A Generic Clustering Framework for the Implementation of High Availability Using Hazelcast

[1]Arijeet Mohapatra, [2]Gaurav Dipte, [3]Naresh Rathore, [4]Vishvendra Rana, [5]Rahul Desai, [6]Rushali Patil

[1,2,3]Information Technology, [4]Computer Engineering, [5]IT Department, [6]Computer Department
Army institute of technology Pune, India
Email: arijeet_firebolt@yahoo.co.in, gauravdipte@gmail.com, nareshsinghrathore7@gmail.com, vsranarocks@gmail.com, rahuldesai@aitpune.edu.in, rushalipatil06@gmail.com

Abstract—This product is a cluster monitoring system developed using Hazelcast. It ensures that the service never goes down in a cluster. This leads to continuous service for the client without any breakdown i.e supporting high availability.

One can easily add more servers in the cluster to meet up the requirements of the enterprise. If at any time master server goes down then no human input is required to make a slave node as the new master. This product will take care of the fact that when a master server goes down then one of the slave servers takes over as the new master and the service is never down.

Hence we achieve High availability of service. It is a back-end product which will work on a cluster of servers which are providing a certain service. Since this product is generic it can work on multiple platforms. The operating systems which it supports are Linux(Centos 6, Ubuntu 13), Windows(7 and 2008 server).

Index Terms—High Availability, Scalability, Hazelcast

I. INTRODUCTION

1.1 Project Description

This project High Availability Clustering Framework using Hazelcast' is for the enterprises and websites which requires maintaining large servers and pool of data. It is high availability and scalable product which can lead to service the client queries at anytime and anywhere. One can easily add more servers in the cluster to meet up the requirements of the enterprise. If at any time master server goes down then no human input is required to make a slave node as the new master. This product will take care of the fact that when a master server goes down then one of the slave servers takes over as the new master and the service is never down. Hence we achieve High availability of service. If we want a computer that will be highly available, we would tradition-ally look for one that has reliable components and incorporates elements of fault-tolerant design. If we're serious about availability, we'll be looking for redundant and hot-swappable power supplies, disk drives, and processor cards as well as error-correcting memory. The problem with these designs is their high cost. This is not a mass-market product. Clusters define a collection of servers that operate as if they were a single machine. The primary purpose of high availability (HA) clusters is to pro-vide uninterrupted access to data, even if a server loses network or storage connectivity, or fails completely, or if the application running on the server fails. HA clusters are mainly used for e-mail and database servers, and for file sharing. In their most basic implementation, HA clusters consist of two server machines (referred to as nodes) that share common storage. Data is saved to this storage, and if one node cannot provide access to it, the other node can take client requests. During normal operation, only one server is processing client requests and has access to the storage; this may vary with different vendors, depending on the implementation of clustering. This product will take care of the fact that when a master server goes down then one of the slave servers takes over as the new master and the service is never down.

1.2 Literature Survey

This chapter reviews and surveys some research papers that are related to the research topic of “HA and HR Web Portal Systems”.

The review starts with “Building Enterprise Portals: Principles to Practice” (Hazra,2002), a paper that discusses how to construct and deploy enterprise Web Portals by using

the component-based approach. In addition, it addresses the challenges in building enterprise Portals from the software engineering point of view. The second research paper is “High

Availability Computer Systems" (Gray & Siewiorek, 1991). It discusses the key concepts and techniques used to build high availability computer systems, and it accesses the future trends of computer systems. The final research paper is "Adding High Availability and Autonomic Behavior to Web Services" (Birman, van Renesse, & Vogels, 2004). This paper addresses the key concepts of Web Services Architecture, outlines the requirements of reliability and vailability, and discusses the model of extending the Web Services Architecture to achieve the goal of "High Availability Web Services".

