White paper
Fujitsu Software openUTM Fit4Cluster

System downtimes are not acceptable for applications deployed in enterprise-critical situations. Fujitsu Software openUTM supports high availability via the UTM cluster functionality according to Fujitsu's *Dynamic Infrastructures* strategy.

Content

Introduction

System downtimes are not acceptable for applications deployed in enterprise-critical situations. Top priorities are high application availability and automatic workload distribution during peak load times; these are also an important feature of dynamic infrastructures. Web applications must be available 24x7, leaving no time for maintenance slots or version upgrades.
Stand-alone UTM applications usually do not meet these requirements, especially with regard to server downtime or failure of a running UTM application, or when a software version is being upgraded.

Cluster as solution

openUTM supports high availability via the UTM cluster functionality, and is part of Fujitsu's *Dynamic Infrastructures* strategy. A cluster consist of a number of similar computers and operating systems (cluster nodes), which are connected via a fast network and share common peripherals. A UTM application can be operated on a cluster as a UTM cluster application.
An automatic load balancing option via LPAP bundles for LU6.1 and OSI TP connections as well as UPIC load balancing for UPIC clients is available for client requests addressed to applications within a UTM cluster application. It is also possible to use external load balancers.

Differences between UTM stand-alone and UTM cluster applications

Stand-alone UTM application

In a stand-alone configuration, the UTM application runs on several similar processes (*tasks* in BS2000) on a single server; see Figure 1. Client requests are managed in a queue and fetched from the queue by UTM for processing. The application's program units run on the UTM processes in BS2000. All processes in the UTM application can access process-specific local data as well as global data held in memory (i.e. UTM storage areas GSSB and ULS, BS2000 *shared memory* or *memory pools* and shared memory reserved on UNIX systems via `mmap`). Separate synchronization mechanisms must be implemented for the storage areas that are not managed by UTM. Larger data volumes are typically held in an external shared data management system (e.g. a database), which can be accessed by all program units.
A stand-alone UTM application is resilient against individual UTM processes or the UTM program units running on them aborting. However, the application will no longer be available if the server on which it runs fails or if the entire UTM application is terminated normally or aborted.

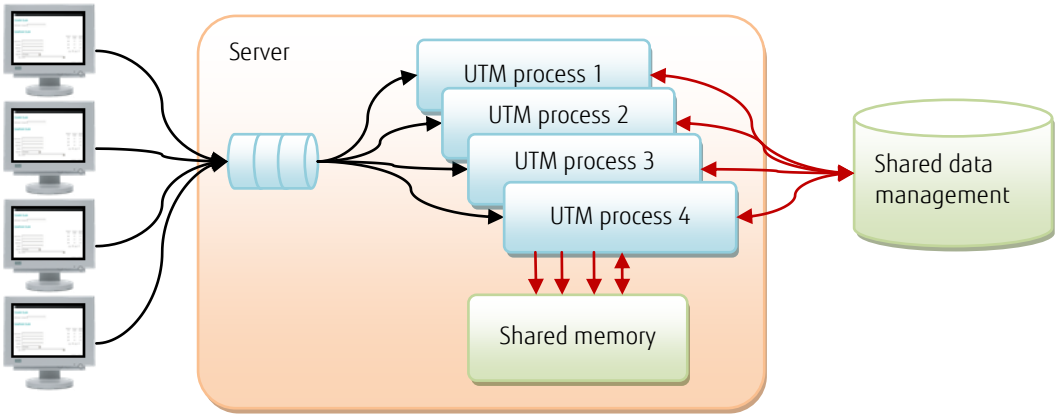

Figure 1: Stand-alone application

UTM cluster application

In a UTM cluster application, the same UTM application runs on several homogenous servers (cluster nodes) running the same operating system. Thus UTM processes are distributed on these cluster nodes and each node runs the same program units; see Figure 2. Of course it is no longer possible to support globally shared data held in memory; any shared data must be held on an external medium. For the storage areas GSSB and ULS managed by UTM, UTM V6.1 ensures that they can be used globally in the cluster. A separate section at the end of this White Paper considers details and specifics of shared data storage.
New conditions apply also to the management of shared data files. As all nodes involved require access to the data, it must be stored on *Network Attached Storage* (NAS) or a *Shared Pubset* and the accesses synchronized via appropriate network protocols (e.g. *Network File System*, NFS or *Cross Coupled System*, XCS). This means that ISAM files and LEASY cannot be supported in BS2000, because the ISAM *shared update* synchronization of XCS pubsets is not supported across multiple computers. Solutions for other databases are described below.
The clients should be connected to the application via a load balancer, which selects a node and then establishes a connection to that node. Client requests are placed in the queue on the respective node and fetched by the UTM processes running on that node. As the same UTM application with the same program units runs on all nodes, the UTM cluster application acts like a single UTM application.
A UTM cluster application will continue to be available even if a node fails. The load is then distributed to other nodes. Details about high availability and load distribution are provided below.

