Bachelor's Thesis (UAS)

Degree Program: Information Technology

Specialization: Data Communication

2010

Newton Kamori

CLUSTERING TECHNIQUES

A Solution For E-business

TURUN AMMATTIKORKEAKOULU
TURKU UNIVERSITY OF APPLIED SCIENCES

BACHELOR’S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Degree Programme: Information Technology

Specialization: Data communication

Date: June 2010

Number of pages: 38

Instructor: Wikström Yngvar

Author: Kamori Newton

CLUSTERING TECHNIQUES

A Solution For E-business

The purpose of this thesis was to provide the best clustering solution for the Archipelago web site project which would have been part of the Central Baltic Intereg IV programme 2007-2013. The entire program is a merger between the central Baltic regions of Finland, including the Åland Islands, Sweden and Estonia.

A literature review of articles and research on various clustering techniques for the different sections of the project led to the findings of this document. Clustering was needed for web servers and the underlying database implementation. Additionally, the operating system used for all servers in both sections was required to present the best clustering solution.

Implementing OSI layer 7 clustering for the web server cluster, MySQL database clustering and using Linux operating system would have provided the best solution for the Archipelago website.

This implementation would have provided unlimited scalability, availability and high performance for the web site. Also, it is the most cost effective solution because it would utilize the commodity hardware.

FOREWORD

The Central Baltic IV programme 2007-2013 is a cross-border co-operation program under the European Territorial Co-operation Objective covering regions from Estonia, Finland including the Åland Islands, Latvia and Sweden and with an allocation from the European Regional Development Fund of more than 100 million Euros. The programme is divided into three main parts; Central Baltic Programme, Southern Finland–Estonia Sub-programme, and the Archipelago and Islands Sub-programme.

The main focus of the entire programme is to provide a safe and healthy environment, an economically competitive and innovative region and attractive and dynamic societies. More information about this program can be found at http://www.centralbaltic.eu/ .

I would like to thank God for giving me the strength I needed to finnish this thesis. My sincere appreciation to my supervisor, Wikström Yngvar, for his guidance and advice. Lastly, my heartfelt appreciation to my loving parents and friends for their overwhelming support through out my academic program.

Author’s name: Kamori Newton

TABLE OF CONTENT

1. Introduction

Most modern systems demand very high computing power which has been a major issue for the past few years. This is because the traditional design of computer systems where one powerful server was used had limited scalability with the growing information systems. Additionally, the server experienced a bottleneck problem because they could not respond as fast as expected hence leading to delays and poor performance.

Therefore, this design would certainly not scale well for the Archipelago website which was a vital part of the Archipelago and Islands Sub-program of the Central Baltic programme. As a result, a better solution was needed to ensure that the Archipelago web solution would provide high availability, reliability, unlimited scalability, and high performance. Most modern systems implement supercomputers or a clustering solution.

A supercomputer is a high performance computer that can be used as a server to increase the processing power. They are the fastest computers in the world and may be suitable for solving most of our challenges in the information systems. However, they are very expensive and most companies may not even afford one. Also, the design provides a single point of failure. This is because a failure of the server would result in the whole system incurring some downtime, unless the back up server is another supercomputer.

Clustering was developed as a cheaper alternative to supercomputers. In clustering, ordinary computers are combined together to form a single unit called a cluster. The processing work is shared between members of the cluster hence providing higher processing power. More computers can be added to a cluster when more processing power is needed. Therefore, clustering is the most scalable solution and has been employed in most modern information systems.

The goal of this thesis was to search for the best combination of clustering techniques that could be used for the project. In this project clustering would have been needed for the web servers and database parts of the web solution. In addition the Operating system used needed to provide the best clustering support and options.

This document is divided into the following sections; project analysis, clustering basics, web server clustering, database clustering, different implementations, project recommendation and the conclusion.

In the first section, a brief description of the Archipelago Web site project is outlined. The second section of clustering basics evaluates the main components of a cluster and the requirements. In the third and fourth sections of web server clustering and database clustering respectively, different clustering techniques being used were assessed. In addition, the database section also reviewed how different techniques are used by different vendors. The other section provided a review of how the techniques above could be implemented under different platforms. In this thesis, Windows and Linux were considered. Finally, a review of the results of this thesis provided a recommendation or what might have been a good combination of techniques to use for the project.

2.Project Analysis

The Archipelago site was meant to connect the business resources to their demands in the effort of improving the economy. The site would also make work easier for all the residents, travelers and businesses especially small scale. However, it was important to note that there are some existing web sites for small businesses or companies. The plan was to embed these small sites into the Archipelago site (appendix 1) so that customers and residents could easily locate what they need as well as enabling small companies to easily benefit from each other. A good example is tourism; consider a case where travel agencies could do business with hotels. Take for instance when a customer does a travelling booking, accommodation is also required. In this case, a customer would easily make a hotel booking as well if the option is presented on the same site. In addition payments would also be made at the same time or later via the same site.

The website would also have some graphic tools that are could be updated automatically to help travelers plan their journey. It would provide the time tables for the ship and ferries, destinations, prices and information of how far (in Km) are specific destination.

Other beneficiaries of the website would be the small scale businessman who would be expected to advertise there product for example, those who rent summer cottages, fishing equipments, maybe small boats and extra.

Also, including some pictures of different beautiful sceneries of the region in the web site could help promote tourism, which would in return help improve the economy of the region. Consequently, advertisements of major events could be posted in the site mainly in the news and events section.

Clustering techniques would have been the best option because they are the most cost effective solutions providing unlimited scalability, high availability and high performance. In this project clustering was required for the database platform and the web servers. Also, the underlying operating system used was required to present the best clustering for both the database and the web server.

The following parts will describe various clustering solutions offered by specific products which were assessed as an effort to search for the best solution that might have been used in the Archipelago website project.

3. Clustering Basics

A cluster is group of independent servers that constitute a single system. Each server within the cluster is referred to as a node. Clusters were developed with the aim of providing a cost effective solution for applications that require high performance computing than a single server can provide. Supercomputers provide an alternative but they are very expensive and hence would not have suited this project.

Clustering was initially developed under the project Beowulf at the NASA Goddard Space Flight center by Thomas Sterling, Donald Becker [1, Chapter 1]. This project aimed at using parallel computing where, applications run across a number of nodes simultaneously [1, chapter 2].

Since then, clustering has been implemented in most organizations and companies with the aims of improving performance, for high availability, scalability and reliability. For example, Google uses over 15,000 computers with fault-tolerant software to provide a high-performance web search service [2].

To set up a cluster, there are various requirements that need to be met. In this section, some of the basic requirements for setting up a cluster were discussed.

3.1. Cluster Nodes

A node is an ordinary server that has been used in a cluster. There are no specifications of how much processing power a node should have but the choice of hardware and system area network affects the general performance of a cluster. Therefore, the choice of hardware to be used by the node should be taken seriously [1, chapter 2]. Consequently, the node must have a NIC (Network Interface Card) installed for connecting the node to the system area network. The network is used for internodes communication [1, Chapter 4].

3.2. Network Requirements

3.2.1. Hardware

Every node should have a NIC (Network Interface Card) connected to the I/O (Input/Output) bus or integrated on the motherboard. The NIC enables devices to send and receive messages across the network. It contains a network link, which mostly depends on the kind of topology to be used. Ethernet networks are the most common due to their low cost of installation, and hence, most nodes might have an Ethernet NIC. However, there are different link speeds, for example, 10Mb/s, 100Mb/s, 1Gb/s, 10Gb/s and many more [1, chapter 4].

In addition, different clustering technique implementations have different requirements. For example, Oracle RAC (Real Application Cluster) recommends that, a node should have at least two NICs; one for the public network and the other for the private network cluster traffic communication [3, Part 4].On the other hand, MySQL AB recommends 100Mb/s as the minimum requirements for a all NICs used in a cluster [4, 9.3.1].

Other hardware devices needed for a cluster network includes hubs, switches, routers and firewalls. A Hub or a switch operates as the central device for interconnecting the cluster nodes. Routers are required when data needs to be conveyed to a different network or subnet [5], while a firewall is mainly implemented for security purposes. It may be placed at the edge of the network where it filters all the data from the internet before it gets to the cluster.

3.2.2. Topology

Most cluster topologies are usually simple comprising of a single switch connecting all the devices in the cluster. However, switches can be added as the cluster increases in size because the ports from a single switch will eventually get used up [1, chapter 4].

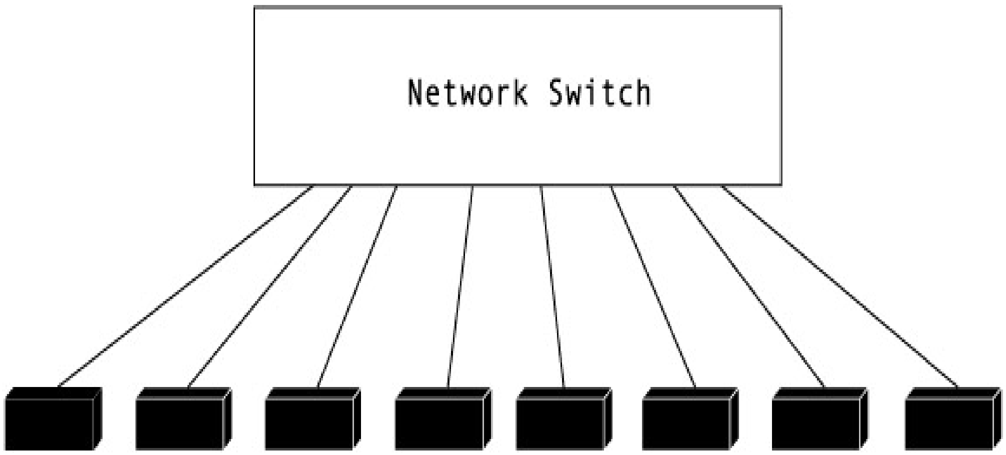


Fig 1: A simple cluster topology [1, chapter 4.2]

From the fig 1 above, all nodes are connected to a single switch; therefore the set up for this topology is very simple.

3.2.3. Software

For the network to operate, some rules needs to be employed. These set of rules are referred to as protocols. The protocols used are responsible for various processes in the network. First, they describe the structure of the message to be transmitted over the network. Secondly, they describe how devices in the network share information about pathways in the network. This helps network devices know how they can send messages or data to specific destinations. Consequently, protocols define how and when error and system messages can be transmitted across the network. Finally, they set up and terminate the communication pathway when needed [5, 2.3.2].

Most networks use a combination of protocols for a successful transmission of data. The TCP/IP (Transmission Control Protocol / Internet Protocol) is the most commonly used especially in the database systems since reliability is important. The TCP sets up the communication path and then monitors it to ensure that data is sent and received hence providing reliability. The IP handles the functions of routing packets of data to their destination. Routing is done by using the IP addresses assigned to the network devices. Large pieces of data are usually fragmented into smaller packets that are then easily transmitted across the network [1, 4.3.1].

3.2.4. Network Security

To avoid latency, data and messages sent within the cluster are not encrypted. Therefore, security is enforced using other means for example, through a hardware firewall at the edge of the network or software firewalls enforced within the network devices. However, the best practice is assigning the cluster a unique network address or subnet address from the rest of the network [4, 9.3.1]. Alternatively, a private address can be implemented for the internodes communication while NAT (Network Address Translation) is used for the traffic leaving the cluster.