For designing ideal EP systems, the target systems must offer features to support scalability, extensibility, and continuous availability. From Hazra's (2002) practical experience in integrating most commercially available Portal solutions, the EP should provide the following key features:

Security – protecting and controlling the access of information across the enterprise

Security – protecting and controlling the access of information across the enterprise

- Reliability – ensuring fail-over and crash recovery of mission-critical business processes
- High Availability – supporting 24 X 7 user access and interactions
- Scalability – allowing future expansion of the system to meet the ever-growing business requirements
- Performance – delivering fast response to users' demands
- Search Capability – allowing users to navigate the proper information
- Personalization – presenting the information as desired by specific users
- Customization – meeting the need of different users
- Collaboration – providing the capability to allow users to form user communities that
- share, interact, or communicate specific information

The industry trends for developing EP were also promising during the early 2000s.Hazra (2002) found that there were two major trends of vendors when providing EP solutions;

one is on establishing a comprehensive and integrated approach to deliver EP solutions; and another one is focusing on providing solutions related to one or more specific EP features. Hazra's research also found out that vendors who are specialized in one trend are most likely to collaborate with vendors from the other trend in order to provide a complete set of EP solutions for the intended clients.As outlined by Hazra (2002), the challenges that companies are facing when deploying the EP were:

Formulation of a corporate strategy to integrate multiple business processes or components Development of prioritized business components, processes, and services in a specific period of time Establishment of 'extended enterprise' with qualities or features such as scalability,

Flexibility, continuous availability, and extensibility to meet ever-changing business requirements. The infrastructure of the EP contains five key components, and these components are "content".

1.2.1 Clustering

Clustering is the use of multiple computers, typically PCs or UNIX work-stations, multiple storage devices, and redundant interconnections, to form what appears to users as a single highly available system. Cluster computing can be used for load balancing as well as for high availability. It is used as a relatively low-cost form of parallel processing machine for scientific and other applications that lend themselves to parallel operations. Computer cluster technology puts clusters of systems together to provide better system reliability and performance. Cluster server systems connect a group of servers together in order to jointly provide processing service for the clients in the network.

1.2.2 Hazelcast

Hazelcast is a clustering and highly scalable data distribution platform for Java. Hazelcast helps architects and developers to easily design and develop faster, highly scalable and reliable applications for their businesses. Hazelcast is pure Java. JVMs that are running Hazelcast will dynamically cluster.

1.2.3 In Memory Data grid

An in-memory data grid (IMDG) is a data structure that resides entirely in RAM (random access memory), and is distributed among multiple servers. Recent advances in 64-bit and multi-core systems have made it practical to store terabytes of data completely in RAM, obviating the need for electromechanical mass storage media such as hard disks. It leads to enhanced performance because data can be written to, and read from, memory much faster than is possible with a hard disk. The data grid can be easily scaled, and upgrades can be easily implemented.

1.2.4 High Availability

High availability implies the continues response of user requests. The information must always be given to the user irrespective of down nodes/servers. High-availability clusters (also known as HA clusters or failover clusters) are groups of computers that support server applications that can be reliably

utilized with a minimum of down-time.

II. OVERVIEW

Background Information

This product is a cluster monitoring system developed using Hazelcast. It ensures that the service never goes down in a cluster. This leads to continuous service for the client without any breakdown i.e supporting high availability.

The product will be performing the following operations: -

1. Create Cluster:

Setting up a cluster includes defining the cluster's name and the cluster's multicast address. If multicast is undesirable or unavailable in an environment, then setting up the Well Known Addresses (WKA) feature is required. A cluster name is a user-defined name that uniquely identifies a cluster from other clusters that run on the network. Cluster members must specify the same cluster name to join and cluster.

- Unique cluster name

A unique cluster name is often used with a unique multicast port to create distinct clusters on the same network. A cluster member uses a system generated cluster name if a name is not explicitly specified. Using the system generated name decreases the chance of having overlapping cluster configurations on the network.

- Set of identifiers

A set of identifiers are used to give a cluster member an identity within the cluster. The identity information is used to differentiate cluster members and conveys the members' role within the cluster. Some identifiers are also used by the cluster service when performing cluster tasks. Lastly, the identity information is valuable when displaying management information and facilitates interpreting log entries.

