availability Component generates data that can be used by any existing third-party management tools installed on a legacy enterprise system for application tracking and reporting.

The High Availability Component performance tools make it possible to measure the application latency of specific methods, which will identify hot spots in the application. Transaction statistics enable the identification of data contention and deadlocks, which would prevent platform scaling on multi-CPU systems.
Event suppression can be set to suppress events when they exceed *x* number of the same event.

3.9 Change Management

Uncontrolled system changes are a frequent cause of system outages. A company must be able to build, test, and deploy new versions of mission-critical applications as often as necessary, while relying on continuous systems and transaction operations. Change management is the process of ensuring system stability during major system changes. It is also necessary to be able to change system configuration and to add or delete hardware devices from the running system without disrupting it.

Change management capabilities ensure that changes are made in a controlled manner with the possibility of backing out unsuccessful changes. Change management enables:

- Deploying a common configuration across multiple servers with high availability
- Online upgrade of service logic and configuration while the system is running
- Loading multiple configurations concurrently with a single active configuration
- Rollback to last known good configuration in the event of service disruption from loading a new configuration
- Logging of configuration changes to help trace system faults back to configuration changes
- Access control to enable only authorized administrators to make configuration changes
- Loading sets and running sets to ensure that configurations are loaded as a known set to avoid loading configurations that have not been tested together

3.9.1 Change Management in Systems Management

Change management consists of the following automated steps:

- Collect and record all change requests.
- Prioritize and group the requests based upon business and technical assessments.
- Test and prepare the implementation, and check that recovery procedures are in place.
- Schedule, defer, or reject the requests.
- Review and approve the requests.
- Implement changes.
- Control and report the status of the requests.

3.9.2 Sun Management Connection Delivery Framework

The Sun Management Connection delivery framework provides an integrated service delivery platform that gives users full visibility into their IT systems and into the underlying business processes that these systems support.

ControlPoint provides access to device status, performance reports, trouble tickets, change management, and inventory and systems configuration information in-context. As a portal to all services, ControlPoint delivers hundreds of on-demand management reports for every monitored device in the system infrastructure.

ControlPoint provides these functions:

- **Information Monitoring:** At the highest level, ControlPoint provides a snapshot of the overall system health including all web, device, and application transactions. The severity and type of transaction incidents immediately become apparent. Click on the transaction to drill down on fault details, showing the details of the transaction.

- **Change Control:** Sun's "best practices" change management process can significantly increase service quality by reducing errors. Users submit change requests via a simple form that includes sections for implementation, tests, and back-up plans. A change ticket number identifies each request, which allows both user and the Sun team to track the status of submitted requests.

- **Inventory Control:** ControlPoint provides a detailed inventory of all devices that make up your systems and applications, including firewalls, IDS, load balancers, routers, servers, and switches. The inventory is searchable via a simple form or a drill-down list categorized by device type. For each device, users can access a detailed list of information, including:

    - Service level
    - Hardware and software configuration (version and patch/service pack)
    - Graphical view of location (cage, rack, slot)
    - Past, current, and planned change tickets
    - Open and closed incident tickets

- **Incident Control and Analysis:** This incident management process provides Sun's engineers with comprehensive procedures for efficiently addressing, notifying and resolving incidents. Using ControlBase's remediation procedures, Sun engineers can resolve the majority of incidents within minutes, at the first line of support. By resolving incidents and gathering performance data from the environment, Sun gains insight into the user's systems and analyzes them for optimization. Users can access this information through ControlPoint. The ControlPoint Incident Control screen shows you all open, resolved, or closed incident tickets. Click on a ticket ID number to see specific details of its working history including category, type, location, and severity. If you subscribe to the monitoring service, you can update incident tickets directly with remediation details.

ControlPoint also allows users to organize and search through all incident and change tickets by defining parameters for the tickets you would like to see.

- **On-Demand Reporting:** ControlPoint provides access to hundreds of trending and status reports about applications, networks, and servers. Reports are generated "on-demand" and rendered as easy-to-read graphs based on user selections. ControlPoint reports include:

    - Incident and change ticket progress
    - Monitoring results
    - Exception and service reports
    - Change management coordination
    - Inventory management information
    - Account management information
    - Execution of user/administrative tasks

3.9.3 Change Management in Application Infrastructure

Kabira integrates software configuration management with an automated solution for managing distributed deployments, so that a company can continue to roll out new releases efficiently and effectively, reducing both cost and risk. This is key to optimizing cycle times for software updates.

Change management tools ensure that only authorized people are allowed to make changes, and that the changes will be applied in a consistent manner. This avoids potential service disruptions caused by user errors or invalid configurations.

Kabira's High Availability Component offers these capabilities:

- Facilitates restoring or upgrading the application, Kabira system software, operating system, or hardware.
- Features a high availability data migration capability, which permits a server to be taken offline and reconfigured after migrating data to a backup server. During that time, the backup server handles new transactions. Once the original server has been restored, the data can be transparently migrated back to the original server. The HA router configuration is updated automatically to indicate that the appropriate server is able to accept new transactions.
- Permits business logic changes. You can build a new model and bring it online from a running system.
- Provides online configuration flows and online variable configuration, giving users the capability to load up configuration data while online.

