



White Paper

Deploying Highly Available

IBM Rational ClearCase On Linux

Abstract

Linux has become a primary platform for both hardware and software development organizations. Protection of software assets being managed on Linux development platforms is a critical function of corporate IT. IBM Rational ClearCase provides a complete set of tools for managing the complete development lifecycle for software products. Deploying a highly available ClearCase environment provides the benefits of a powerful integrated development environment with the assurance of protection against loss of data or productivity that can result from planned and unplanned outages.

Table of Contents

Introduction	3
Configuring Rational ClearCase on Linux	3
The Importance of Protecting Software Assets	4
ClearCase High Availability	5
High Availability Configurations.....	7
Active/Standby Configuration with Local Shared storage	7
Active/Standby Configuration with Network Attached Storage	8
Active/Active Configuration with Local Shared Storage	9
Active/Standby Configuration with Replicated Storage	10
Configuration Considerations to Ensure High Availability.....	11
Configuration Steps for LifeKeeper High Availability	12
Summary	13
References.....	14

Introduction

In the past five years, Linux® has gained wide support from major corporations such as IBM, Oracle and Novell for use in business-critical environments and is now a Tier-1 operating system at every major server hardware and software provider (with the notable exception of Microsoft). Linux has also emerged as a platform of choice among IT shops due to its scalability, cost-efficient hardware choices and command set similarity to well-known UNIX environments. Software development tasks which have been hosted on UNIX platforms are migrating to Linux to leverage these advantages as well as the emerging set of next generation development platforms which are being developed by the Linux community, such as the GNU toolset and a host of JAVA-based development, build and debug tools.

IBM Rational® ClearCase® is a leading solution used by development organizations as they leverage Linux to build, integrate, modernize and deploy software projects. Rational support for Linux is currently focused in two areas: facilitating the development of Linux applications through comprehensive IDE support and reducing the total cost of ownership for software development infrastructure by providing Linux based platform support for managing software assets. With support for all of the roles and activities within the software life cycle, IBM Rational products allow organizations to perform all aspects of product development on Linux with the software asset management capabilities provided by Rational ClearCase supporting teams of any size, from small, co-located teams to globally distributed teams working in parallel. The solutions are available on Linux running on diverse platforms from PCs to mainframe computers.

ClearCase provides complete life cycle management and control for software development projects. Through integrated version control, automated workspace management, parallel development support, baseline management, and build and release management, Rational ClearCase provides the capabilities needed to create, update, build, deliver, reuse and maintain business-critical assets. Rational ClearCase can help increase productivity through parallel development, reduced build/release cycle times, and increased software reuse. Integration with leading IDEs including Rational Application Developer, WebSphere Studio, Microsoft Visual Studio .NET and the open source Eclipse framework further streamlines development.

Local, remote and web interfaces for Rational ClearCase enable access at anytime and from virtually anywhere. Support for development on Linux, Windows, Unix and mainframe (z/OS) systems provides a broad range of choices for application and build environments and allows organizations moving to Linux to easily migrate from existing platforms and receive all benefits of a ClearCase development environment.

Configuring Rational ClearCase on Linux

The IBM document, [ClearCase Install Guide for Linux](#), provides detailed instructions on installation within a Linux environment. Rational supports both 2.4 and 2.6 kernel versions as distributed via RedHat, RHEL 3 and 4, and Novell SUSE, SLES 8 and 9. Due to differences in the kernels on which the Linux distributions are based, the steps required to configure the kernel for ClearCase installation, primarily to rebuild the Linux vnode layer, must be followed based on the specific Linux distribution used.

ClearCase on Linux supports platforms built on Intel x86, AMD64, Xeon64bit/EM64T, s390 and POWER chipsets. While all IBM BladeCenter, eServer, xSeries and pSeries systems

which are certified with Linux are supported with Rational ClearCase 2003.06.x, not all systems are supported on all Linux distributions. Refer to [Supported ClearCase Family Releases](#) for specifics on the combinations available.