- Cluster members

Cluster members use multicast communication to discover other cluster members and when a message must be communicated to multiple members of the cluster. The cluster protocol makes very judicious use of multicast and avoids things such as multicast storms. By default, data is only transmitted over multicast if it is intended for more than 25% of the cluster members. A multicast address (IP address and port) can be specified for a cluster member. Cluster members must use the same multicast address and port to join and cluster.

- Distinct clusters

Distinct clusters on the same network must use different multicast addresses. Using the default multicast address and port (and the system generated cluster name) increases the chance of having overlapping cluster configurations on the network. This can lead to cluster members accidentally joining an unexpected cluster. Always use a unique port value to create a distinct cluster.

2. Monitor Cluster

Cluster Performance Monitoring**:**

Monitoring coordination performance at all levels is necessary to ensure that clusters are efficient and effective coordination mechanisms, fulfilling the core cluster functions, meeting the needs of constituent members, and supporting delivery to affected members.

- Identify the level of performance of the Health Cluster's core functions
- Identify actions for improvement or requests for support
- Monitor progress over time
- Polling :

Polling is also known as Heartbeat of a cluster. Through this we monitor that our service is running and master server is responding to clients. If in any case polling is stop then the entire services will also stop. So we keep on monitoring the process.

- Assigning new IP address:

Whenever the master server changed and one of the slaves take over and become master then we have to assign the new cluster IP to that particular slave.

- Cluster Member Status:

  Following attributes can be monitored**:**

- Cluster**:**
- Configuration
- Cluster name
- Count of members and their address (host: port)
- Operations: RESTART, STOP, START, REMOVE
- Member**:**
- Inet address
- Port, Member Status

3. Monitor Service:

In monitoring of services we test our product on two use cases which are :

- Mysql
- Nagios

4. Manage Cluster

Managing a cluster is a very important function because all the functions like working failover or upgrading a server etc. are done in this. Adding or Removing the node for providing the high availability or because the down of any server. So this is the very important function after the setting up a cluster for running of that cluster very smoothly.

There are mainly two types of failover configuration in clustering as:- active/active and active/passive clustering. In active/active mode, when fail over occurs, the secondary companion takes over the devices, client connections and so on from the primary companion. The secondary companion services the failed-over clients as well as any new clients, until the primary companion fails back and resumes its activities. But in active/passive mode, when a system fails over, the adaptive server and its associated resources are relocated to and restarted on the secondary node.

- Adding a node:-

When the node is added in the existing cluster, it provides the high availability of data using the Replication Mechanism. A cluster name is user defined name that uniquely identifies the cluster from the other clusters that run on the same network. The cluster member must specify the same cluster name to join and cluster. A cluster member does not start if the wrong name is specified when attempting to join an existing cluster.

- Removing a node:-

If any server fail or shutdown then that server is to be removed from the cluster and data that was present at that server is to be assigned to the other servers using replication mechanism. For removing of that server we use the IP address of that server.

- Change Master:-

If the master server goes down in that case any other server should be the master so we regularly check the master server that is it down or not? If it is down then other server is made master and the IP of the master server is assigned to that server and all other servers are informed that master server is changed.

- Assigning the cluster IP:-

The cluster IP address is the virtual IP address that is assigned to the cluster. Client requests are sent to the cluster IP address. The cluster IP address has a corresponding subnet mask that is part of the cluster IP address specifications. The criteria for specifying the cluster IP address and subnet mask are the same as for all other IP addresses. When we configure the cluster manually, we must configure the cluster IP address and subnet mask identically on all cluster hosts within the cluster. We can specify it after the installation under cluster control by first configuring network adapters and then by going there and changing the configuration accordingly.

- Change interface:-

To change the network interface, we have to edit the operational override file and an element that specifies the IP address to which the socket is bind.

5. Replication

Maintaining redundant and updated copy of same data at different server for optimized performance of cluster is done by replication. The Cluster Replicator looks in the Cluster Database Directory to determine which databases have replicas on other cluster members. The Cluster Replicator stores this information in memory and uses it to replicate changes to other servers.