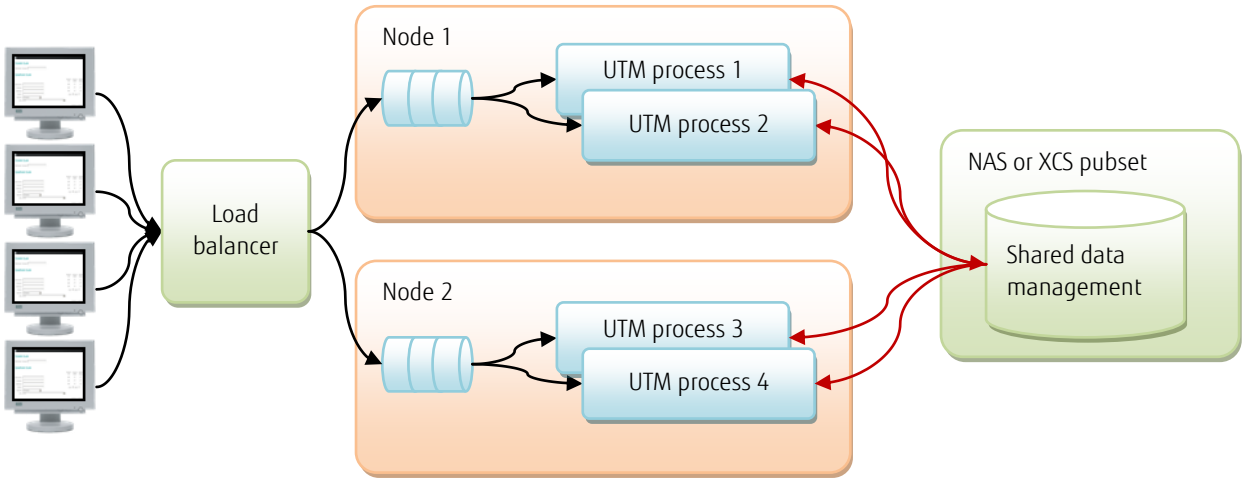

Figure 2: Cluster application

Global administration

A UTM cluster application should behave like a single UTM application, and this requirement is supported by the UTM administration functions. Administrative actions which affect the cluster application as a whole are executed globally, i.e. not only on the node application where the action was initiated but also on all other node applications. This also applies to user administration, such as creating new users or changing passwords. Current settings on existing nodes are propagated to any nodes subsequently added to the cluster. Administrative actions that refer to a single node, e.g. power down and power up, will obviously not affect any other nodes.

Databases

SESAM and UDS in BS2000

In BS2000, SESAM/SQL and UDS/SQL databases are typically configured so that the *Database Handler* (DBH) runs on the same processor as the UTM application. With SESAM and UDS it is possible and common practice to distribute the data with respect to their contents on several databases, e.g. on a customer database and on an article database, because several databases can be processed by the same DBH. If the databases are used in a UTM cluster application, they can be distributed over the cluster nodes, where each node can access its own databases as well as the remote databases of other nodes, see Figure 3. This is achieved by installing a database-specific DB distribution component (SESDCN for SESAM or UDS-D for UDS) on each node which has remote access to the DBH to provide transactional security for modifying remote database accesses. Obviously with this configuration access to remote databases is slightly slower than access to local databases.