IBM xSeries platforms such as the x460, x366, x346, and x306 combined with Linux and Rational ClearCase provide a highly scalable, reliable and affordable solution for application development in organizations ranging from the Fortune 500 to sole proprietorships.

The Importance of Protecting Software Assets

Software assets are development organizations' most precious resources. As such, the source code, development environment, build tools and the resulting binaries which make up these assets must be protected. Protection against data loss and against downtime resulting from both planned and unplanned outages is critical to an organization's continued productivity, success and longevity.

There are numerous ways to backup and protect the information you have stored in a ClearCase repository on Linux. Using standard backup to tape archive procedures, you can store away information securely so that in the event of a serious failure, you can use these archives to restore your repositories. However tape-based restoration can take significant time and will result in loss of all data since the last backup was taken.

If ClearCase availability is critical to your organization, using an approach which includes real-time continuous data protection and/or high availability clustering to maintain a redundant replica enables you to restore normal operations faster and, if using data replication for continuous data protection, a catastrophic outage will cause lost data only since the last replication (typically only minutes) instead of the last backup (typically the previous day or before). A high availability solution can result in dramatic decreases in lost ClearCase data and dramatic increases in developer productivity.

This white paper explains how you can build a highly available ClearCase environment to enable quick switching to a backup server to prevent end user downtime in the event of either planned or unplanned outages of components in the ClearCase architecture. SteelEye Technology's LifeKeeper Protection Suite for ClearCase is shown as one example of a high-available clustering solution that can be used to build a redundant environment. Other solutions for providing ClearCase high availability on Linux exist, but none have specific monitoring and recovery modules built for ClearCase as exist in LifeKeeper.

ClearCase High Availability

Rational ClearCase is made up of several independent yet interacting components as represented in the drawing below. While these are shown as separate physical servers, the services are often consolidated in a ClearCase configuration. So, a single server can host one or more of the functions depicted.

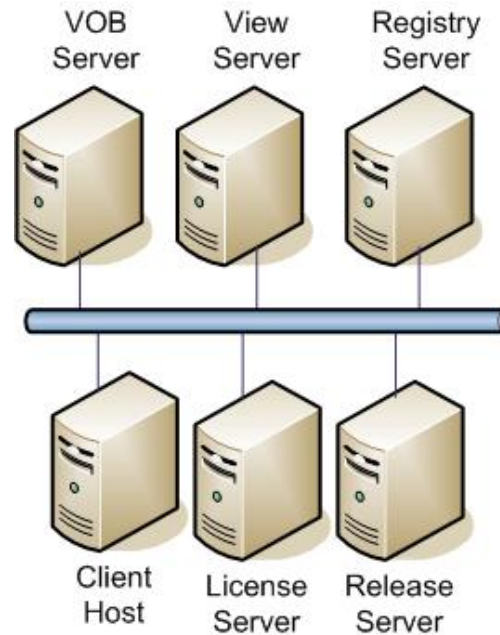


Figure 1: Rational ClearCase Components

Rational ClearCase requires several network wide resources: a license server host (one or more), a registry server host (1 plus an optional backup) and a release host. Additionally, the configuration can contain a number of client hosts and server hosts. The function of each of these critical system components and the requirements for high availability protection is described below:

License server host – Most functions in ClearCase must have a valid license to perform the requested task. The ClearCase license server performs the role of allocating a license when requested by a ClearCase command. It also ensures that the maximum number of licenses allowed as defined in the license.db file is not exceeded. ClearCase allows multiple license servers to prevent license server access from being a single point of failure.

Registry server host – The ClearCase registry server contains databases used to determine the locations of ClearCase data structures. This information is stored within the server host in files in the directory `/var/adm/rational/clearcase/rgy`. This server must also be highly available and can be protected by LifeKeeper to ensure its availability.