4. Web server clustering

The internet has grown at an exponential rate over the past several years. This has led to the development of web clustering techniques which are now implemented in most web servers today. A web cluster is made up of a front-end-node referred to as a web-switch (or the dispatcher). A web switch is responsible for dispatching request to several back-end servers. The servers then process the requests and respond to their clients [6] as shown in the Fig. 2 below.

The web-switch is the key component to this design. It acts as a proxy for the web clients, receiving all the requests and dispatching them to the suitable back-end server. The web-switch is usually configured with the cluster IP address and hence all clients direct all the requests here without knowing of the existence of the cluster.

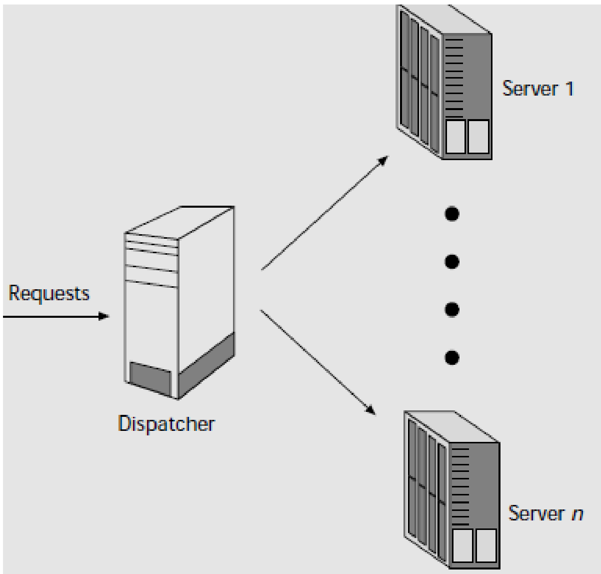


Fig. 2: A high-level view of a basic server cluster showing the Dispatcher and n servers in the server pool [7, 2]

From the Fig.2 above, the dispatcher (web-switch) may use various dispatching policies to forward the request to the back-end severs. The type of policy used depends on the architecture and the web-switch. Web-switches are classified according to the OSI (Open System Interconnection) layer they operate in, for example, there are Layer 4 and Layer 7 web switches.

Layer 4 web-switches operate at the TCP/IP level. The web switch maintains tables for every TCP session. It then examines every incoming request by checking the header flag field to see if it belongs to an existing session or a new session. Packets that belong to the same session are forwarded to the same server.

Layer 7 switch operates at the application layer of the OSI layer model. Before dispatching the request, a thorough content evaluation of the packet is done to determine the most suitable back-end server. Unlike the Layer 4 switch, this switch deploys content aware information distribution [6, 3].

4.1. Web Cluster Architectures

As mentioned above, different architectures will require varying dispatching mechanisms that are enforced by the web switch. Some of the most common clustering techniques used for the web involve; L4/2 (OSI Layer 4/2) clustering, L4/3 (OSI Layer 4/3) clustering and L7 (OSI Layer 7) clustering. The former two methods use static policies to dispatch requests to the back-end server. Static policy only alters the packets header and then forwards the client's request to the selected back-end server through simple algorithms like RR (Round Robin) and RAN (Random). RR distributes the load in a circular manner with the help of a pointer to help know the last server that was used, while RAN distributes the requests uniformly to all nodes. As a result, static policies provide a faster response time, but they might lead to a very poor performance.

However, there are some Layer 4 web switches that use WRR (Weighted Round Robin), a dynamic algorithm. It is an advanced RR algorithm that uses some weights to select the appropriate server. The weights are directly proportional to the server state and are periodically calculated by the web switch. They vary depending on the load on the CPU (Central Processing Unit), disk and network resources at each server.

On the other hand, the L7 clustering uses dynamic policies like LARD (Locality Aware Request Distribution) and MC-RR (Multi-Class Round Robin). Dynamic policy inspects the URL (Universal Request Locator) request to consider what kinds of services are required. A suitable back-end server is then selected and the packets are forwarded.

4.1.1. L4/2 Clustering

As the name suggests, this approach operates at layer 4 (transport layer) and layer 2 (data link layer) of the OSI layer model. The IP address of the cluster is configured as the primary address of the dispatcher and as an alias to all the back-end servers. As a result, all servers in the cluster share the same IP address as shown in the fig 3 below.

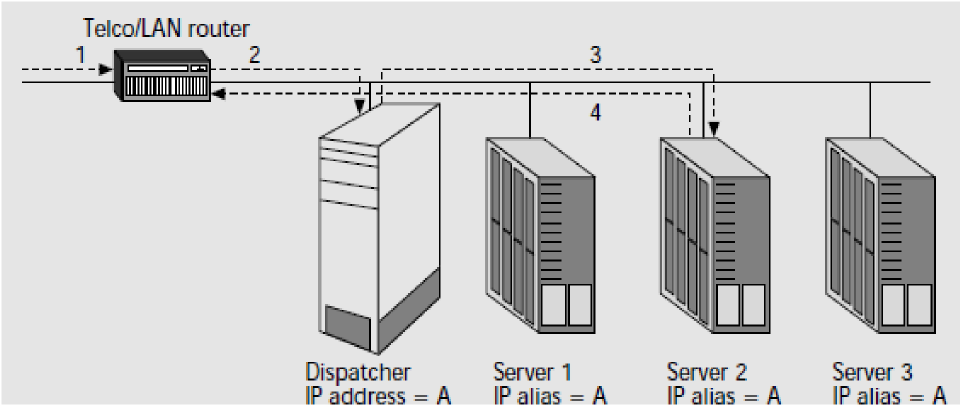


Fig 3: L4/2 clustering [7, 4]

From the Fig 3 above, the router send all requests to the dispatcher with address A. The dispatcher then, modifies the packet by changing the MAC (Media Access Control) address in the received packet to the MAC address of the server it has selected, for example in this case it can be either server 1, server 2 or server 3.The packet is then forwarded to the back-end server. Any other packets received by the dispatcher belonging to the current session are forwarded to the same server in a similar fashion.

After the request is selected the selected server can communicate directly with the client resulting to a better response time than L4/3. However, the only restriction of L4/2 clustering is that all the servers must be directly connected to the dispatcher. Some of the commercial products used to implement this clustering are, ONE-IP, IBM's eNetwork Dispatcher, LSMAC, and ACEdirector.

ONE-IP was developed by Bell Laboratories and was the initial implementation of L4/2 clustering. All traffic is received by the dispatcher, where the client's address is hashed to indicate the back-end server (or node) that will process the request. This approach also provides fault tolerance for the dispatcher and the servers through a watchdog deamon.

The eNetwork dispatcher, developed by IBM uses weighted round-robin algorithm to distribute client's requests to the cluster nodes (servers). It is capable of speeds of up to 2200 requests/s. It also provides fault tolerance for both the dispatcher and the nodes.

LSMAC, developed by the University of Nebraska-Lincoln is a portable user-space application that runs on the commodity system. It is capable of delivering a speed of 1800 connections/s and also provides fault tolerance to the servers and the dispatcher.

Finally, the ACEdirector, develop by Alteon, is implemented as an Ethernet switch. It uses round-robin and least connections algorithm to dispatch client's requests. It can offer a speed of up to 25000 connections/s. Also, it offers fault tolerance for other servers in the cluster and uses another switch as a standby [7, 2-4].

4.1.2. L4/3 Clustering

In this method, all servers have unique IP address. Clients see the dispatcher as a single host but to the back-end servers it appears as a gateway. Traffic sent from clients is received by the dispatcher as shown in the fig 4 below. At the dispatcher, the packet is modified by changing the source IP address to the IP address of the back-end server selected. In addition, all integrity codes affected like, the CRC (Cyclic Redundancy Check), and ECCs (Error Control Check), are recalculated and appended to the packets of data.

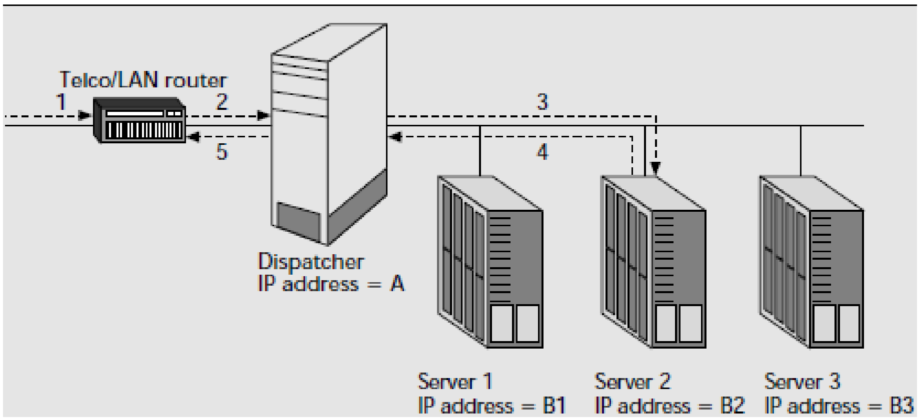


Fig 4: L4/3 clustering [7, 5]

The packets are then forwarded to the selected server as shown in the fig 4 above. Consequently, the dispatcher records the entry of all the TCP sessions initiated and the selected server in a table. This enable the dispatcher know how to forward packet belonging to the same session. There are several different approaches to L4/3 clustering for example LSNAT (Load Sharing Using Network Address Translation), Magicrouter, and Cisco’s LocalDirector.

LSNAT runs in standard hardware under Linux or modern UNIX systems. Its performance is rather slower than other approaches like using the Cisco’s LocalDirector. In this implementation, if the dispatcher (LSNAT) fails, one of the servers in the pool detects the failure it reconfigures itself to take the role as the dispatcher.

Also, magicrouter is a software based approach that dispatches traffic using the round robin, random and incremental load algorithms. It also provides fault tolerance by employing the primary/backup model.

Consequently, Cisco’s LocalDirector can provide support for up to one million simultaneous connections. It implements round robin, weighted percentage, fast response, and least connections algorithms to dispatch traffic. The localdirector offers the best response time of the three approaches [7, 5-6].

Unlike in L4/2 clustering all replies to the client must be channeled through the dispatcher. As a result, L4/2 operates faster than L4/3. Also, L4/3 need to recompute some codes making it even slower. However, in L4/3 design, there is no need for physical direct connection to the dispatcher since all IP addresses are unique.

4.1.3. L7 Clustering

L7 clustering operates at the application layer of the OSI layer model. The dispatch criterion is based on the content of the client's request. Therefore, this method promises to provide better performance and results because it can make good judgment of the suitable back-end server. L7 uses different clustering algorithms like, LARD and MC-RR, to forward client's requests.

4.1.3.1. LARD

In this approach, several different subtypes of a request are defined and each category assigned to a specific server. When a clients request is received at the dispatcher, the requests is examined and categorized accordingly, then forwarded to the appropriate servers of that category as shown in the fig 5 below.