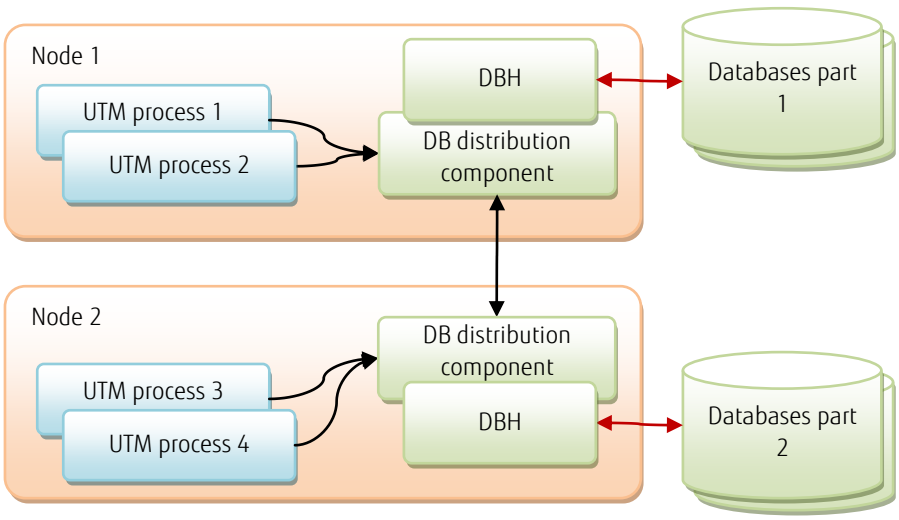

Figure 3: Distribution of databases

Oracle® RAC

The openUTM cluster concept harmonizes well with the Real Application Cluster (RAC) from Oracle®. Figure 4 below shows the recommended configuration: a primary RAC node is associated with each UTM node. Additionally, each UTM node uses the other RAC nodes as a fallback stage for failover. The node application communicates with a DB distribution component which maintains a list `tnsnames.ora` of the connected RAC nodes.

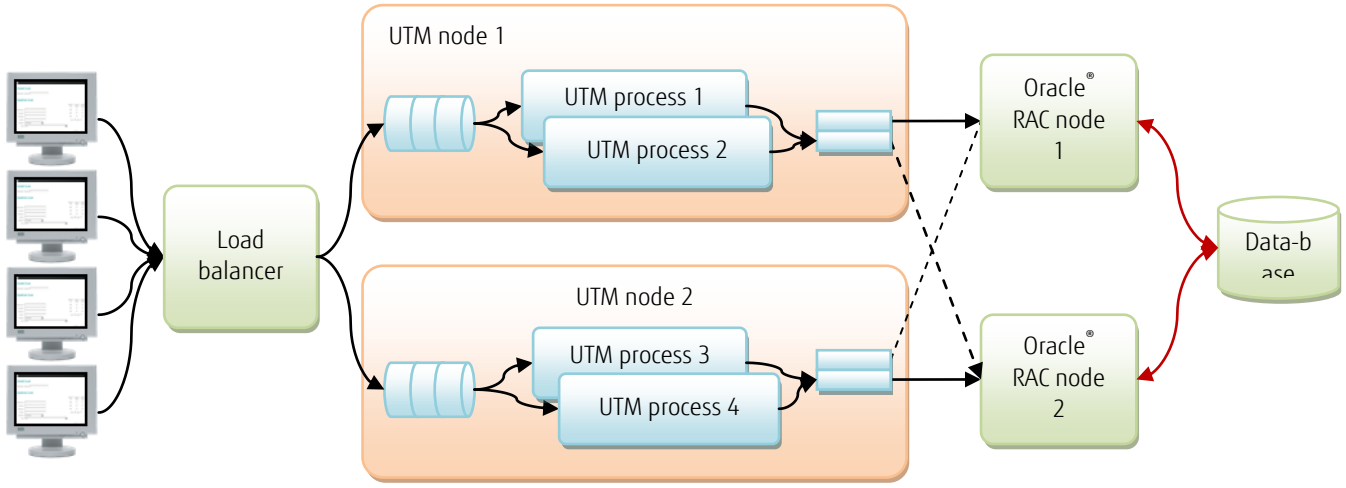

Figure 4: UTM cluster and Oracle RAC

Load balancing

Load balancing with UPIC

The UPIC client software includes a remote load balancer; see Figure 5. At sign-on, the UPIC client randomly selects one node application from a list of available node applications and sets up a connection to this node application, which is maintained until the client disconnects. With the next connection request, a node is again selected randomly. As all UPIC clients behave in the same way, the result is a statistically uniform load distribution across the available node applications. A static weighting can be assigned to the individual nodes: for example, if the nodes differ with respect to their performance, more powerful nodes can be listed several times in the lists of node applications and will statistically be selected more frequently for connection set-up.

If the node selection hits a failing node application, an attempt is made to establish a connection with another node application. When a node application fails and an existing connection breaks down, a new connection can be established to another node application that has not failed.