Release host – The Release host serves as the network-wide housing area that provides storage for the entire ClearCase product distribution including executable files, configuration files, and online documentation. This host is essentially a file server. ClearCase provides a link install type, which creates symbolic links to the

Release Area via NFS mount points. Thus, it must be highly available if an install with links has been selected.

Client host – A Client host is a system that runs the ClearCase client programs which constitute the user-level interface to ClearCase. This interface includes cleartool, checkout, etc. A Client Host by itself does not require high availability protection.

Server host – A Server host stores ClearCase data such as VOB and View storage directories. ClearCase server processes execute on these hosts, as needed, to communicate with client programs through remote procedure calls. Because of the storage directories and server processes, high availability protection is required on these hosts to manage NFS resource for local storage access and IP resources for storage on NAS devices.

Whether running on individual dedicated servers or grouped onto a fewer number of machines, the ClearCase registry server, release (area) server, license server and VOB and View storage servers represent potential single points of failure and need to be protected in a high availability cluster configuration.

Providing a mechanism to recover protected ClearCase services from a failed primary server onto a backup server and to detect failures at both the server level via a heartbeat mechanism and at the individual ClearCase resource level by monitoring the ClearCase daemons, is a primary function of high availability clustering software. The remainder of this document presents several highly available configurations that can be deployed for Rational ClearCase on Linux.

High Availability Configurations

This section contains definitions and examples of several typical highly available ClearCase configurations which can be built using SteelEye LifeKeeper for Linux.

Active/Standby Configuration with Local Shared storage

In this Active/Standby configuration, *NodeA* is the primary ClearCase server providing ClearCase Registry, Release Area, and storage locations for Views (the system is a View server). All storage resides on a shared array between the cluster servers. High Availability clustering software provides monitoring of the ClearCase functions and recovery as needed. *NodeB* acts as a hot-standby for ClearCase while also running other applications/services.

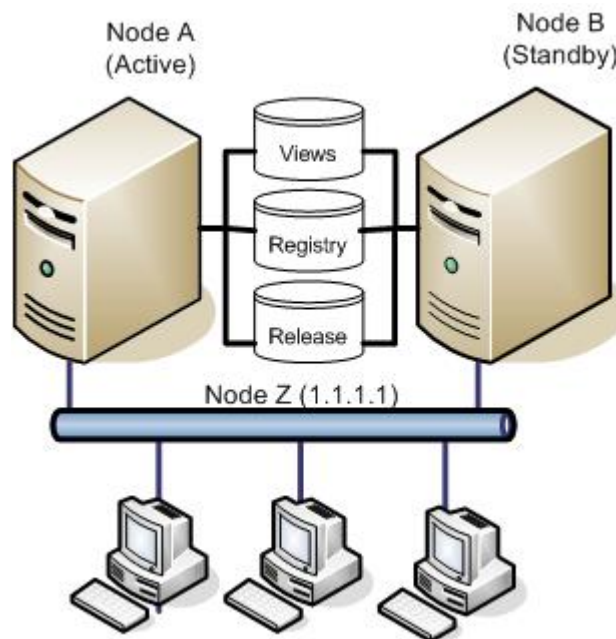


Figure 2: Active/Standby Local Shared Storage Configuration

Configuration Notes

- The clients connect to the ClearCase servers using the DNS entry *nodeZ* designated to float between the servers in the cluster. The name uses the protected IP address (1.1.1.1).
- The *rgy_hosts.conf* file on all ClearCase systems is set to *nodeZ*.
- The *alternate_hostnames* file on *nodeA* (the primary server) contains the following entries: *nodeA* and *nodeZ* (one name per line in the file).
- The *alternate_hostnames* file on *nodeB* (the backup server) contains the following entries: *nodeB* and *nodeZ*.