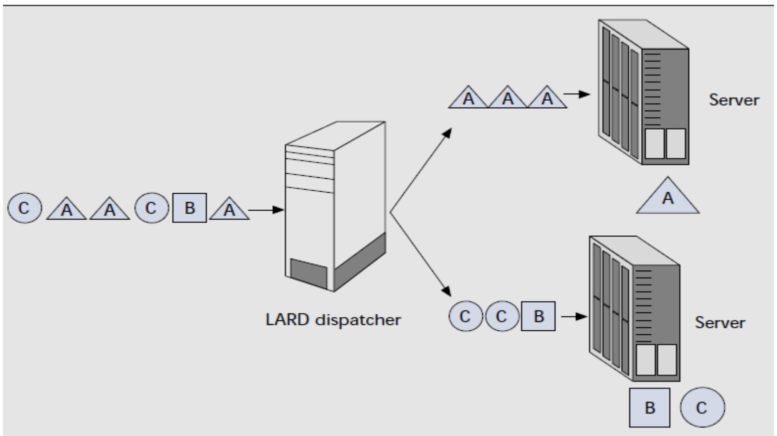


Fig 5: An overview of LARD [7, 6].

Each back-end server serves requests of a specific category as seen above. The dispatcher then informs the selected back-end server about the status of the connection enabling it to communicate directly with the client [7, 6].

In operation, LARD assigns all requests for a target file to the same node until a certain utilization threshold point is attained. At that point, the least loaded node is then selected.

4.1.3.2. MC-RR

MC-RR was developed as a combination of WRR and LARD. It divides clients request into several classes based on the resource requirements for example, transactions requests like html forms, data retrieval requests, multimedia requests among others. The web switch then forwards the different classes to the back-end servers, ensuring that the weight is evenly distributed. The fig 6 below shows a representation of this algorithm.

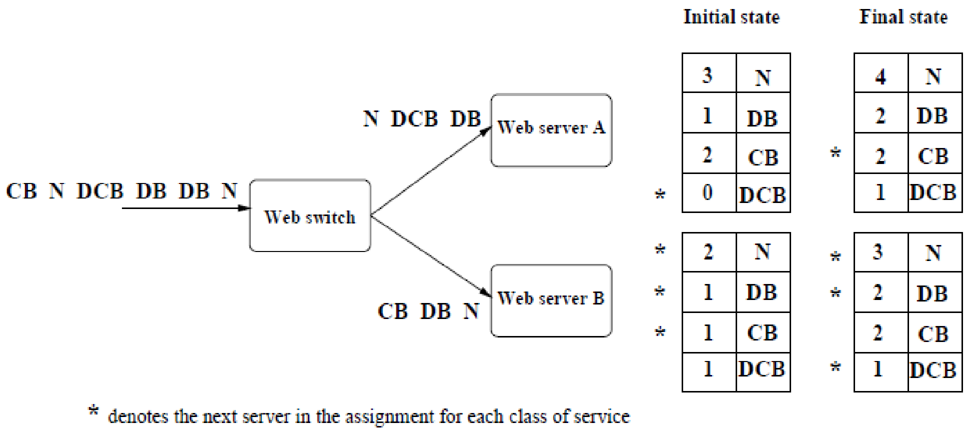


Fig 6: MC-RR algorithm policy example [8, 4]

The policy defined in the example defines four classes of web traffic; CPU bound (CB), web publishing services (N), disk and CPU bound (DCB), and disk bound (DB) services [8]. The final state of the cluster shows that the load is evenly balanced on both servers with each server having the same number of requests of different classes. As a result, MC-RR is considered to provide the best scalability of all the other algorithms. In addition, it guarantees robust results for different classes of web service because it does not need any hard tuning of system parameters [8, 4].

5. Database Clustering

Databases are the backbone of the information systems in the modern world. Over the last decade, the sizes of the data warehouses have increased tremendously. This has resulted in a decline in database performance hence the need to develop systems that can accommodate the growth and the challenges of the traditional database designs. Modern systems must be scalable, highly available, maintain a high performance and they must be reliable.

Clustering provides the best solution for databases and has now gained interest among vendors; hence, most DBMS (Database Management System) software supports it across many platforms. There are three different clustering architectures; Shared-memory, Shared-disk (SD) and the Shared-nothing (SN). In the shared-memory design, all of the CPUs in the cluster share a single memory and a single collection of disks. In the shared-disk design all hosts in the cluster have there own CPU and memory but they share storage. Finally, in the shared-nothing architecture, every host is independent and does not share any resources with other hosts in the cluster [9].

This section examines the three clustering architectures as well as providing examples of how they have been implemented in some DBMS programs like MySQL, Oracles, and SQL-Server.

5.1. Shared-memory

All nodes in the cluster share the same memory and disk. Data can therefore be easily accessed by the nodes hence no need for complex locking mechanism. This eliminates the need for the distributed locking and commits protocols hence making it easy to implement [9, 1].

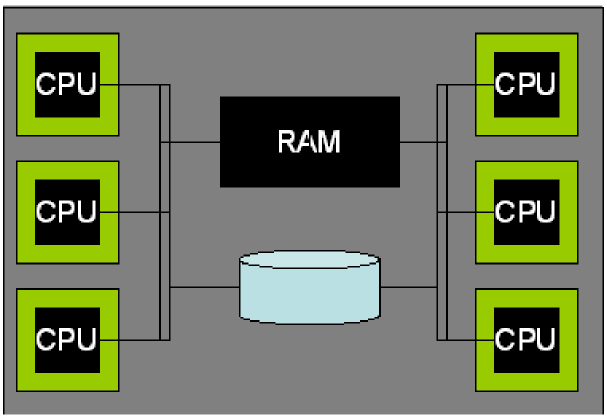


Fig 7: Shared-memory architecture [10, page 166]

Conversely, shared-memory provides limited scalability, since all CPUs compete for the same bus resulting to the bus being a bottleneck.

Most modern systems implement nodes with more than one processor installed. As a result, shared-memory is a standard implementation since all CPUs in one node will share the same disk and memory.

5.2. Shared-disk

Shared-disk was introduced in the early 1980s by Digital Equipment Corporation for use in the VAXcluster [9]. Every node is independent of its RAM (Random Access Memory) but they all share the same storage. Multiple disks are usually connected to form a SAN (Storage Area Network) or NAS (Network Area Storage) system. The fig 8 below show an example shared-disk environment that comprises of six nodes [10].

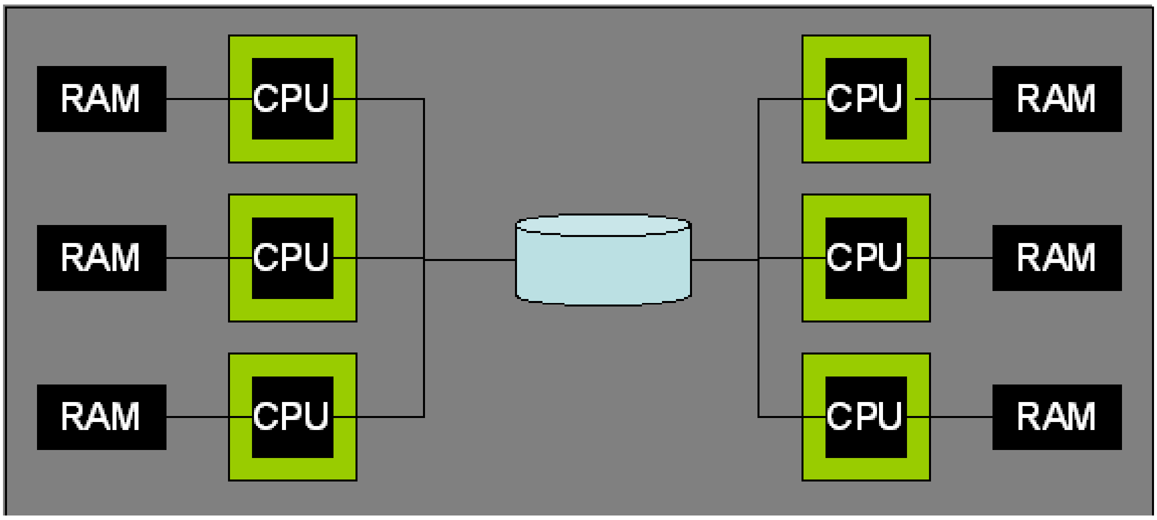


Fig 8: Shared-disk architecture [10, page 170]

From the fig 2 above, all nodes access the same storage which means that a failure of any node does not result to any loss of data since the nodes are not responsible for storage as it is the case in SN. In addition, this design has proven to offer better load balancing than the other architectures. Mike Hogan, CEO of ScaleDB states that "shared-disk provides inherent and dynamic load-balancing across the entire cluster" [11].

However, shared-disk poses some drawbacks over other designs. First, all nodes need to access the same storage which may at times result to an I/O bottleneck. Additionally, centralized storage poses a threat of a single point of failure for example if the storage area fails. Further counter measures needs to be taken to account to achieve redundancy and hence some added cost.

Also, since memory is not shared, each node uses it own RAM for locks and buffer pool. As a result, “explicit coordination of data sharing across the machines is needed. SD depends on a distributed lock manager facility and a cache-coherency protocol for managing the distributed buffer pool.” [10, page 171] This poses more software bottleneck challenges to the system. However, some vendors implement a shared-cache design to help cope with this drawback. The shared-cache benefits the system by, allowing nodes to primarily check if a requested data is in the cache, then checks the cache of any other node in the cluster and finally get the page from the disk if it is not found from the first two options. At last, there is a limited scalability issue with SD because as the system grows, there is an increase in the I/O functions which result to the bus contention between the CPUs and the disk. This raises more bottleneck problems for the system [9, page 2].

5.3. Shared-nothing Architecture

As the name states, nodes in the SN architecture do not share any hardware resources. As a result, there is minimal interference of the hardware resources which leads to an improved performance. Data is shared among nodes by means of horizontal partitioning hence every node stores a subset of the entire data. When data is then requested, all nodes access the query to discover what is required. The node then creates a request to search the data that is stored locally. Since data is spread across several nodes, all necessary nodes execute their requests in a parallel way [10, page 168].

Every node is responsible for its own lock table and buffer pool eliminating the need for complicated locking mechanism like in SD where the locking spreads over the network. [9, page 3]. As a result, SN is regarded as the best scaling architecture because it reduces interference of the network and only sends questions (queries) and answers (results) through the network [12, page 6].

On the other hand, this design is more challenging to implement than the other architectures. First and foremost, availability of the system needs to be planned and implemented well because; any loss of a node will in return lead to loss of data. To avoid any losses, redundancy with failover strategies is implemented by configuring replication. Every node in the cluster holds at least one copy of data for another node.

Also, SN only offers static load balancing by using fixed partitioning schemes. As a result, the load balancing cannot adapt to business requirements like the dynamic load balancing than SD design provides [13].

The fig 9 below shows an example of the shared-nothing architecture design.