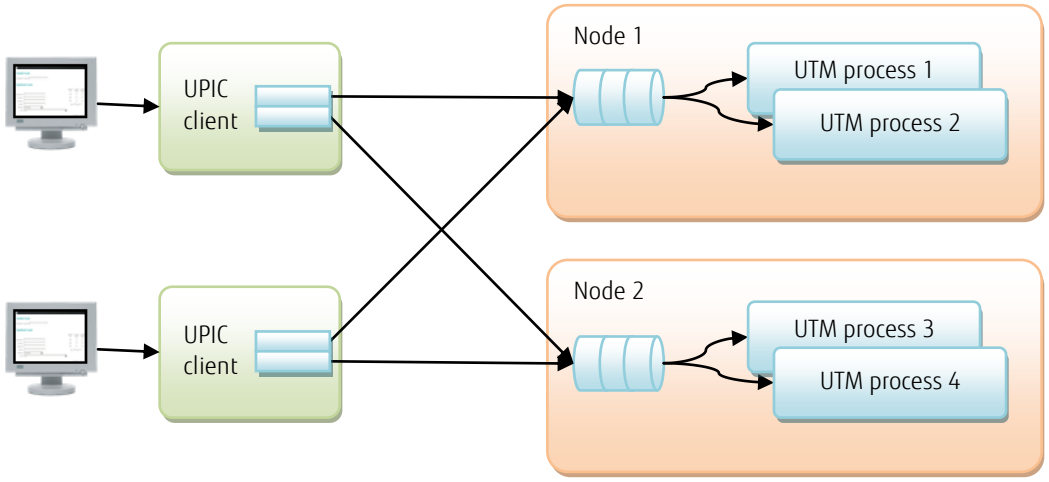

Figure 5: Laod balancing with UPIC clients

Load balancing between partner applications using a LPAP bundle

When a UTM application 1 communicates via LU6.1 or OSI-TP with a partner application 2, and the partner application is implemented as a UTM cluster application, then load balancing can be implemented using a LPAP bundle; see Figure 6. Application 1 addresses a master LPAP, which selects a slave LPAP that is connected to one of the nodes of the partner (cluster) application 2. The master LPAP has a list of all relevant slave LPAPs. If the slave LPAP selection hits a failing node application, an attempt is made to establish a connection with the next slave LPAP.

To avoid a single point of failure, application 1 can also be configured as a cluster application. A configuration of this kind makes sense if, for example, the UTM application offers a web interface which is implemented in cluster application 1 while the underlying business application runs in cluster application 2.

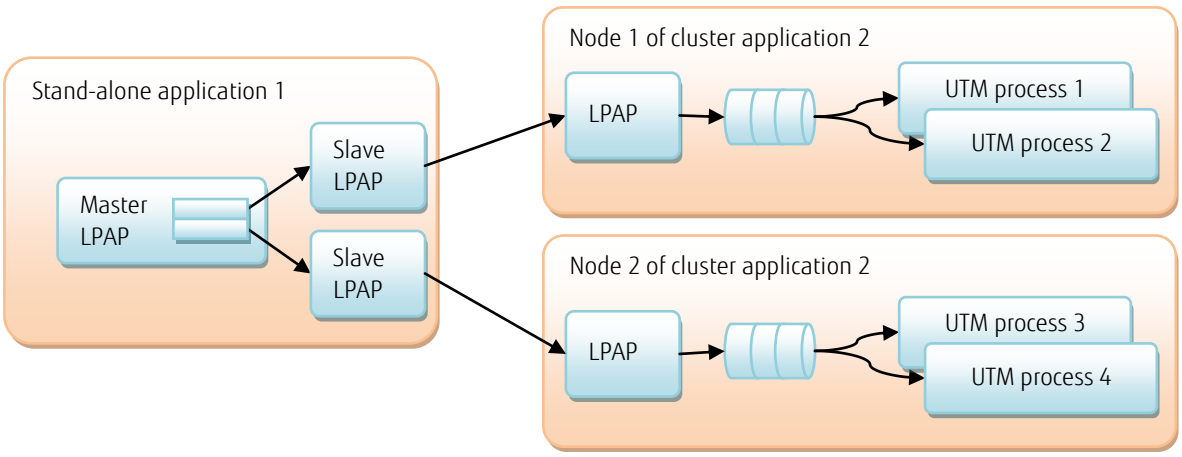

Figure 6: Load balancing with LPAP bundle

Figure 7shows a symmetric partnership between two cluster applications. In this constellation, both cluster applications can act as service user or service provider, with the initiative for communication coming alternately from a node in application 1 or from a node in application 2.