Active/Standby Configuration with Network Attached Storage

In the Active/Standby configuration, *nodeA* is the primary LifeKeeper Server. It protects storage locations for VOBs. All storage is located on a NAS device. While *nodeB* may be handling other applications/services, its role in the ClearCase configuration is acting as a backup for the ClearCase resources.

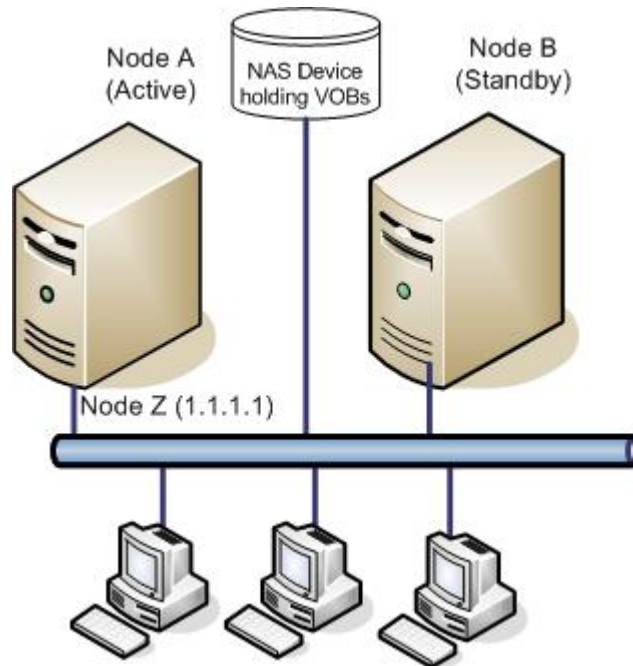


Figure 3: Active/Standby with Network Attached Storage Configuration

Configuration Notes

- The clients connect to the ClearCase servers using the DNS entry *nodeZ* designated to float between the servers in the cluster. The name uses the protected IP address 1.1.1.1.
- The *rgy_hosts.conf* file on all ClearCase systems is set to *nodeZ*.
- The *alternate_hostnames* file on *nodeA* contains the following entries: *nodeA* and *nodeZ*.
- The *alternate_hostnames* file on *nodeB* contains the following entries: *nodeB* and *nodeZ*.

Active/Active Configuration with Local Shared Storage

In the Active/Active configuration below, both *nodeA* and *nodeB* are primary LifeKeeper servers for ClearCase resources. Each server is also the backup server for the other. In this example *nodeA* protects the shared storage array for Views and the Registry as the primary server. *nodeB* protects the shared storage array for VOBs as the primary server. Additionally, each server acts the backup for the other, which in this example means that *nodeB* is the backup for the protected View storage and Registry on *nodeA*, and *nodeA* is the backup for the protected VOB storage on *nodeB*.