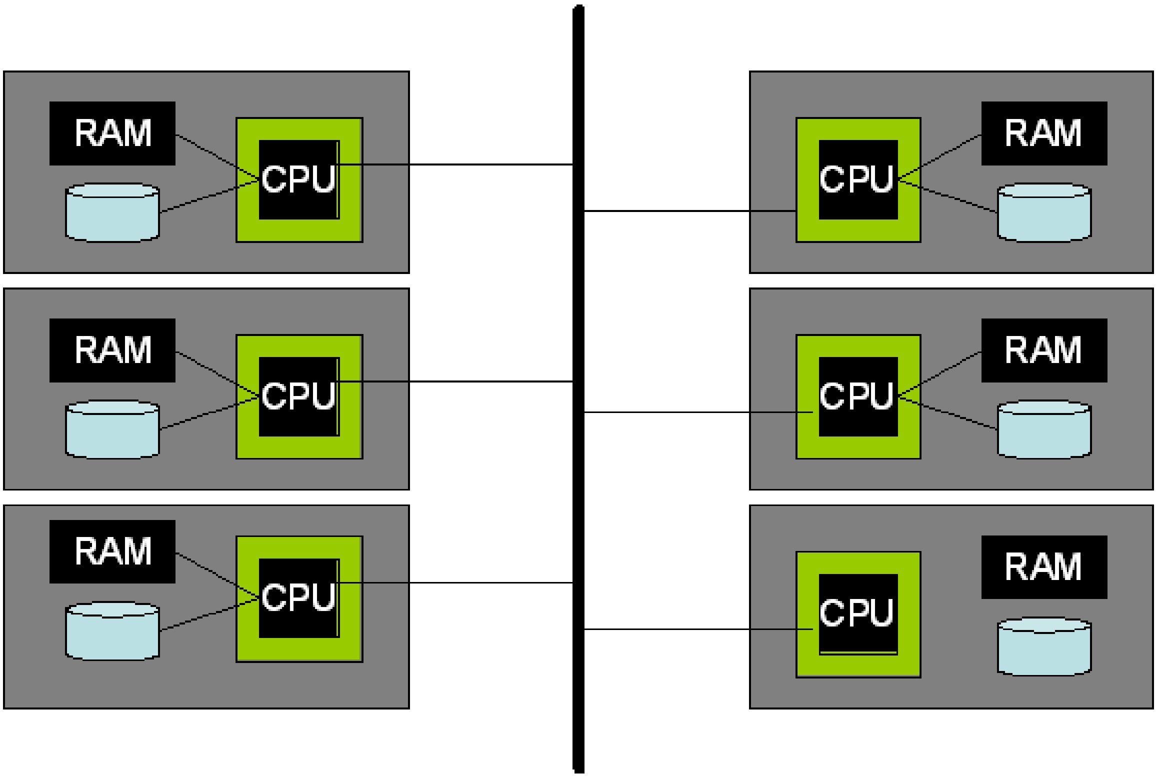


Fig 9: Shared-nothing architecture [10, page 167]

The cluster in the example above comprises of six nodes each with its own resources. This architecture has been implemented on various vendor DBMS for example MySQL.

6. MySQL Clustering

As mentioned earlier MySQL clustering was introduced in early 2004 to help improve availability, scalability and reliability. It was based on the shared-nothing technology which implies that each computer within the cluster uses its own resources like memory, storage, CPU, among others. This technology enables the database system to scale easily since it is easier to delegate tasks to separate hosts without slowing down the network [14, page 144].

A cluster integrates the standard MySQL server with an in-memory clustered storage engine called NDB (Network Database). It is made up of at least four nodes each handling certain tasks for instance; one node for the management, another node serves as a normal MySQL server for accessing the cluster and two nodes for the data storage.

Nodes in a cluster need to be interconnected and TCP/IP is the mostly used protocol with the standard 100 Mbps or faster Ethernet hardware. Other protocols like SCI (Scalable Coherent Interface) could be used but, since it requires special software and hardware, it is not recommended [15].

6.1. Benefits of MySQL clustering

6.1.1. High Availability

Availability is a measure of the system's overall ability to offer resources and services despite any faults. The entire business model in the modern society revolves around information and the access to it [16]. Availability of information has therefore become very important especially for mission critical and real time applications. Availability can be measured theoretically with the five nines (99.999%) rule. This implies that a database attains a 99.999% uptime all year long. Table1 below show some acceptable downtime with this type of measurement.

Table 1

Uptime Percentage	**Acceptable Downtime**
99%	3.65 days
99.9%	8.76 hours
99.99%	52.56 minutes
99.999%	5.26 inutes

6.1.2. Reliability

Reliability is a measure of the number of faults a system experiences within a given time interval. Clustering helps improve reliability by providing failover strategies. This is implemented by ensuring that every node acts a back up of the data that the other node in the same group holds [17]. As a result, the failure of a node does not affect data access hence the system user continue working as normal.

6.1.3. Scalability

With the shared-nothing design, each node is independent of memory, storage and processing power. As a result, only questions (queries) and answers (results) needs to be moved over the network while the processing and disk access are done locally at each node. This allows a more scalable design by minimizing traffic in the interconnection network [18].

6.2. MySQL Clustering Architecture

As mentioned earlier the MySQL Clustering involves the use of NDB (Network Database) storage engine. This engine allows a MySQL Server to delegate storage and retrieval of data to MySQL Cluster data nodes, offering the scalability and high availability features that are characteristic of a MySQL Cluster [19]. The cluster topology is made up of three sections, the clients section, MySQL server and the storage engine. Fig 10 below represents an example of a MySQL cluster.

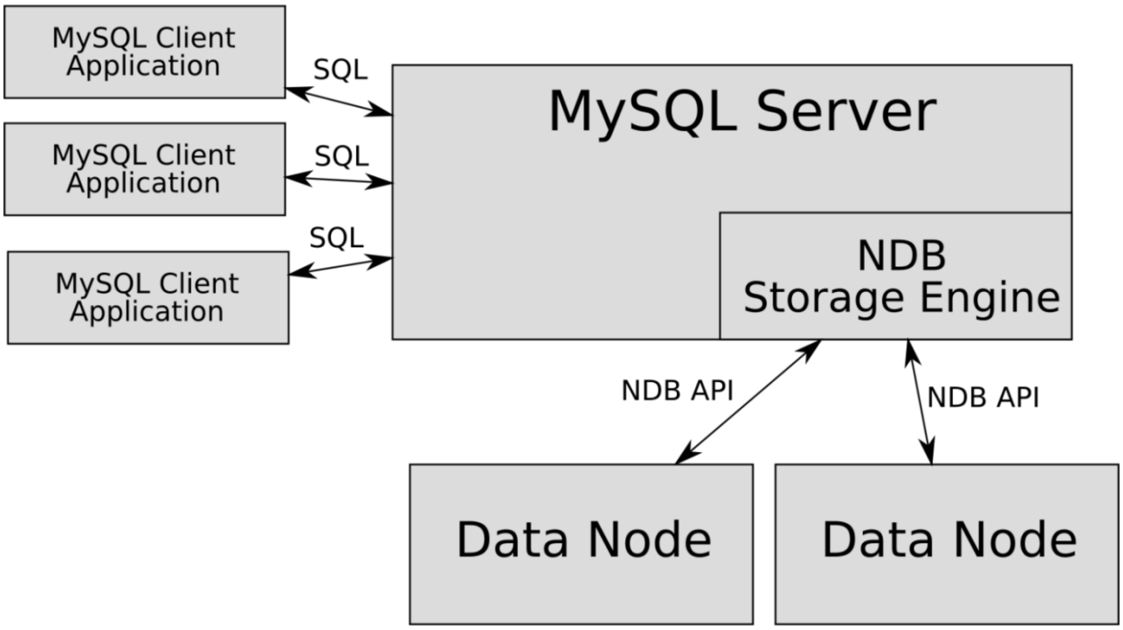


Fig 10. MySQL cluster topology [19]

The clients sends request to the server inform of SQL statements. The server passes the request to the NDB storage engine which locates the data node that holds the requested data. However, Data nodes in MySQL cluster have the ability to implement synchronous data partitioning. This means that it may not be possible to extract all the data from one data node. The NDB API (Application Programming Interface) was then implemented to help control data nodes. As a result, the NDB storage engine maps all storage engine API calls performed by the MySQL server to a set of NDB API calls. Each NDB API call may result in data being returned from more than one data node [19].

6.2.1. Nodes

The name node is used to refer to a process or the hosts in a cluster. There are three types of nodes; management nodes, data nodes and SQL nodes. The management node usually manages all the other node, the data node is mainly for storage of cluster data and the SQL node is used by clients to access the cluster data.

6.2.2. Management node

It is the node that manages all the other nodes in the cluster by providing configuration data, starting and stopping nodes, running backups, among other tasks. This node must be started first since other nodes depend on it at startup [17]. Once the cluster is up and running, the management node is not an essential part of the cluster. In most cases, a single node is enough for the cluster but additional nodes can be configured to increase availability [20]. This node can be started using the command `ndb_mgmd`.

6.2.3. Data node

A cluster comprises of several data nodes depending on the cluster requirements. They are responsible for storage of the entire cluster's data. The command *ndb* is used to start the data node process. Several data nodes can be combined to form a node group [21].

6.2.4. Node group

A node group is a node or a set of data nodes combined to form a group that stores partitions data and replicas. All node groups configured within a cluster must have the same number of nodes. The number of node groups configured is deprived from the formulae:

[*number_of_node_groups*] = *number_of_data_nodes* / NumberOfReplicas

For example; for a cluster with 4 data nodes and the number of replicas set to 2 in the configuration.ini file, there are 2 node groups: 4 / 2 = 2.

Every node within the group stores a specific potion of the partition data. Replicas are then generated and stored in a different node within the group to improve availability. Fig 11 below represents an example that will help us understand this concept.

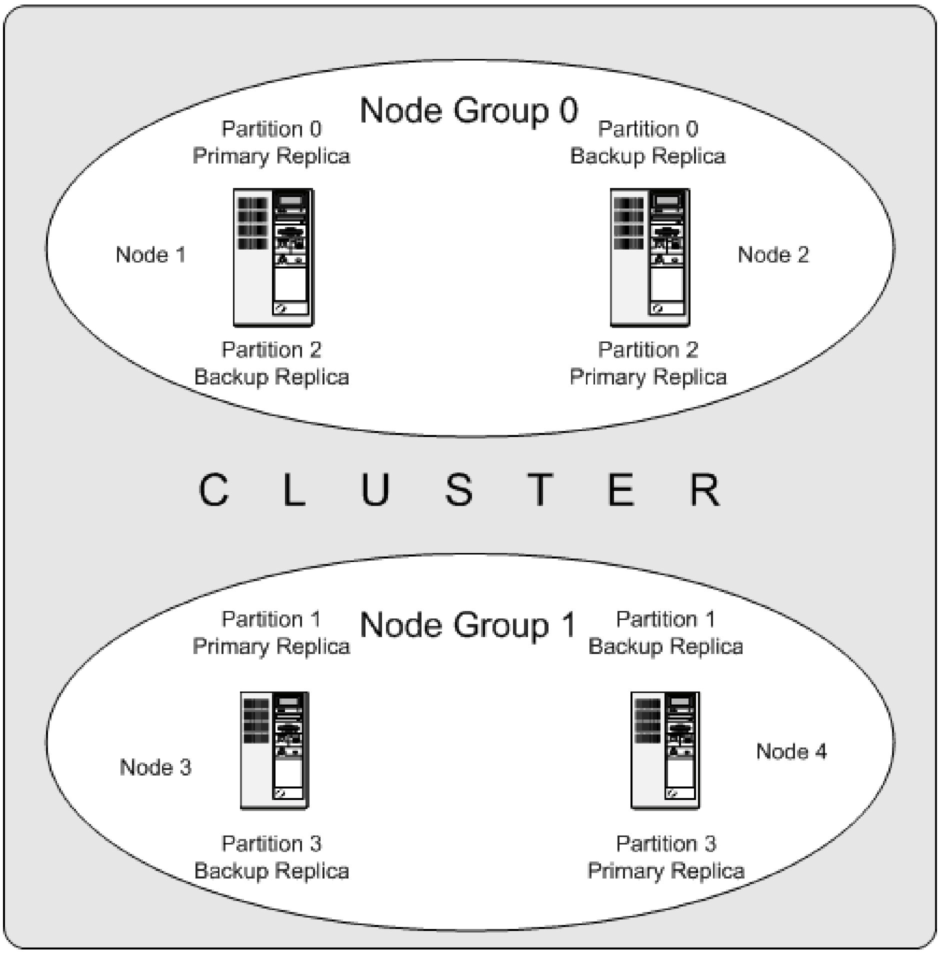


Fig 11. A cluster example [21]

From the example, node group 0 has two nodes; node1 and node 2. Node 1 stores partition 0 original data and acts as the backup of node 2, which is Partition 2 data. Node 2 stores partition 2 original data and acts as the backup of node 1, which is partition 0 data. Consequently, the same applies for node group 1 where node 3 stores partition 1 original data and acts as the backup of node 4. Node 4 stores partition 3 original data and acts as the backup of node 3 [21].