Maintaining redundant and updated copy of same data at different server for optimized performance of cluster is done by replication.

- Need**:**

In a cluster data is not primary to only one server. The data is distributed to other servers. If one server is busy processing more requests or is down, the clients request is sent to another server which maintains the copy of the same data and provides the required service.

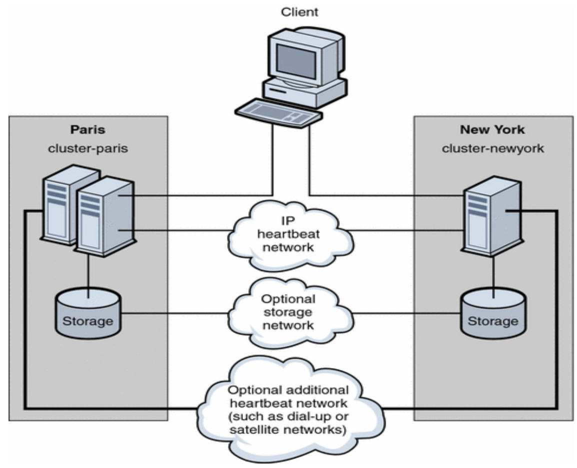
- How it Works:

The Cluster Replicator looks in the Cluster Database Directory to determine which databases have replicas on other cluster members. The Cluster Replicator stores this information in memory and uses it to replicate changes to other servers.

- Active/Passive Clusters:

The clusters can be active clusters (which reflects the changes in them instantaneously in other replicas) or passive clusters, updating of data over different server replicas are done by observing its type i.e. active/active or active/passive cluster.

Replicator do not maintain replica of every database on every server. The number of replicas you create for a database depends on how busy the database is and how important it is for users to have constant access to that database**.**



III. PRODUCT FEATURES

The high availability clustering framework has the following main features :-

3.1 High availability and Scalability

3.1.1 Description and Priority

This feature is of the highest priority. The High availability feature is provided with the help of replication mechanism. Even if a server goes down the service would never stop. If our servers are facing too much traffic and are having difficulties in handling the traffic then new servers can be added very easily. Hence high availability is achieved.

3.1.2 Stimulus/Response Sequences

When a client sends a request to the cluster and the master server is either down or facing too much traffic then the server transfers the request to another server which has a replica of the required data. This way high availability is achieved.

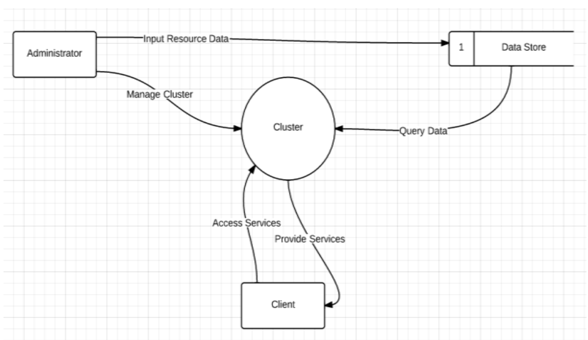
3.2Generic
3.2.1 Description and Priority

This feature is of medium priority. This product is compatible with different operating systems like Centos, Ubuntu, Windows 7, Windows 2008 server.

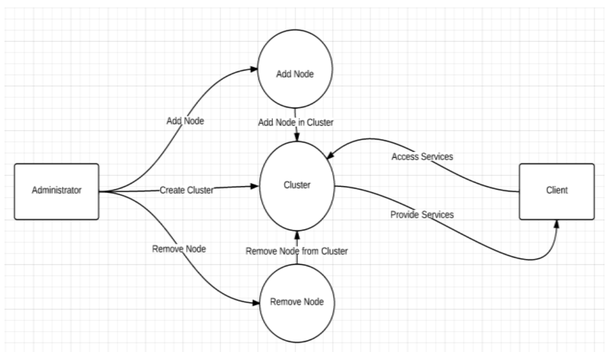
IV. DESIGN

DFD's(Up to Level 2)