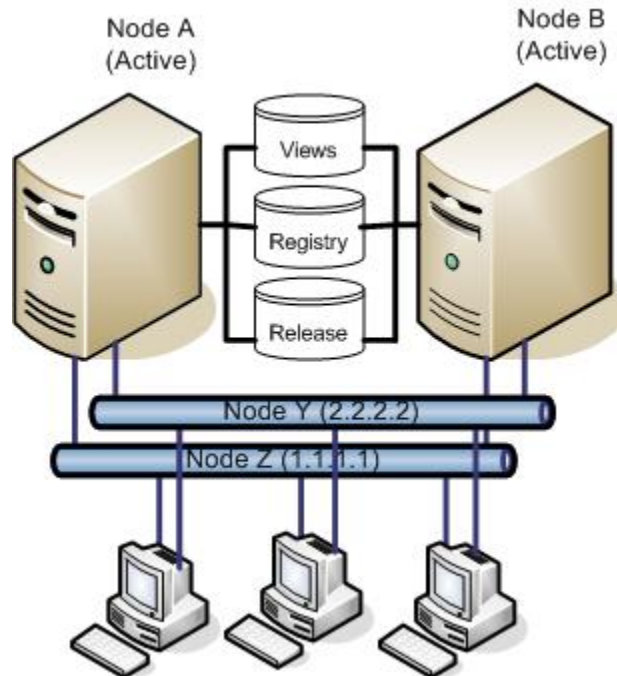


Figure 4: Active/Active Local Shared Storage Configuration

Configuration Notes

- The clients connect to the ClearCase Servers using the DNS entries *nodeZ* and *nodeY* designated to float between the servers in the cluster. The names use protected IP addresses 1.1.1.1 and 2.2.2.2, respectively.
- The *rgy_hosts.conf* file on all ClearCase systems is set to *nodeZ*.
- The *alternate_hostnames* file on *nodeA* contains the following entries: *nodeA*, *nodeY*, and *nodeZ*.
- The *alternate_hostnames* file on *nodeB* contains the following entries: *nodeB*, *nodeY* and *nodeZ*.
- When the ClearCase resources for *nodeA* are brought in-service on *nodeB*, ClearCase must be stopped and restarted for the **albd_server** process to understand it is now providing access to the Registry.

Active/Standby Configuration with Replicated Storage

In the Active/Standby configuration below, *nodeA* serves as the primary ClearCase server providing ClearCase Registry, Release Area, and storage locations for Views (the system is a View server). All ClearCase data is made available to the standby server, *nodeB*, via data replication. As data is written to the local disks on *nodeA*, it is automatically replicated to *nodeB*'s disks. High availability clustering software provides monitoring of the ClearCase functions on *nodeA* along with recovery as needed. *NodeB* acts as a hot-standby for ClearCase while also running other applications/services.

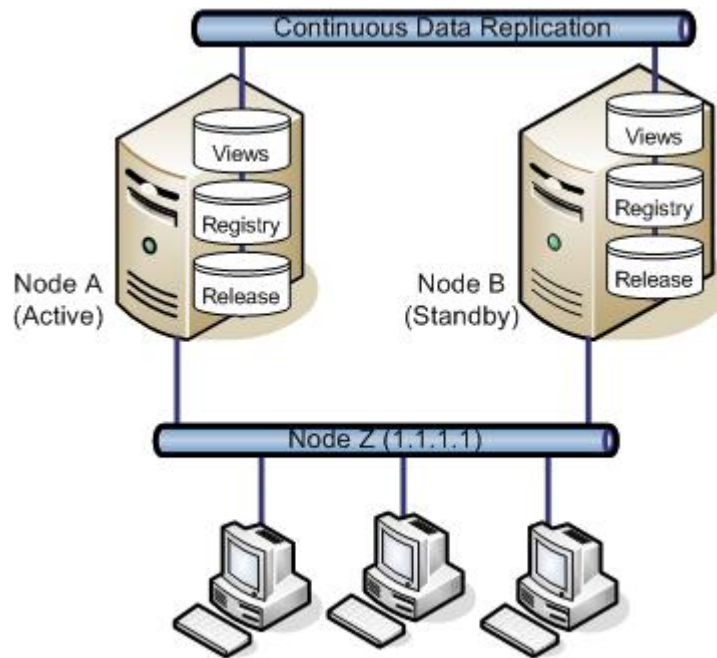


Figure 5: Active/Standby Replicated Storage Configuration

Configuration Notes

- The clients connect to the ClearCase servers using the DNS entry *nodeZ* designated to float between the servers in the cluster. The name uses the protected IP address 1.1.1.1.
- The *rgy_hosts.conf* file on all ClearCase systems is set to *nodeZ*.
- The *alternate_hostnames* file on *nodeA* contains the following entries: *nodeA* and *nodeZ*.
- The *alternate_hostnames* file on *nodeB* contains the following entries: *nodeB* and *nodeZ*.