As a result, the Cluster remains active even when any node fails from either node groups because every node has a replica and hence all the data in that node group. For instance from the figure above, any one node in group 0 and one from group 1 are enough to keep the cluster running and hence, high availability in enforced in the cluster.

6.2.5. Partitioning

Partitioning is the act of dividing data into distinct portions called partitions. It reduces the amount of data read for particular SQL operations so that overall response time is reduced [22]. There are two types of partitioning; horizontal and vertical partitioning.

Horizontal partitioning involves dividing rows of data to separate nodes. It can be used for tables that contain so many rows like several years long data.

Vertical partitioning involves splinting wide tables to separate nodes. Each set of columns contains all the rows intact.

MySQL employs horizontal partitioning which enables it to provide increased performance and simplified data management [22].

6.3. Pros and Cons

To begin with, the SN design that MySQL uses is very scalable. Since no resources are shared, the network is used to transfer queries and responses other than intensive data between nodes, and between nodes and the shared storage. It also provides redundancy through replication hence high availability.

Secondly, MySQL clustering is very cost effective to implement both in terms of hardware and software. It utilizes low commodity hardware with the SN design and does not require any special hardware. Consequently, MySQL is an open source software and it can be freely downloaded and installed. However, for commercial purposes, some charges may be incurred but they are not anywhere close to other software like Oracle RAC and Microsoft SQL server licenses.

On the other hand, MySQL relies on horizontal partitioning for load balancing. This may not always be sufficient because the partitioning scheme is static. Additionally, a failure of more than one node in the same node group might render some data inaccessible or even lead to loss of data.

7. Oracle Real Application Clusters

Oracles use RAC (Real Application Clusters) to provide high availability, scalability and high performance for modern systems. It implements the shared-disk architecture where all nodes share the same storage. The traditional shared-disk design posed some challenges for the oracles RAC especially because all nodes had to read the data from the disk. This meant that if a node requires a block of data that is already been used by another node, then it had to wait until the block is written back to the disk. A new technology called cache fusion was developed that enables nodes to share the data in the cache and therefore have direct access to each others cache content using the cache fusion protocol. This resulted to a shared-cache design which provided improved performance by reducing the response time because nodes could easily get required data directly from each others cache [23].

7.1. Hardware components of Oracle RAC

The main hardware components of the RAC are similar to what was earlier discussed in the shared-disk section. A cluster comprises of nodes which are connected via a high speed interconnect. All nodes are connected via a network switch or a hub. They then connect to a common storage which can be a NAS or SAN. Fig 12 below shows a clear representation of this components and how they are connected.

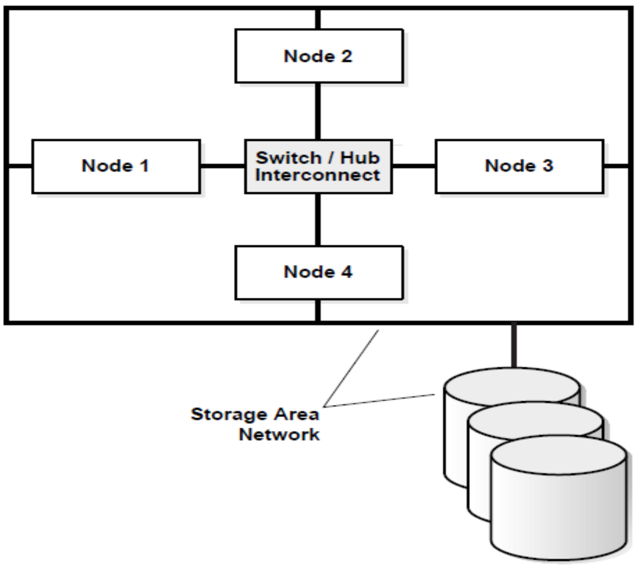


Fig 12: RAC hardware components [24]

As shown above, the cluster is made up of four nodes and they all use the same storage which is implemented using a Storage Area Network.

7.2. Software components

There are two main components of the RAC software that manage all the cluster operations. The OSD (Operating System Dependent) clusterware and the GCS (Global Cache Service).

7.2.1. OSD clusterware

The clusterware is dependent on the operating system being used. Oracles provide the clusterware for windows 2000 and NT while other vendors provide the OSD for other operating systems like UNIX [24, p 44]. It acts as the middleware of the entire cluster by managing the cluster, monitoring the nodes and controlling the interconnect.

7.2.2. Global Cache Service (GSC)

With the new cache fusion employed by Oracle, the GSC acts as the engine that manages how instances (nodes) share the data blocks in the cache. It ensures that only one instance at a time can modify the data. Fig 13 below provides an overview of how the GSC controls instances in a RAC.

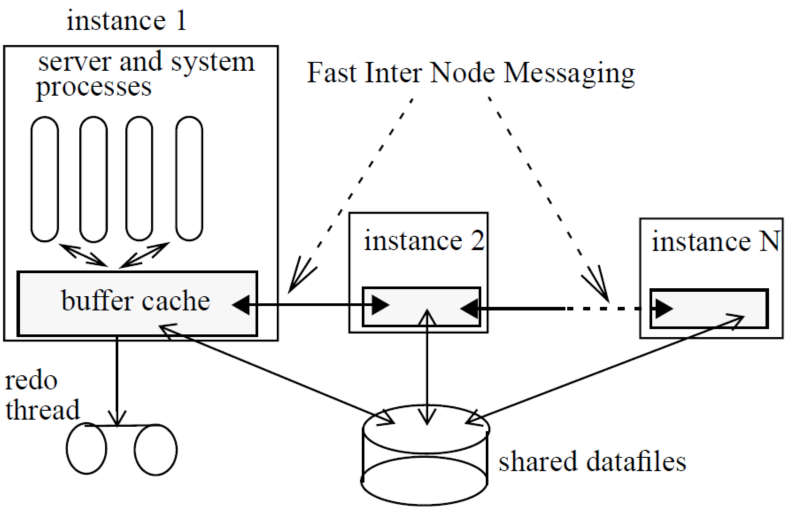


Fig 13: Oracle RAC instance [23]

The GSC and the nodes use the cache fusion protocol to send fast inter node messages to each other. For example, if Instance 2 wants to update some data, it will forward the request to the GSC which will in turn forward the request to the instance holding the data. The holder of the data will then pass the data directly to the requester and finally the GSC will update holder information to indicate the new holder [23].

7.3. Pros and Cons of RAC

Firstly, it provides high availability using the shared-disk design. In shared-disk, the cluster continues operating even in times when a node fails. The clients connected to that node are automatically transferred to the remaining nodes.

Secondly, RAC provides a low cost of ownership because it utilizes the low cost commodity hardware other than expensive supercomputers.

Also, RAC offers expanded scalability because nodes can be easily added to improve performance when needed [24, 33].

Finally, the new cache fusion technology enables the cluster to offload some I/O operations from the shared-disk by sending inter-node messages for data within the shared-cache. Additionally, shared-cache helps during the recovery procedures by providing the data that it holds [23].

On the contrary, the RAC shared-disk design provides a single point of failure because all the data is stored in the same medium. Oracle offers data guard to cope with this problem where a standby database is used. The log shipping technology keeps the standby server synchronous with the primary server but Oracle charges full price for this [25, 4].

Consequently, Oracle claims that RAC offer high scalability for up to 100 nodes but this is only true in theory. Bryan Thomas, an expert in Oracle databases states that he has seen little evidence of this. He adds that most RAC implementations have 2 to 4 nodes and it is clear that most customer use RAC for high availability rather than scalability [25, 3].

In conclusion, the RAC software is very expensive (about US $20k per processor) and therefore offsets any cost saved by the low commodity hardware [25, 3]. However, it has had its praise in stability and therefore many companies find it worth to implement.

8. Microsoft SQL Server Failover Cluster

The initial SQL server cluster was introduced in version 6.5 but it was very crude and was rarely implemented in the real world. Later, version 7.0 was released with a major improvement of clustering and hence the first promising SQL server cluster. It is based on the shared-disk architecture where all nodes share the same storage. Therefore, there is need for extra redundancy effort for the storage incase of a disaster. There are two types of cluster, Active/Active clusters and Active/Passive cluster [26].

8.1. Active/Active Cluster

In Active/Active cluster, each node in the cluster runs its own instance of SQL server and hence two instances of SQL server are running. In this implementation, when a node fails, the other node can take over all the tasks ensuring that no downtime is incurred. However, clients may experience a slight degradation in performance since both SQL instances will now run on a single node. On the other hand, this design utilizes all the resources in the cluster since all nodes are active [27].

8.2. Active/Passive Cluster

In the Active/Passive clustering, only one node is active while the other acts as a standby node and will take over at time when the active node is down. It is recommended that both or all node must be identical it terms of resources so that to ensure a smooth failover transition. The hardware requirements for this implementation may be higher than the former because only one node is active at a time. Also, this implementation does not utilize all the hardware resources since only one node is running at a time. On the contrary, Microsoft does not require a license for the passive node hence the license cost is lower [27].

8.3. Hardware requirements

As already discussed earlier in the shared-disk architecture, most of the hardware components of the failover clusters are similar. It is however important to put into consideration the type of cluster you choose to implement while selecting the hardware for the nodes. Also consider what kind of hardware to use for setting up backup for your storage. In addition, Microsoft dictates that any choice of hardware selected must be in its HCL (Hardware Compatibility List) [28]. As a result, SQL clustering may not utilize the commodity hardware that fails the compatibility check.

8.4. Software Requirements

The software requirements depend on the number of nodes that one intends to implement in a cluster. For two nodes, Windows 2000 advanced server SP4 (Service Pack 4) can be used. If you need more than 2 nodes, Windows 2000 datacenter server SP4 or Windows 2003 server enterprise edition support up to 4 nodes. Also, the 32 bits version of Windows 2003 server Datacenter edition supports up to 8 nodes while the 64 bits version only support 4 nodes [29]. It is important to remember that, a license is needed for every node in the cluster. Therefore, implementing a cluster with four nodes using Windows 2000 Datacenter Server will require four licenses. In addition, the SQL server software is needed and a license is required for every node in an Active /Active implementation. However, only one license is needed for the Active/Passive implementation [30].

8.5. Pros and cons

The only compelling reason for using SQL server clustering is that, it is considerably easy to install, administer, and maintain [31].