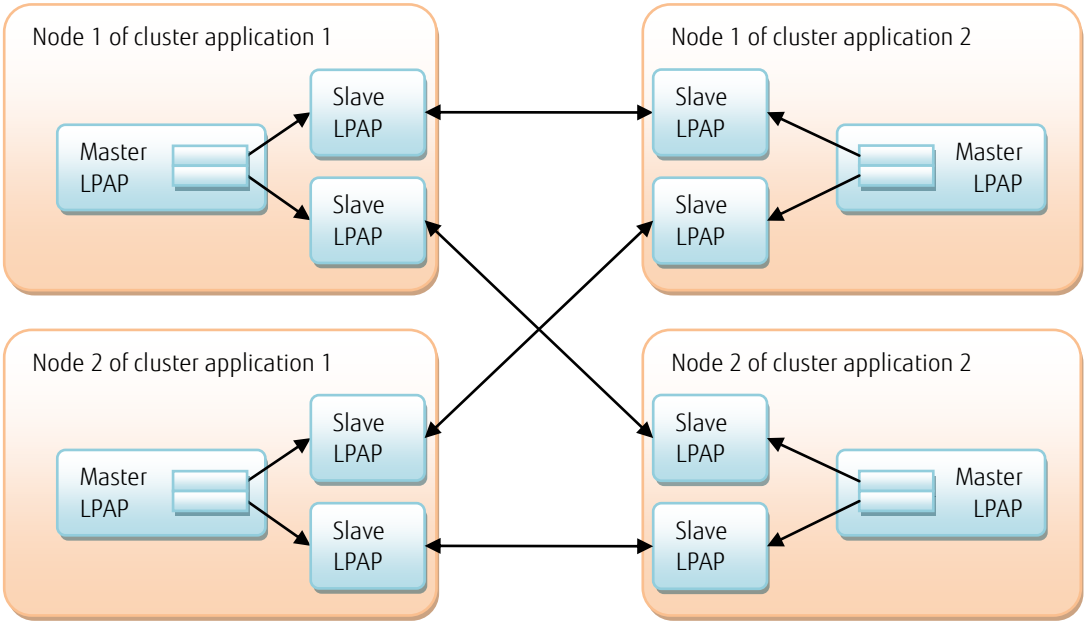

Figure 7: Symmetric load balancing with LPAP bundle

External load balancer

When using an external load balancer clients issue their requests using a virtual IP address which the load balancer converts to the IP address or MAC address of the selected node; see Figure 8. Responses are also routed through the load balancer, which replaces the IP address of the responder with the virtual IP address and then forwards the response to the client.

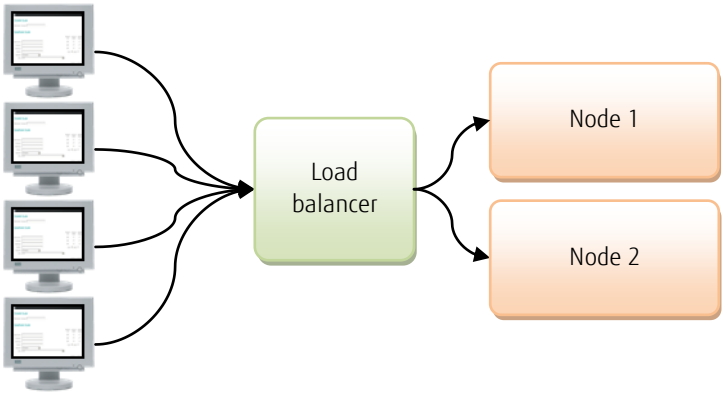

Figure 8: External load balancer

A client may initiate an administrative action that needs to be applied globally (e.g. changing a user password) on a single node selected by the load balancer; openUTM will automatically propagate these administrative actions to all the other nodes. Generally, however, administrative actions are carried out (e.g. via WinAdmin) on a node selected by the administrator and not by the load balancer. This means that it is possible to issue global administrative commands that are propagated and executed automatically by UTM on all nodes, as well as commands that are only effective locally on the selected node (e.g. power down and power up).
An external load balancer can be a software solution on a front-end processor, e.g. *Linux Virtual Server* (LVS); an off-the-shelf *redirector* switched between clients and servers can also act as load balancer.