Configuration Considerations to Ensure High Availability

There are a number of considerations when building a ClearCase high availability configuration. These are required to ensure that the path to critical ClearCase resources (VOB storage, Registry, Release Area) are highly available via the use of virtual IP addresses and an associated system name in DNS. ClearCase services are provided by a number of daemon processes. The main daemon processes are **albd_server** and **lockmgr**, which must be monitored and protected in a high availability configuration. To provide a highly available ClearCase Application, a number of configuration changes are required.

The need for these changes can be illustrated via the following example. A standard ClearCase installation requires a host to be designated as the Registry server, *nodeR*. When ClearCase is installed on *nodeR*, a number of files are created in the directory */var/adm/rational/clearcase/rgy* that act as the repository of information for the operation of ClearCase. When other systems are installed, they are setup to contact *nodeR* via the setting in */var/adm/rational/clearcase/rgy/rgy_hosts.conf*. If *nodeR* fails, then the ClearCase Registry is no longer accessible and the client cannot perform any ClearCase related commands, such as checking out a file for edit. If a backup Registry was defined then someone with root access must switch the client machine to point to the backup. If *nodeR* is clustered with another server, *nodeB*, then access to the Registry from a client can be made highly available via the following configuration changes

1. Place the directory */var/adm/rational/clearcase/rgy* on a shared disk (via the use of shared storage, NAS devices or via data replication) so that either system in the cluster can have access to the data.
2. Designate a name (*nodeZ*) and IP address (1.1.1.1) that can float between *nodeR* and *nodeB* (the name and address must be in DNS).
3. Define an IP resource to represent this new floating (or virtual) IP address and create a file system resource that represents the Registry directory. Both of these new resources will be monitored and recovered by the high availability software subsystem.
4. Edit the *rgy_hosts.conf* file on the client and alter the name from *nodeR* to *nodeZ*. Perform the same edit on the Registry server.

Now if *nodeR* fails, the HA software will detect the failure and migrate both the IP and file system resource to *nodeB* so that any client accessing the Registry on *nodeZ* via IP address 1.1.1.1 will now find it transparently on *nodeB* without any intervention on the part of an administrator and without disruption in access to the ClearCase services.

A similar configuration change will be required for VOB and View storage locations. When creating a VOB or View with storage on a local system you have the option of specifying the host, global path to the storage area and the local path to the storage area or letting ClearCase determine it. If you let ClearCase determine that information, it creates paths using the system name (from **uname -n** output). For a cluster with names of *nodeA* and *nodeB* creating storage on *nodeA* and letting ClearCase determine values will lead to a host name of *nodeA* and paths starting as */net/nodeA/...*

As was done in the Registry example above, we need to use a system name and an IP address in DNS for the host name and path entries during creation of the storage area that can float between the servers in the cluster. So if *nodeZ* with IP address 2.2.2.2 is designated to float between servers in the cluster the VOB creation would use **-host nodeZ**, **-gpath /net/nodeZ...** and **-hpath /net/nodeZ...** to allow the storage to float between the servers in the cluster. Note that the storage area must reside on storage that each server in the cluster can access either through the use of shared storage, via a NAS

device or through data replication. To limit the configuration changes to the ClearCase Registry for an existing ClearCase installation that has correctly defined paths, the current server name could be used for the virtual name which would then require a change to the actual system server name. For example, if **uname -n** currently returns "nodeA", make "nodeA" the virtual name and simply change the real name to return "nodeZ" on a **uname -n** call.

Using floating server names requires the use of the ClearCase `alternate_hostnames` file located in `/var/adm/rational/clearcase/config` directory. In this file are listed all of the names by which the system is known. Using the Registry protection example from above, the primary server would have "nodeR" and "nodeZ", and the backup server would have "nodeR" and "nodeB". The file `/var/adm/rational/clearcase/rgy/rgy_hosts.conf` also requires changes related to the floating server names. This file points to the Registry host server and will need to be modified on all systems running ClearCase in order to switch the name from the true host name to the floating host name ("nodeR" to "nodeZ" in the above example).