However, the shared-disk design does not protect your storage medium hence extra back up procedures must be used to achieve redundancy of the SAN or NSA. The best way to cope with this problem of redundancy would be by adding database mirroring to your cluster. Mirroring involves creating a secondary copy of the primary database in a secondary server. Entries from the transaction log of the primary database are transferred and applied to the secondary server hence ensuring that both servers are kept synchronous at all times [32]. In this case, the cluster would be the primary server and a secondary server would be employed for the mirroring resulting to redundant system. Nevertheless, extra expenses will be incurred for the secondary server and licenses as well making SQL clustering more expensive than other DBMS clustering for example, MySQL.

9. Different Implementation

We have looked at different clustering techniques for both web servers and databases in the previous sections. However, the underlying platform on which the software is installed must have clustering capabilities for the system to function as expected. In this section two most widely used operating systems that include clustering capabilities were evaluated.

9.1. Microsoft Windows Clustering

The first Microsoft windows cluster was introduced to the market in 1995 under the Window NT 4.0 enterprise server software. This product did not meet the customer's business demands and other software providing better performance and scalability overshadowed it. In the year 2000, Microsoft released Windows 2000 which also had a version of their server, Windows 2000 Advanced server and Windows 2000 Datacenter server. Both server software offered Microsoft clustering services and NLB (Network Load Balancing) [33].

Other server software have been released since then like Windows server 2003, both enterprise and datacenter. These later products offer an added clustering option, CLB (Component Load Balancing).

Microsoft clustering services offers failover clustering support which is more about availability than scalability. It implements the shared-nothing design which means that every server uses its own resources. However, in this implementation, all servers are configured to access the same storage but only one node uses all the resources at a time. All the other nodes act as standby nodes monitoring the primary node and incase of failure one of the surviving nodes takes over [34].

NLB cluster is mainly used for web clustering. It acts as a front-end cluster that receives client's requests and forwards them to the servers (nodes) in the cluster. Windows server 2003 supports up to 32 servers sharing the same virtual IP address in the cluster. NLB provides high availability to clients by enforcing failover strategy as well as incremental scalability [34].

CLB provides dynamic load balancing for COM+ components that are attached to separate servers. CLB can balance the load up to 8 nodes therefore improving on the components performance.

The number of nodes supported in a cluster depends on the software used by the nodes as show in the table below.

Table 2. Different Windows Server software and the number of nodes supported [35]

Software	Number of Nodes
Windows NT 4.0 Enterprise Edition Windows 2000 Advanced Server Windows 2000 Datacenter Server Windows Server 2003, Enterprise Edition Windows Server 2003, Enterprise x64 Edition Windows Server 2003, Datacenter Edition Windows Server 2003, Datacenter x64 Edition	1-2
Windows 2000 Datacenter Server	1-4
Windows Server 2003, Enterprise Edition Windows Server 2003, Enterprise x64 Edition Windows Server 2003, Enterprise Edition for Itanium-based Systems Windows Server 2003, Datacenter Edition Windows Server 2003, Datacenter x64 Edition Windows Server 2003, Datacenter Edition for Itanium-based Systems Windows Server 2008, Enterprise Edition Windows Server 2008, Enterprise Edition for Itanium-based Systems Windows Server 2008, Datacenter Edition Windows Server 2008, Datacenter Edition for Itanium-based Systems	1-8
Windows Server 2008, Enterprise x64 Edition Windows Server 2008, Datacenter x64 Edition Windows Server 2008 R2 Enterprise Windows Server 2008 R2 Datacenter Microsoft Hyper-V Server 2008 R2	1-16

As displayed above, different software provides varying support for clustering. If two separate software for example Windows 2000 Datacenter Server (support 4 nodes) on one node and Windows Server 2003 enterprise (support 8 nodes) on another node, are used in the same cluster, the total number of nodes supported will be four nodes [35].

Before any cluster installation in windows, all the hardware must be counter checked with the windows HCL (Hardware Compatibility List). This implies that certain commodity hardware might not be used [36].

9.2. Linux clustering

The original Linux cluster was developed in 1994 by Donald Becker and Thomas Sterling for NASA. It was called the 'Beowulf cluster', comprising of 16 nodes [37]. Modern Linux clusters can scale to even thousands of nodes and hence are implemented in most large organizations.

Today, Linux clustering technology provides, High Availability (HA) clusters, and High Performance Computing (HPC) clusters.

9.2.1. High Availability (HA) clusters

HA clusters, are mainly implemented where a system needs to be highly available like for databases and web server farms. Two or more servers (nodes) are used and they mirror each others functions. This can be categorized into two schemes.

The first one could implement one node as the active server and the other as a standby waiting to take over incase of failure.

On the other hand, both nodes could be active hence sharing the load. In this design, care should be taken to ensure that the load stays under 50% in capacity so that incase of one nodes failure, the remaining node can take all the responsibilities without being overloaded. This design also uses the shared-disk design. It is mainly implemented for load balancing especially in web-server farms [38, 5].

9.2.2. High-Performance Computing (HPC)

HPC clusters would be used in situations where a high throughput and performance are required. This was the initial idea of the Beowulf cluster where nodes in the cluster works together by implementing parallel processing algorithms, to process high intensive tasks that demand high computing power for example, data mining, internet search engines, image processing, among others.

In HPC, a master node distributes all the workload to the slave nodes resulting to a parallel processing. This design can use one of the two message passing mechanisms for inter-process communication [39, 5].

The first one is the PVM (Parallel Virtual Machine) where one node runs the PVM console. The PVM console has information of all its slave nodes and can distribute the workload.

The second mechanism is the MPI (Message Passing Interface) is used when applications need to interact with several CPUs on different motherboards. This physical trait of MPI makes it more popular for most HPC clusters [38, 33].

10. Project Recommendation

So far, different clustering implementations that could have been required for the Archipelago website project have been evaluated in the previous sections. The main goal was to develop a solution that would be highly available, scalable and that would offer a high performance. This section provides an insight on what techniques would have suited this project.

The three parts of this section provide recommendations for the three major areas where clustering was needed. The first section considers which platform would have been better, that is, either Windows or Linux. The next section addresses the web server clustering technique and finally, the last part evaluates which database program suited the project best.

10.1. Windows or Linux?

Both operating systems offer a variety of impressive clustering techniques. However, one platform out ways the other in terms cost and performance. This section examines the main competitive aspect of both clustering techniques, and by comparing them, the search for the solution was narrowed down.

To begin with, Windows clustering services are easy to build and manage as compared to Linux clusters. A lot of expertise is needed to make a Linux cluster perform as expected. This was mainly a set back some years ago but currently, there are so many experts on this area who can make the Linux cluster installation easier.

Secondly, Windows clustering can scale up to 16 node for standard clusters and 32 nodes for the load balancing clusters. This may not be sufficient at times especially when designing a system that will not change in many years to come. Continuous expanding scalability is a very important aspect and the scalability in Linux is unquestionably impressive. Linux load balancing clusters can scale to thousands of nodes for example, the Google search engine cluster comprises of about 15,000 nodes, among others. Additionally, for standard clusters, Linux offers better scalability than Windows.

Another selling point for Linux is that it is open source software and therefore developers are flexible to alter the source code to suite their needs.

Finally, the high prices of Microsoft's Windows licenses might be too expensive especially when there is a cheaper and better alternative. In addition, Microsoft poses a restriction of the commodity hardware that can be used because all hardware must be in the hardware compatibility list of its products. This may add some costs of required commodity hardware during installation

In conclusion, Linux offers the best, scalability, performance and availability features at a very comfortable cost. It also offers a lot of flexibility that would have come in handy with the Archipelago project.

10.2. Web Clustering Solution

The L4/2 and L4/3 clustering offer a commendable performance once implemented with a dispatcher that uses a broader algorithm like WRR (Weighted Round-Robin). In this case the dispatcher can make more intelligent decisions about which node in the cluster will service the client's request. However, the competitive features of L7 cluster out way L4/2 and L4/3 clustering techniques.

The L7 uses more dynamic policies that can be customized to suite the systems need. This ensures that any mission critical request are given priority and hence no delays are incurred that would render the requests insignificant. Also, the L7 dispatcher is able to make more intelligent decisions about which node to distribute a specific request. Some dispatchers in this implementation use the LARD algorithm while others use the newer MC-RR algorithm.

The former algorithm examines clients request received by the dispatcher and categorizes them according to their needs. The request is then forwarded to the most suitable server.

On the other hand, the later algorithm uses some defined classes of traffic that are based on the resources required. The classes are then forwarded to the back-end servers ensuring that the weight is evenly distributed.

Therefore, implementing L7 with MC-RR algorithm would have been the best option for the web server cluster of the project.

10.3. Which database to prefer?

Both Microsoft SQL server and Oracles clustering have been implemented using the shared-disk architecture. However, they both differ a lot in operation and in terms of cost.

The only selling point for SQL server is that the licensing cost is cheaper than Oracle's RAC. Nevertheless, SQL failover clusters do not offer scalability since only one server is in operation at a time. The Active/Active clusters may offer an improved performance but care should be taken while selecting the hardware because incase a node fails the remaining node must have enough resources to keep the cluster running.

Oracle's RAC shared-cache design provides a more stable cluster with very high availability. Even though, the SAN or NAS may provide a single point of failure, there is no doubt that Oracle's RAC will always outperform SQL server clustering and should be considered regardless of its high cost.

On the other hand, MySQL implements the shared-nothing architecture. This design offers more scalability because all nodes are independent and they do not share any resources. Each node stores and manages its own data that is distributed by means of horizontal partitioning. Nodes also store back copies of each other hence providing high availability in time when a node fails.

Conversely, the load balancing scheme in MySQL cluster is only based on partitioning scheme and hence not as dynamic as what Oracle's RAC provide. In addition, Oracles RAC wins in the battle of High Availability. Since data is stored in a central place, any node failure has insignificant effects to the cluster.

In conclusion, the choice between these two products might be a hard one because they are both very competitive. However, MySQL being open source software provides the cheapest and reasonable option. Its performance is also very commendable even for the project at hand especially where both scalability and availability are important. In my opinion, MySQL would be the best solution to implement the underlying database system for the Archipelago website project.

11. Conclusion

The main goal of this thesis was to search for the most cost effective clustering solutions for the Archipelago website project. The purpose of the study was to provide recommendations incase the project was to be implemented.

The Archipelago site was focused at improving the economy of the Central Baltic regions of Finland, including the Åland Islands, Sweden and Estonia. As a result, there was need for developing a system that would provide high availability, scalability and high performance.

Availability would ensure that the system is tolerant to any disasters that would incur some downtime which would in return lead to high losses. Scalability would ensure that the system could grow as more and more people and companies invest in the project, and finally, high performance would guarantee that as the system grows it could not surfer from resource starvation which would render it unreliable.

By implementing a clustering solution, all of the above goal would have been achieved. During this study, different clustering techniques were reviewed for the database, web servers and the underlying platform software.

In the end, a solution that involved MySQL clustering as the database program, using L7 clustering with MC-RR, and all implemented under Linux operating system, provided the best cost effective solution for the project. Even though, Oracles RAC offered a very competitive solution for the database, its high price provides a disadvantage because a reasonable solution was available using MySQL.