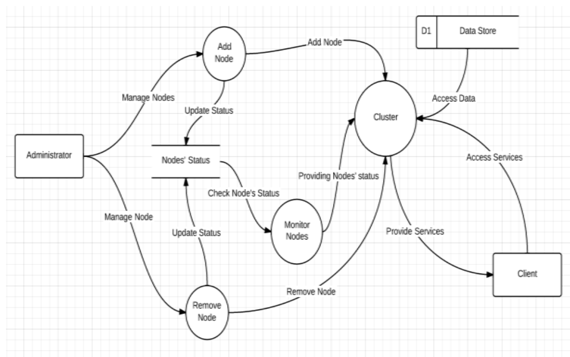
1. Context Level



2. Level 0



3. Level 1



V. SIMLATION AND RESULT

This chapter discusses the research results on HA and HR Web Portal Systems. We start with the discussion of the research findings on HA and HR fundamentals, and then

Provide a detailed description on hardware selection for building HA and HR Web Portal Systems. We also propose the critical specification for server software components when constructing HA and HR Web Portal Systems. Finally, we offer a high-level design for Web Services and the target HA and HR Web Portal System.

For constructing HA and HR Web Portal Systems, we propose the cluster server based architecture for the hardware backbone. The intention for choosing the cluster server architecture is to apply the concepts of designing a fault-tolerant system:

1. Modularity – each server in the cluster is a module, and all components and parts of the server are modules as well. We can take any module offline for maintenance or repair purposes. Most of the server hardware vendors do provide modularity within the design of their server hardware

2. Fail-fast – All servers (nodes) within the cluster either operate correctly or stop working. When a server fails to deliver the expected result, this failure server will be brought offline immediately.

3. Independent failure modes – All nodes in the same cluster are independent. Any given node within the cluster does not depend on any other nodes. If a node fails for any reason, the failure will just affect the current node without affecting other nodes in the cluster.

4. Redundancy and repair – All nodes within the cluster are redundant, and we can take down any single node from the cluster to perform the maintenance and repair task without affecting the normal operation of the entire cluster.

There are features and capabilities offered by the RDBMS that can help us to achieve the goal of building HA and HR Web Portal Systems. There are some mandatory requirements that RDBMS has to provide in order to achieve the target of being an HA and HR database server:

1. Data Failure Protection – The RDBMS should provide the mechanism to protect the data being stored in the repository from failure. The RDBMS should be able to detect failures and correct them accordingly in order to maintain the integrity of the database. Data replication, data mirroring, or data backup and restore are typical mechanisms for protecting data.
2. Database Server Crash Recovery – the Database Server Crash Recovery mechanism is important to the database server's reliability. In order to increase the database server's availability, the downtime of the server has to be minimized, and how it recovers from a database server crash is a critical factor to select an HA DB server.

3. Database Clustering – the concept of database clustering is similar to the one discussed in the computer server clustering. We organize database clustering in a way that there are different instances of database servers running on different computer systems, and we chain all of the database servers into a cluster; a database cluster.

4. Database Reorganization – the database reorganization capability provides the system the flexibility to modify the existing data structure of the database without shutting down the database. This feature can contribute to a high degree of database availability when a database alteration is required to meet the business objective, while the database is under the process of data modification and still providing online services to the intended users.

VI. CONCLUSION

To develop communication simulator over wired network by modifying UDP protocol. The scope of work is to improvise the reliability, latency of system during data (like text, audio, video) transfer to make it work for real time applications.

VII. ACKNOWLEDGEMENT

We take this opportunity to thank our project guide Prof. Rahul Desai and Prof.Rushali Patil and Head of the Department Dr. Mrs. Sangeeta Jadhav and S. Dhore for their valuable guidance and for providing all the necessary facilities, which were indispensable in the completion of this project report. We are also thankful to all the staff members of the Department of Information Technology and Computer Engineering of Army Institute of Technology, Pune for their valuable time, support, comments, suggestions and persuasion. We would also like to thank the institute for providing the required facilities, Internet access and important books. I also thank to the Dr. V.P. Gosavi, Principal of Army Institute of Technology for providing me all the necessary facilities to carry out the project work.