If snapshot views are used, then the storage and database directories must be co-located on the same file system. ClearCase only stores the location of the database files in the Registry and not the View Storage. Co-location solves this problem and ensures that the high availability solution is protecting both directories.

Configuration Steps for LifeKeeper High Availability

This section provides steps which should be followed to configure ClearCase resources within a LifeKeeper cluster in order to ensure the highest levels of availability protection.

Step One: Plan your ClearCase configuration.

This includes the following:

- Determine what ClearCase services to protect, e.g., all services on the server including VOB and/or View storage, Registry, and Release Area, or protect just VOB and/or View storage.
- Determine the permanent and floating server name(s) to be used in the cluster and ensure DNS has been updated to contain all the names and IP addresses.
- If the ClearCase Registry is to be protected, ensure it is located on a LifeKeeper shared resource (shared storage, replicated storage or NAS device).
- Determine the changes required to the Release Area `site.dat` file as well as the changes required to the Registry if adding this kit to an existing ClearCase installation.

Consideration should be given to the type of configuration (Active/Standby vs. Active/Active). It is recommended that if you are protecting the Registry server in an Active/Active configuration that the non-Registry server protects only VOB storage. The reason is that the cleanup which occurs when taking a ClearCase resource out of service will terminate active views on the server, which will occur when the resource protecting the Registry server performs a failover.

Step Two: Set up your ClearCase configuration files.

This includes the *rgy_hosts.conf* file on already installed systems and the *site.dat* file on the Release Host used in installing future systems.

Step Three: Create protected IP addresses under LifeKeeper for the floating server names noted above.

The floating server names must match the host names used in the Registry and when updating the *rgy_hosts.conf* file. Refer to the *LifeKeeper for Linux IP Recovery Kit Administration Guide* for details on setting up IP resources. Once created, test the protected IP addresses by pinging them from a number of clients and servers.

Step Four: Create LifeKeeper NFS resources for each local (non-NAS) storage area used for VOBs and Views as well as the Release Area.

Ensure that the correct IP resource is used with the NFS hierarchy (if the path is */net/nodeZ/export* make sure that IP resource used resolves to *nodeZ*).

Step Five: For existing ClearCase installations, update the Registry with the new host names, global and local paths.

Step Six: Create the ClearCase resource hierarchies in LifeKeeper to protect the Registry, Release Area, and Storage Areas as needed.

Step Seven: Finally, test access to the VOB data while the ClearCase hierarchies are in-service and protected on the primary server.

Summary

Rational ClearCase provides a complete and powerful platform for software development projects on Linux. With full support for 2.4 and 2.6 kernel-based distributions from Red Hat and Novell and available for the complete line of IBM xSeries servers, users can build a development environment which is tailored to their requirements today with the assurance of easy scalability as business needs change in the future. The availability of development resources is critical to productivity and asset protection; SteelEye LifeKeeper provides automated monitoring and recovery of Rational ClearCase and allows the building of multiple configurations that can be used to ensure always-available development resources.

References

IBM Support Policy for ClearCase on Linux

<http://www-1.ibm.com/support/docview.wss?rs=0&uid=swg21159882>

Linux Application Development Environment

<http://www-306.ibm.com/software/rational/linux/>

Supported ClearCase Family Releases

<http://www-1.ibm.com/support/docview.wss?rs=727&uid=swg21136950>

ClearCase Install Guide for Linux

<http://www-1.ibm.com/support/docview.wss?rs=0&uid=swg27005767>

SteelEye LifeKeeper Protection Suite for Rational ClearCase

<http://www.steeleye.com/solutions/>

Trademark Notice

SteelEye Technology, SteelEye and LifeKeeper are registered trademarks of SteelEye Technology, Inc. IBM Rational and ClearCase are registered trademarks of IBM Corporation. Linux is a registered trademark of Linus Torvalds.