12. REFERENCES:

[1]. Gropp William, Lusk Ewing, Sterling Thomas, 2003, *Beowulf cluster computing with Linux*, Second Edition, The MIT Press, Cambridge , Massachusetts London, England

[2]. Barroso Luiz André, Dean Jeffrey, Hölzle Urs, Google, 2003, "*Web search for a planet*", The Google cluster architecture, IEEE Computer Society, Available: http://static.googleusercontent.com/external_content/untrusted_dlcp/labs.google.com/en//papers/googlecluster-ieee.pdf

[3]. Zeis Chris, Wessler Michael, Ruel Chris, *Oracles 11g for Dummies,* 2009, Wiley Publishing, Indianapolis, Indiana

[4]. MySQL AB , April 2006, MySQL *administrator's guide and language reference,* MySQL AB, United States of America

[5]. CCNA2 Curriculum

[6]. Chen Chien-Hung, Hou Chun-Liang, Lee Shie-Jue , '*A Context-based request dispatching policy for Layer-7 web cluster*', Department of Electrical Engineering, National Sun Yat-Sen University, Taiwan

[7]. Shroeder T., Gddard S., Ramamurthy, B., May/June 2000 , 'Scalable *webserver clustering technologies*', IEE Network, vol. 14 no. 3, pp. 38-45

[8]. Casalicchio Emiliano, Colajanni Michele, '*Scalable Web Clusters with Static and Dynamic Contents',* University of Rome Tor Vergata, Italy

[9]. DeWitt J. David, Ph.D., Madden Samuel, Ph.D., Stonebraker Michael, Ph.D., '*How to build a High-performance Data Warehouse'*

[10]. Hellerstein M. Joseph, Stonebraker Michael, Hamilton James, 2007, *'Architecture of a Database System'*, Vol. 1 No. 2 , pp.141–259

[11]. Hogan Mike; CEO ScaleDB, '*Shared-Disk vs. Shared-Nothing'*, Comparing Architectures for Clustered Databases, ScaleDB Inc.

[12]. DeWitt J. David, Gray Jim, January 1992 , '*Parallel database systems'*, The Future of High Performance Database Processing

[13]. Stonebraker Michael, *'The Case for Shared Nothing',* University of California, Berkeley, Ca.

[14]. Hellerstein M. Joseph, Stonebraker Michael, 2005, '*Readings in database systems'*, Fouth Edition, The MIT Press, United States of America

[15]. MySQL, MySQL Cluster Nodes, Node Groups, Replicas, and Partitions. [Online]. Available: http://dev.mysql.com/doc/refman/4.1/en/mysql-cluster-nodes-groups.html [Accessed: 10 May 2010].

[16]. MySQL, August 2006, "*A Guide to database high availability*", An Introduction to database high availability concepts and requirements, MySQL AB, Available: http://www.mysql.com/why-mysql/white-papers/mysql_db_high_availability.php

[17]. Sun Microsystems, "MySQL Cluster 7.0:Architecture and New Features", Sun Microsystems, Inc, U.S.A., September 2009

[18]. MySQL, MySQL Cluster Overview. [Online]. Available: http://dev.mysql.com/doc/refman/4.1/en/mysql-cluster-overview.html [Accessed: 11 May 2010].

[19]. Sun Microsystems , MySQL 5.1 Cluster DBA Certification Study Guide. [Online]. Available: http://docs.sun.com/source/820-5417/ccsg-ndb.html [Accessed: 11 May 2010].

[20]. MySQL, MySQL Cluster Core Concepts. [Online]. Available: http://dev.mysql.com/doc/refman/5.1/en/mysql-cluster-basics.html [Accessed: 11 May 2010].

[21]. MySQL, MySQL Cluster Nodes, Node Groups, Replicas, and Partitions. [Online]. Available: http://dev.mysql.com/doc/refman/5.1/en/mysql-cluster-nodes-groups.html [Accessed: 11 May 2010].

[22]. Robin Schumacher, Improving Database Performance with Partitioning. [Online]. Available: http://dev.mysql.com/tech-resources/articles/performance-partitioning.html [Accessed: 12 May 2010]

[23]. Lahir Tirthankar I, Srihari Vinay, Chan Wilson, Macnaughton Neil, Chandrasekaran Sashikanth, "Cache Fusion: Extending Shared-Disk Clusters with Shared Caches," Oracle Corporation

[24]. Bauer Mark, *Oracle9i Real Application Cluster*, Oracle Corporation. March 2002. [E-book]. *Oracle9i Database,* Available: http://download.oracle.com/docs/cd/B10501_01/rac.920/a96597.pdf

[25]. Bryan Thomas , "Solutions for Highly Scalable Database Applications: An analysis of architectures and technologies"

[26]. McGehee Brad, An Introduction to SQL Server Clustering. [Online] . Available: http://www.sql-server-performance.com/articles/clustering/clustering_intro_p1.aspx [Accessed: 16.May 2010]

[27]. Ford Tim, SQL Server Clustering Active vs. Passive. [Online]. Available: http://www.mssqltips.com/tip.asp?tip=1554 [Accessed: 16 May 2010]

[28]. McGehee Brad, An Introduction to SQL Server Clustering. [Online] . Available: http://www.sql-server-performance.com/articles/clustering/clustering_intro_p3.aspx [Accessed: 16.May 2010]

[29]. Policht Marcin, SQL Server 2005 Part 3 - High Availability and Scalability Enhancements - Failover Clustering. [Online]. Available: http://www.databasejournal.com/features/mssql/article.php/3444181/SQL-Server-2005-Part-3---High-Availability-and-Scalability-Enhancements---Failover-Clustering.htm [17 May 2010]

[30]. McGehee Brad, An Introduction to SQL Server Clustering. [Online] . Available: http://www.sql-server-performance.com/articles/clustering/clustering_intro_p2.aspx [Accessed: 17 May 2010.

[31]. Ford Tim, Getting started with SQL server clustering. [Online]. Available: http://www.mssqltips.com/tip.asp?tip=1541 [Accessed: 20 May 2010]

[32]. Policht Marcin, SQL Server 2005 Part 2 - High availability and scalability enhancements - database mirroring. [Online]. Available: http://www.databasejournal.com/features/mssql/article.php/3440511/SQL-Server-2005-Part-2---High-Availability-and-Scalability-Enhancements---Database-Mirroring.htm [Accessed: 21 May 2010]

[33]. Lamb Joseph M., What is a Microsoft Cluster. [Online]. Available: http://www.informit.com/articles/article.aspx?p=25748 [Accessed: 22 May 2010]

[34]. Cavale Mohan Rao, Introducing Microsoft cluster services (MSCS) in Windows server 2003 family. [Online]. Available: http://msdn.microsoft.com/en-us/library/ms952401.aspx

[35]. Microsoft, Maximum number of nodes supported. [Online]. Available: http://support.microsoft.com/kb/288778

[36]. Microsoft, Microsoft cluster service installation resources. [Online]. Available: http://support.microsoft.com/kb/259267

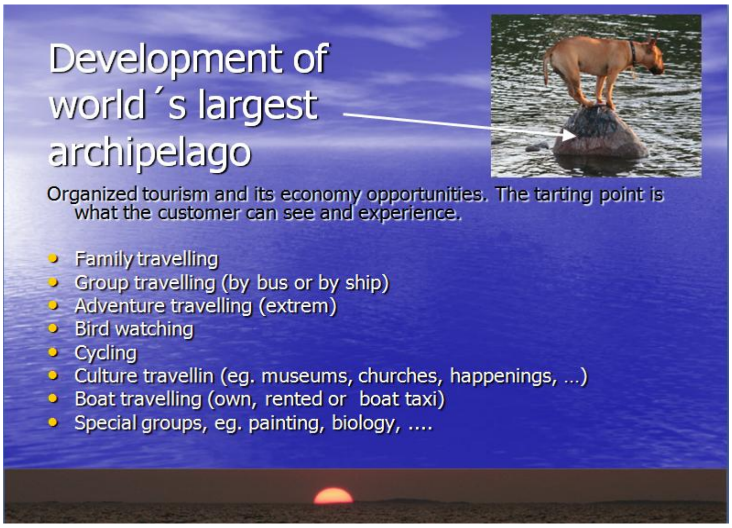
[37]. Merkey Phil, Beowulf history. [Online]. Available: http://www.beowulf.org/overview/history.html

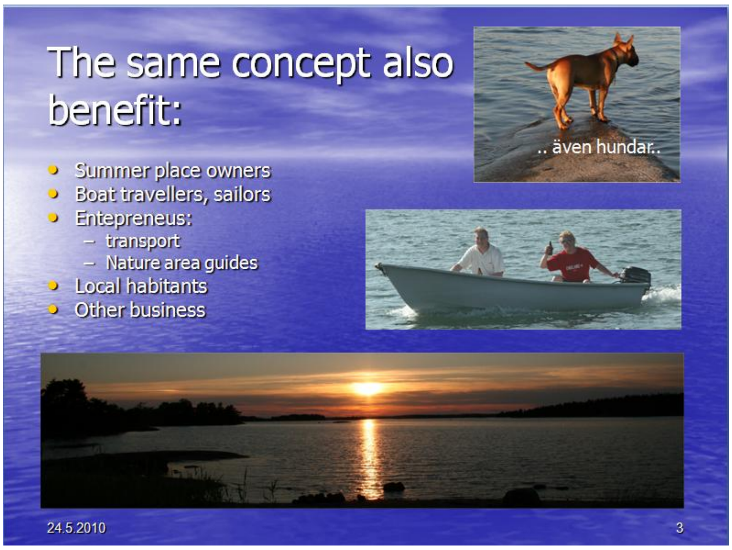
[38]. Blue Tuxedo Team, Linux HPC Cluster Installation, IBM Corporation, June 2001, [E-book]. [Available]: http://www.redbooks.ibm.com/redbooks/pdfs/sg246041.pdf [Accessed: 23 May 2010]

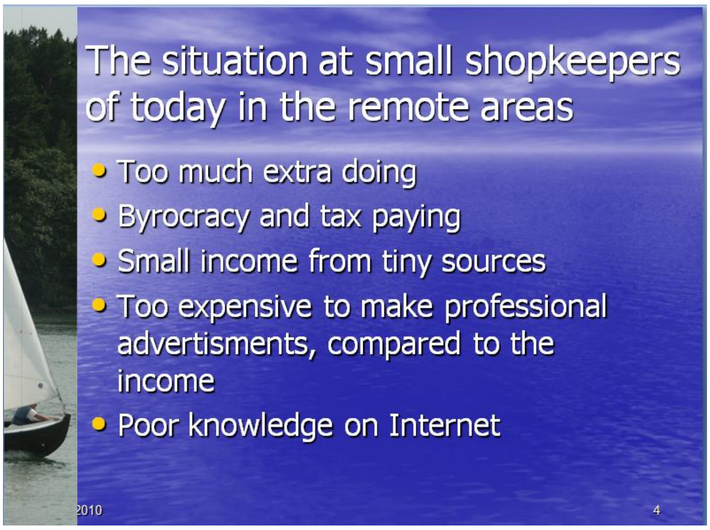
[39]. Luoma Jyri-Matti, "Installation and Administration of Linux Cluster," Bachelor's thesis, Information Technology, Turku University of Applied Sciences, Turku, 2004