3.9.4 Non-Disruptive Online Changes

A number of reconfiguration tasks can be controlled at the application level in order to maintain, repair or upgrade elements of a system without needing to shut down or restart.

3.9.4.1 Dynamic Hardware Reconfiguration

A dynamic reconfiguration allows an installation to add, delete and modify:

- Processors
- Storage
- Buses or channels
- Processors
- Control units
- I/O devices
- Power, cooling

3.9.4.2 Dynamic Storage Reconfiguration

Dynamic storage reconfiguration allows you to add or take away parts of the central or expanded storage in certain increments. These parts of the storage must have been previously defined as reconfigurable, and are brought online and offline via system commands. This feature is also useful in balancing storage between logical systems in response to a change in workload requirements.

3.9.4.3 Dynamic Processor Reconfiguration

Dynamic processor reconfiguration makes it possible to remove and bring back physical processors without interrupting the running system. For example, a physical processor that was previously removed for repair or part replacement can be put back online into the configuration without an outage to the running system. This feature dramatically reduces planned outages and helps avoid unplanned outages caused by hardware defects.

3.9.4.4 Dynamic I/O Reconfiguration

Dynamic I/O reconfiguration is the ability to select a new I/O configuration without having to perform a system restart. This greatly enhances overall system availability by allowing you to eliminate the scheduled outages that were previously necessary:

- When a new I/O was added
- When paths were changed
- When I/Os were deleted permanently
- When maintenance was necessary

Non-disruptive change management at the application level gives enterprise server customers the latitude to modify and configure real-time applications on the fly without bringing down mission-critical systems. A user can add and remove nodes and additional servers, and migrate application data from server to server, all while the system continues to run.

High availability failover and restore capabilities enable an IT department to take servers offline for upgrade of operating systems, platform software, and application infrastructure software with no interruption in service. The servers can be taken offline with the data migrated to the backup server. When the platform is brought back online, the data and transactions can be migrated back to the upgraded server without data loss.

3.9.4.5 Data Partition Management

Application objects are partitioned into object partitions to load balance the application work across one or more nodes. Objects can be partitioned by instance or by type to enable geographic partitioning. Partitions can be migrated from one node to the other for maintenance.

Non-disruptive changes and maintenance capabilities include:

- Online redefinition for applications
- Redundancy/modularity
- Granularity/isolation (the property of an application to be independent from any problems occurring outside its domain)
- Pre-provisioning: New application versions, channels, or sets of configuration data can be tested on the production system before activating them

3.9.4.6 Stand-in Processing Alternatives

Applications are designed and data is placed so that host access is not required to provide service; this allows users to keep running a subset of the functions of an application while the host (or the connection to the host) is out of service.

3.9.4.7 Exploit Subsystem Recovery Features

As with subsystem availability attributes, application-level design utilizes existing recovery and restart features inherent in the High Availability Component. This includes checkpointing and logging features. Using high-speed memory to recover and copy, these features reduce the time needed to resume service.

3.9.4.8 Non-Disruptive Maintenance for Software Upgrades and Migration

Designing for non-disruptive maintenance gives IT professionals the ability to modify the coding of applications without interrupting the service they provide. Redundancy/modularity and isolation features of KTS contribute to this ability. Multiple versions of channels and flow specifications can be loaded in shared memory and new versions activated without application downtime. Only one instance is active at one time. Flow specification is audited for correctness before being activated.

3.9.4.9 Application Reuse Without Source Code Changes

Applications can be reconfigured to simultaneously support multiple network protocols (or protocol versions), transports, and business logic – without source code changes – by adding or changing configuration files for channels, or by changing properties of consumers and rules in the flow specification files. Thus, components can be configured for different scenarios without altering the component, or even needing to understand its inner workings.

3.9.4.10 Co-existence

Co-existence allows multiple versions of the same program or function to run concurrently. This is particularly useful during application upgrade processing, where you need to run two different versions of a program at the same time.

Multi-versioning support also eliminates the outage required to fall back to a previous version if problems arise with modules that were newly introduced into your production environment.

3.9.4.11 Online Redefinition of Applications

Loading multiple versions of configuration objects into shared memory enables online re-configuration of applications, and new versions can be activated without application downtime. The loaded version undergoes an application-specific audit for correctness before it can be activated. When configuration has been successfully loaded and audited, the specified configuration version can be activated; the previous version is retained in shared memory and can be activated as a fallback.

3.9.4.12 Pre-provisioning

Pre-provisioning reduces risk in software upgrades or migration. During pre-provisioning, a new software version or set of configuration data is loaded onto a production system and tested, but is not activated for live traffic.

4 Summary

Sun Enterprise servers and Kabira Transaction Switch combine to provide continuous availability from the ground up, starting with Sun's underlying hardware and extending to the native, fault-tolerant functionality integrated into the Kabira High Availability Component.

The solution from Sun and Kabira goes beyond the challenge of high availability to demonstrate a continuously available, low-latency system using open system hardware and software. With applications processed and stored in memory, this enterprise computing system is designed to offer unmatched speed, reliability, availability and security.