Static load balancing based on location

In some situations it can be beneficial to handle load balancing and database distribution statically based on location, and to adapt the application accordingly. For instance, if customer-related data is held in a database, the database, the node applications, and their associated clients can be distributed by region so that the normal access to local data in the respective region is fast. However, a client and node in one region will also be able to access data in another region, although access to the remote databases will take longer. Figure 9 shows a configuration of this kind with BS2000 and distributed SESAM databases.

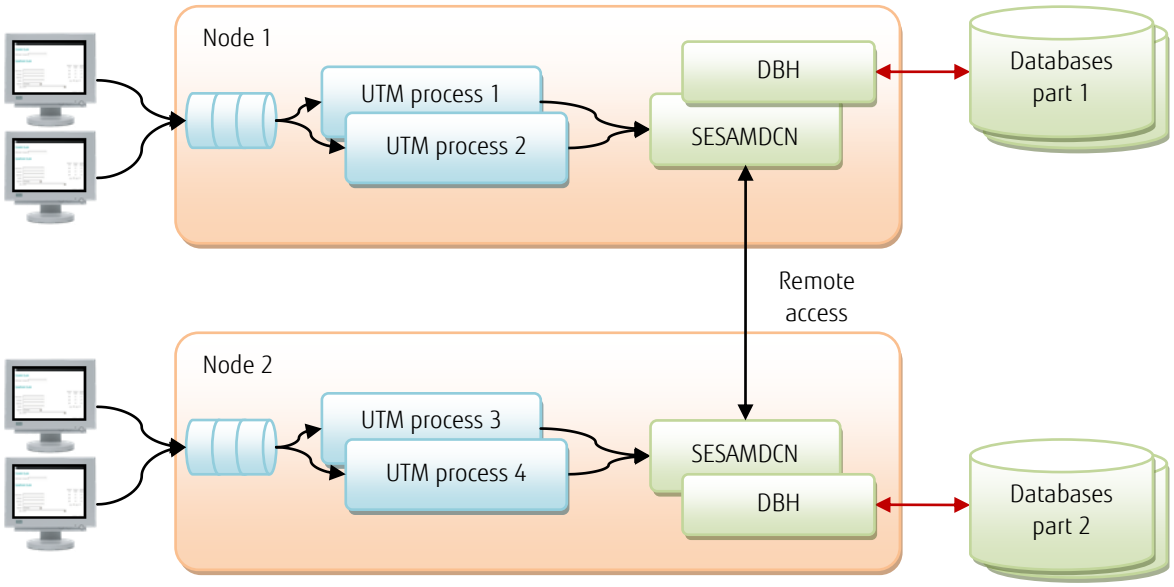

Figure 9: Static load balancing according to location principles

Manual load balancing

A UTM cluster application can also be run without automatic load balancing. In this case, the client chooses the node to which he wishes to connect manually during the sign-on process. It is the responsibility of the client to detect a high workload on the node and, if necessary, to attempt to establish a connection to another node with a smaller workload.
Even with manual load balancing, UTM cluster applications are better in terms of high availability because it is quicker – when a node fails – to sign-on again to another node that is already running than to wait until a system administrator has solved the problem on a stand-alone application (e.g. powering up the failed processor and restarting the UTM application and, if necessary, taking additional repair measures).

High availability

Ring monitoring

No additional software is required to monitor the UTM cluster application. The node applications monitor each other; each node monitors one of its neighbors; see **Fehler! Verweisquelle konnte nicht gefunden werden.**. When a node application fails, it is possible to trigger follow-up actions by a script or command procedure automatically called on the monitoring node. Examples for possible follow-up actions are: automatic check (via `ping`) whether the entire node processor or only the node application has failed; automatic remote restart attempt for a failed node application, automatic remote reboot attempt of a failed node processor, automatic temporary removal of the failed node from the configuration list of an external load balancer, information for a system administrator or operator via automatically issued e-mail or SMS. Further, if remote restart fails too, automatic node recovery can be invoked to release locks possibly still held by the failed node application. Failed nodes are removed automatically from ring monitoring by openUTM; nodes powered up again are automatically reinstated in ring monitoring. The UPIC load balancer automatically ignores failed nodes but takes them into consideration again as soon as they are available.