13. Appendix 1

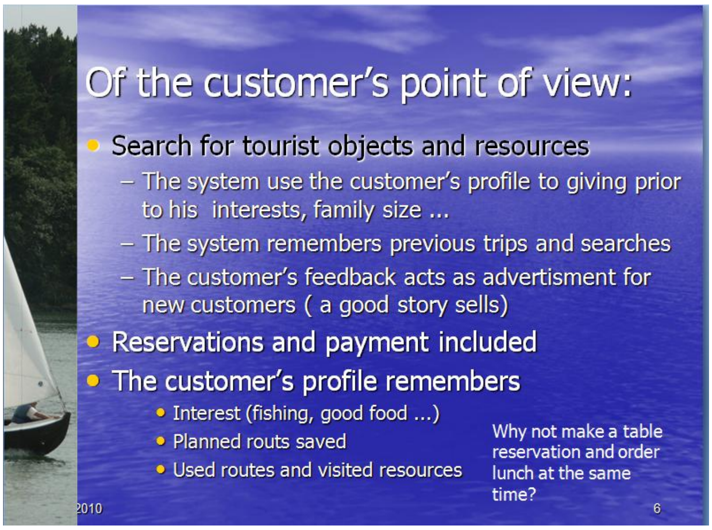
The original Archipelago project concept slides

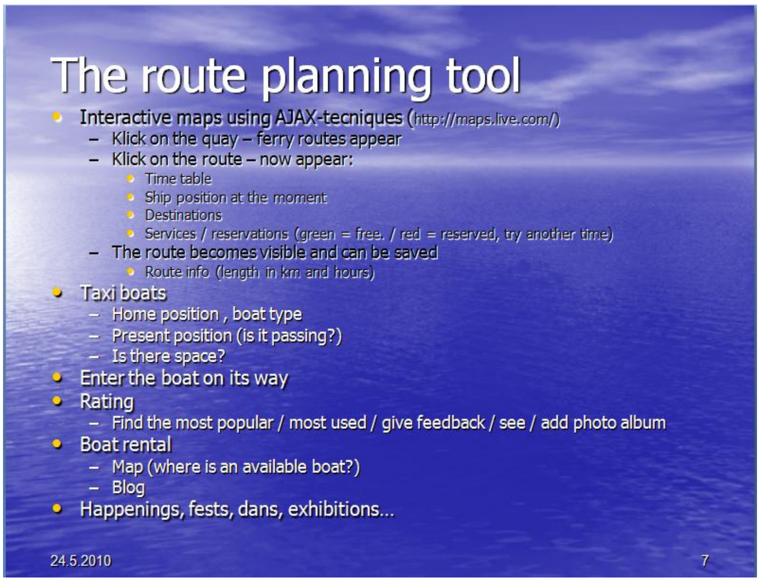




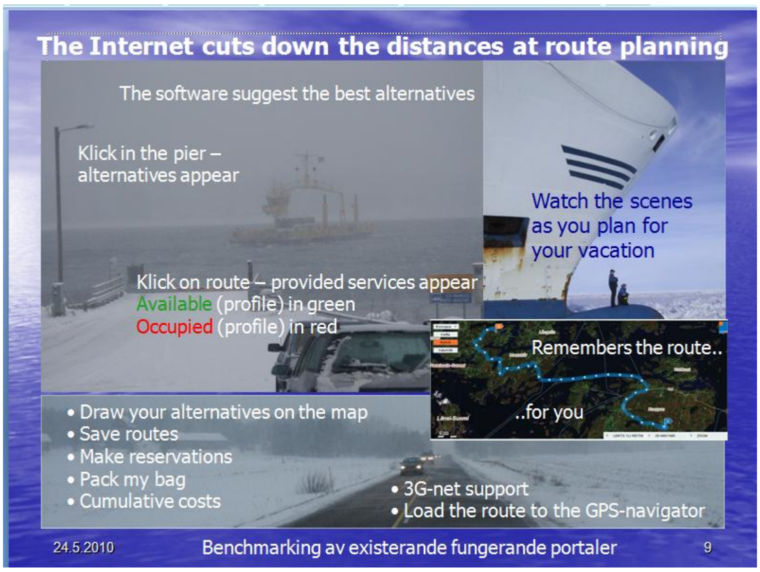

The situation at small shopkeepers of today in the remote areas
Too much extra doing
Byrocracy and tax paying
Small income from tiny sources
Too expensive to make professional advertisments, compared to the income
Poor knowledge on Internet
2010
4


A simple solution is a Internet site which connects the business resources to the demand
Every travel target has its own web site (often embedded to existing)
The software coordinaes
Target / resource searching
Reservation / available resources
Payment
Graphic route planning tool
Automatic internal statistics update
Adapts to reserved resources (makes unavailable at that moment)
The demand type becomes visible (you can see what is searched for)
The statistics is available to all shopkeepers, systematic pricing of resources.
The system develops local communities and impacts networked business
24.5.2010
5


Of the customer's point of view:
Search for tourist objects and resources
The system use the customer's profile to giving prior to his interests, family size ...
The system remembers previous trips and searches
The customer's feedback acts as advertisment for new customers (a good story sells)
Reservations and payment included
The customer's profile remembers
Interest (fishing, good food ...)
Planned routs saved
Used routes and visited resources
Why not make a table reservation and order lunch at the same time?
2010
6


The route planning tool
Interactive maps using AJAX-tecniques (http://maps.live.com/)
Klick on the quay – ferry routes appear
Klick on the route – now appear:
Time table
Ship position at the moment
Destinations
Services / reservations (green = free. / red = reserved, try another time)
The route becomes visible and can be saved
Route info (length in km and hours)
Taxi boats
Home position , boat type
Present position (is it passing?)
Is there space?
Enter the boat on its way
Rating
Find the most popular / most used / give feedback / see / add photo album
Boat rental
Map (where is an available boat?)
Blog
Happenings, fests, dans, exhibitions…
24.5.2010
7

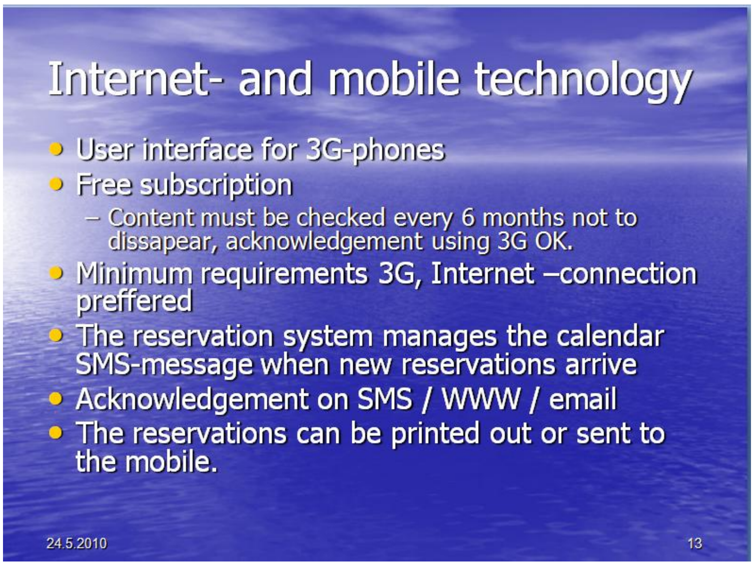

Business advantages:
Easy to create a local business network, one customer reservation shared by several shopkeepers
The supply of services visible all over earth
The younger generation expect everything to be found on the Internet
Easier coordination
The reservations database acknowledge if the service is available at the desired moment
Looks up alternative services in the region
The season extend as the demand controls the supply
It is easier to supply services, even a single row boat is profitable
Payments on the net, credit cards, book keeping service
Demand of a new type of travelling guides
System design includes connections and integration to present data systems and web pages
24.5.2010

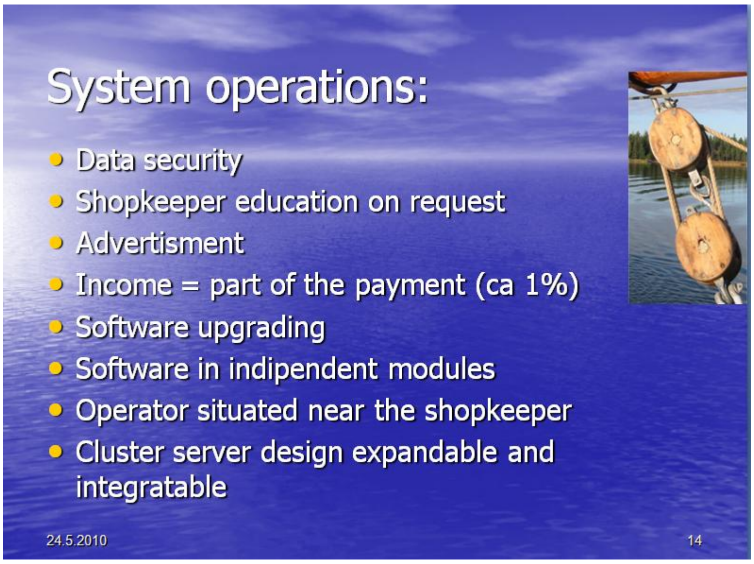

The Internet cuts down the distances at route planning
The software suggest the best alternatives
Klick in the pier – alternatives appear
Watch the scenes as you plan for your vacation
Klick on route – provided services appear
Available (profile) in green
Occupied (profile) in red
Remembers the route..
..for you
• Draw your alternatives on the map
• Save routes
• Make reservations
• Pack my bag
• Cumulative costs
• 3G-net support
• Load the route to the GPS-navigator
24.5.2010
Benchmarking av existerande fungerande portaler
9


Adventures
Everyday episodes of the archipelago is an adventure to the townsfolk
Anything can be productisized
..from father to son..
..even large families..
24.5.2010
10


Lovely views and virtual travelling
Many attractions are not bound to a shop keeper
The attraction is given an own web site
• Photo gallery, webcam
• Environmental and historical info
• Blog
24.5.2010
11


Shop keeper
A formal web page
(or embedded into present)
• Supplied service
• Contact info
• Pictures, videos
• Blog
• Reservations
• Payment
• If not Internet, mobile
• Customers' pictures and stories
• Free subscriprion
• Help desk services
.. Egna produkter till försäljning..
..kestävä kehitys..
.. Reliable services..
.. Keeping clean..
24.5.2010
12


Internet- and mobile technology
User interface for 3G-phones
Free subscription
– Content must be checked every 6 months not to dissapear, acknowledgement using 3G OK.
Minimum requirements 3G, Internet –connection preffered
The reservation system manages the calendar SMS-message when new reservations arrive
Acknowledgement on SMS / WWW / email
The reservations can be printed out or sent to the mobile.
24.5.2010
13


System operations:
Data security
Shopkeeper education on request
Advertisment
Income = part of the payment (ca 1%)
Software upgrading
Software in indipendent modules
Operator situated near the shopkeeper
Cluster server design expandable and integratable
24.5.2010
14


Project sections
Business development
Tourism development
software
Brand and visual design
Mobil communication
Map management
Field work, customer and shopkeeper contacts
System design
Integration
24.5.2010
Skärgårdscentrum
Saaristokeskus

14.


SWOT
Strength:
- Fiber trunk net in the area
- Not too much investments
- Integration of existing technologies
- Technology is available
Opportunities:
- Existing data tecnology
- Knowledge available in the yniversities
- R&D –project
- Integration with existing systems
Weakness:
- The number of potential shopkeepers decreasing
- A good project manager required
- Requires cooperation with many organizations
Thread:
- Maintenance costs?
- Requires good penetration
- Financial issues must be acknowledged at start
- Determination for cooperation
- Resistans against integration
24.5.2010
16


Thanks for your attention
24.5.2010
17