Lastly we would also like to thank G.S. Lab, Pune for sponsoring our project and giving much required motivation.

VIII. REFERENCES

[1] M. Chandy and L. Lamport. Distributed snapshots: Determining global states of distributed systems. ACM Transactions on Computing Systems, 3(1):63–75, 1985.

[2] J. Duell, P. Hargrove, and E. Roman. The design and implementation of berkeley lab's linux checkpoint/restart. Technical report, Berkeley Lab, 2003.

[3] M. Elnozahy, L. Alvisi, Y-M. Wang, and D.B. Johnson. A survey of rollback-recovery protocols in message-passing systems. Technical Report CMU-CS-99-148, Carnegie Mellon University, June 1999.

[4] Ibrahim Haddad, Chokchai Leangsuksun, and Stephen L. Scott. Ha-oscar: the birth of highly available oscar. Linux J., 2003(115):1, 2003.

[5] Richard Koo and Sam Toueg. Checkpointing and rollback-recovery for distributed systems. IEEE Trans. Softw. Eng., 13(1):23–31, 1987.

[6] R. Lottiaux and C. Morin. Containers : A sound basis for a true single system image. In IEEE International Symposium on Cluster Computing and the Grid, May 2001., 2001.

[7] Renaud Lottiaux, Benoit Boissinot, Pascal Gallard, Geoffroy Vallée, and Christine Morin. Openmosix, openssi and kerrighed: A comparative study. In Cluster Computing and Grid 2005 (CCGRID 2005), Cardiff, England, May 2005.

[8] C. Morin and I. Puaut. A survey of recoverable distributed shared memory systems. IEEE Trans. on Parallel and Distributed Systems, 8(9):959–969, 1997.

[9] Christine Morin, Pascal Gallard, Renaud Lottiaux, and Geoffroy Vallée. Towards an efficient Single System Image cluster operating system. Future Generation Computer Systems, 20(2), January 2004.

[10] Christine Morin, Renaud Lottiaux, Geoffroy Vallée, Pascal Gallard, Gaël Utard, Ramamurthy Badrinath, and Louis Rilling. Kerrighed: a single system image cluster operating system for high performance computing. In Proc. of Europar 2003: Parallel Processing, volume 2790 of Lect. Notes in Comp. Science, pages 1291–1294. Springer Verlag, August 2003.

[11] openSSI.org. Introduction to the SSI Cluster. http://www.openSSI.org/. [12] Eduardo Pinheiro. Truly-transparent checkpointing of parallel applications.

[13] Brian Randell. System structure for software fault tolerance. Software Engineering, 1(2):221– 232, 1975.

[14] D. L. Russell. State restoration in systems of communicating processes. IEEE Trans. Software Eng., SE-6(2):183–194, March 1980.

[15] Esposito Mastroserio Tortone. Openmosix approach to build scalable hpc farms with an easy management infrastructure.

[16] Geoffroy Vallée, Renaud Lottiaux, David Margery, Christine Morin, and Jean-Yves Berthou. Ghost process: a sound basis to implement process duplication, migration and checkpoint/ restart in linux clusters. In The 4th International Symposium on Parallel and Distributed Computing, Lille, France, July 2005.

[17] Geoffroy Vallée, Renaud Lottiaux, David Margery, Christine Morin, and Jean-Yves Berthou. Ghost process: a sound basis to implement process duplication, migration and checkpoint/ restart in linux clusters. In The 4th International Symposium on Parallel and Distributed Computing, Lille, France, July 2005.

[18] Geoffroy Vallée, Renaud Lottiaux, Louis Rilling, Jean-Yves Berthou, Ivan Dutka-Malhen, and Christine Morin. A case for single system image cluster operating systems: Kerrighed approach. Parallel Processing Letters, 13(2), June 2003.

[19] www.hazelcast.com/org.

[20] R. Nelson, "Exploring High Availability Issues with BEA Tuxedo and Third Party High Availability Software", Aurora information Systems, January 1998.

◈◈◈