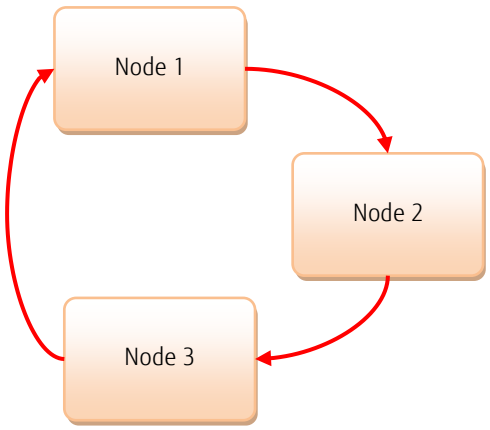

Figure 10: Ring monitoring

Renewed sign-on after node failure

If the connection of a client to a node application gets lost, the user can attempt a new sign-on; in this case the original session context will be lost. With `MULTI-SIGNON = NO` he would normally wait for the period of one attempt to automatically warm-start the node application. If the warm-start fails, openUTM will call a script or command procedure provided by the application developer which may administratively sign-off all users still signed-on to the failed node application.

A UPIC load balancer on the client side will automatically ignore the failed node application and establish a connection to a different node. The same applies to UTM partner applications where the master LPAP ignores the failed node applications when distributing the workload. If an external load balancer is in use, the affected node should be removed from its configuration list when the node application fails (see *Ring monitoring*), so that this node is ignored for any subsequent sign-on attempts. This too can be done by the script or command procedure called by openUTM.

Online maintenance

Software upgrade

Stand-alone applications already support the replacement of application programs without requiring the entire application to be shut down. The replacement is carried out on a process-by-process basis; this means that – during a transition period – some UTM processes will still run with the old application program while some UTM processes will already be using the new application program.

A UTM cluster application allows the node-specific version upgrade of UTM or of the underlying operating system. The node application on a node is shut down, the software upgrade is carried out on this node and then the node application is started again. For a node application to be shut down properly, all active users of the UTM cluster application must sign off. However, the users can sign on again immediately after the affected node application has been shut down and – if an external load balancer is in use – removed from the distribution list; the user will then be connected to one of the other node applications. After the software upgrade has been carried out and the node application has been started up, it can be re-entered into the load balancer's distribution list and is then immediately available again.

Replacing hardware or adding new nodes

The procedure for hardware replacement on a specific node is initially the same as for a software upgrade. UTM can be informed of the node's new configuration via the administration interface before the node with the new hardware is powered up. When defining a new UTM cluster application reserve nodes should be included in the generation; these reserve nodes can be reconfigured for specific new hardware at a later stage as required and added to the UTM cluster application without needing to shut it down. All the global administrative changes made to the UTM cluster application since the initial generation (e.g. the list of users and their passwords) will be applied automatically to the new nodes and they will be included in ring monitoring. Entering the new nodes in the load balancer's distribution lists makes them available for immediate use.

Online import

Asynchronous requests held in the queue of a node application that is being shut down for maintenance purposes can be moved to other node applications and executed there – all under administrative control. This guarantees that these requests will not be lost.

Shared memory

Shared program memory

UTM program units provided in memory shared by all UTM processes (in BS2000 e.g. *shared memory* or *memory pools*) in a stand-alone application can continue to be used if the UTM application is converted into a cluster application. The program units must be installed on all nodes. The node-specific installations can then still be provided in shared memory (e.g. *memory pools*) on each individual node.

Shared data memory

Multiple UTM program units and/or UTM processes may use shared data held in main memory or on external files. The following sections look at some selected examples and the implications when converting a stand-alone application into a UTM cluster application. The final section examines the general case.

Connection-local memory

Many applications manage connection-specific data in shared memory that can be accessed by several program units (in BS2000 e.g. in *memory pools*). This concept can be retained unchanged in a stand-alone UTM application converted into a UTM cluster application so long as the connection-related data exists only for the duration of the connection and does not need to be accessed after the connection is cleared down. Each node then manages its own connection-specific shared data memory. As every connection is associated with only one node, the connection-related shared data memory is also only used locally on one node.

Read-only access to data that is mostly constant

Many applications hold constant global data in shared memory areas or files (e.g. application-specific configuration values), so that this data can be accessed by all program units in the UTM application. This data is modified only when the application programs are updated, but not while the application is running.
This concept can also be retained as there is no synchronization requirement when accessing constant data. This data must be installed on all nodes together with the program units or initialized in main memory as part of application start-up.

The general case

In general, global data (which may be held in e.g. files, LEASY databases, memory pools, or in AREAs), which can be read or written by all program units across conversation or connection boundaries, will have to be stored on external memory and accesses to it must be synchronized across nodes. Either openUTM storage areas GSSB and ULS or database tables are best suited for this purpose as openUTM or the database system will synchronize access to the data.
From openUTM V6.1 on storage areas GSSB and ULS act cluster globally, i.e. all node applications have the same view of their contents, and openUTM ensures comprehensive access synchronization.

Disaster recovery

Where high availability support and the ability to react to the failure of a node application are not enough, a fully-fledged disaster recovery system may be required, in which standby servers and peripherals are located in a second data center at a remote site. If a catastrophic failure occurs and the first data center fails completely, system operations are relocated to the standby data center. The disk storage units operated in the first data center are constantly mirrored in the standby data center so that the standby data center is able to take over system operations at any time using the current data; see Figure 11. In a disaster situation, the servers and applications in the standby data center power up and work continues on the mirrored disk storage units. The switchover can normally be automated (e.g. on BS2000 with HIPLEX-AF).

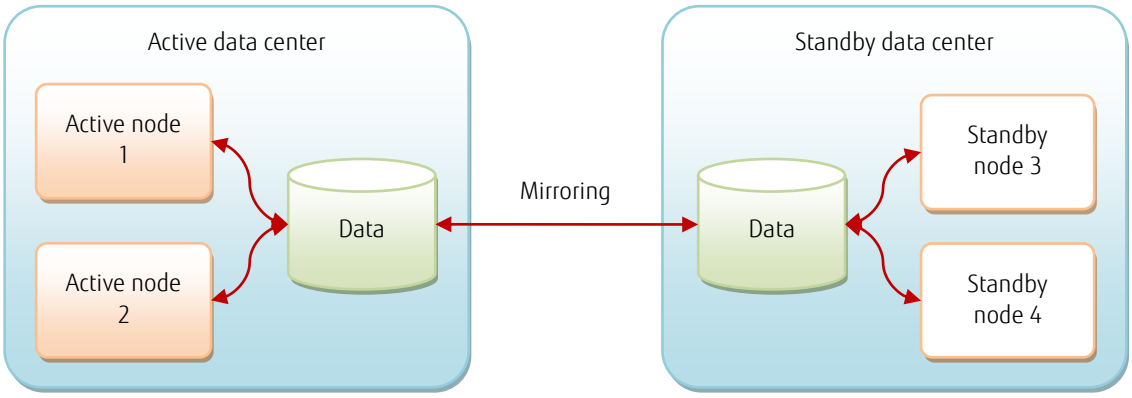

Figure 11: Spatially separate active and standby data center

For performance reasons, active UTM cluster nodes should be connected with each other via a fast network. Peripherals are also shared. This means that the nodes must be located close to each other, usually in the same data center. However, the UTM cluster concept still fits neatly into the Disaster Recovery concept described above:
The servers in the first data center and the servers in the remote second data center are configured as UTM cluster nodes for the same UTM application. Both data centers have their servers and local peripherals connected to each other via a fast local network.
n the standard situation only the node applications in the first data center are started and running, while the node applications in the standby data center are inactive, i.e. on standby; all active cluster nodes and their shared peripherals are connected via a fast local network to ensure good performance.
A disaster situation triggers a switchover to the nodes in the standby data center, which are also connected to each other and to the mirrored peripherals via a fast network. During the switchover, the nodes in the standby data center are powered up and the node applications started. If some node applications in the failed first data center should still be running, they are shut down, leaving only the nodes in the standby data center active. This guarantees good performance also after the switchover.

All the administrative changes made to the UTM cluster application since its initial generation in the first data center are backed up in the mirrored administration journal and are automatically propagated and applied to the node applications started in the standby data center. Once this process has been completed, the application is available for users to sign on.

Conclusion

openUTM supports high availability via the UTM cluster functionality according to Fujitsu's *Dynamic Infrastructures* strategy. A cluster consist of a number of similar computers and operating systems (cluster nodes), which are connected via a fast network and share common peripherals. A UTM application can be operated on a cluster as a UTM cluster application.

Contact
FUJITSU Technology Solutions GmbH
Address: Mies-van-der-Rohe-Str. 8, 80807 München, Germany
E-mail: openSEAS@ts.fujitsu.com
Website: www.fujitsu.com/de
2015-03-30