Multi-Channel Clustered Web Application Servers

Masters Thesis Proposal
American University in Cairo

Proposed by **Karim Sobh** (kmsobh@aucegypt.edu)
Supervised by **Dr. Ahmed Sameh** (sameh@aucegypt.edu)

October, 2008
Outline

• General Description
• Thesis Areas of Research
• Background Summary.
• Limitations and drawbacks
• Proposed Solution and new Architecture
• Approach
• Work Done So far
  ➢ Experiments
  ➢ Results
Thesis Areas

- Communication Protocols
- Application Level Protocols
- Transaction Dispatching
- Clustering Techniques
- Fault Tolerance and Check Pointing
- Load Balancing
- Business Logic Isolation
- Web Caching
- Web Traffic generation and benchmarking
It is very important to highlight that this thesis targets web applications that have considerable amount of processing compared to data transfer.

The key idea is how to parallelize delays resulting from processing to gain speed. Thus applications that would benefit most from the new proposed architecture should be decomposable by nature.

This thesis does not target for example streaming environments which are totally based on high data throughput.
Traditional Web Environments Components

- Front End Web Server
- Application Deployment Container
- Application Interface Plug-ins
- Dispatchers and Load Balancers
- Security Components
  - **Network Security** (Firewalls and Access Policy Managers)
  - **Communication Security** (SSL and Trusted Certificates)
  - **Application Security** (User Authentication)
- Web Caches
- Traffic Shapers
Traditional Web Environments Components

Simple Web Environment

Multi-Channel Clustered Web Application Servers

October, 2008
Traditional Web Environments Components
Dispatcher Base Web Environment

Simple Dispatcher Environment

- Ethernet Switch
- Layer-4 or Layer-7 Dispatcher
- Web Servers
  - For Communication, UI Deployment, Business Logic Deployment
  - Using CGIs of Loadable Modules

October, 2008
Traditional Web Environments Components

Single Tier Environment

Layer-4 or Layer-7 Dispatcher

DNS Cluster For Dispatcher DNS Rotation

Heart Beats

Web Servers
For Communication,
UI Deployment,
Business Logic Deployment

Using CGIs of Loadable Modules

Ethernet Switch

Dispatchers 1, 2, 3

October, 2008
Multi-Channel Clustered Web Application Servers
Two Tier Environment

Tier 1
Web Servers
For Communication

Tier 2
Web Application Servers
For UI + Business Logic
Deployment

Multi-Channel Clustered Web Application Servers
Three Tier Environment

DNS Cluster For Dispatcher DNS Rotation

Layer-4 or Layer-7 Dispatcher

Tier 1
Web Servers
For Communication

Tier 2
Web Application Servers
For UI Deployment

Tier 3
Application Containers
For Business Logic Deployment

Ethernet Switch

Dispatcher 1

Web Server 1

Web Application Server 1

Web Application Server 1

Ethernet Switch

Tier 2
Web Application Servers
For UI Deployment

Web Server 2

Web Application Server 2

Web Application Server 2

Ethernet Switch

Tier 3
Application Containers
For Business Logic Deployment

Web Application Server 3

Web Application Server 3

Ethernet Switch

Inter Cluster Communication
Application Level Interaction

HPC Libraries are used like
MPI, PVM, and OPEN-MP

Also Web Services are used
For applications to
Communicate
Traditional Enterprise Web Application Server (2/3)

- Serving over **HTTP** Standards
- Use **TCP** as the network communication protocol
- Powerful standards in service development that help business logic isolation and distribution of their execution
- Web Caching
- Ability to extend through different **Clustering** Techniques
- Different **Dispatching** Techniques
- **Availability, Fault Tolerance, and Exception Handling**
- Mature data exchange standards such as Web Services e.g. **SOAP** and **XML-RPC**
- Service **re-usability** and extension
• The Minimum Unit of dispatching and distribution is a web transaction.
• Expensive communication TCP protocol that requires hand shacking mechanisms as per web transaction
• Dispatching is either expensive (Layer-7) or application unaware (Layer-4)
• Fault Tolerance mechanisms for handling a failing transaction will require a full implicit/explicit re-execution of the corresponding service
• Load Balancing clusters do not make full use of hardware in most cases especially when the cluster is not fully loaded
• Less utilization of transaction back-end distribution provided by technologies like EJBs as all the results of a single web transaction will be queued on the server side
• Caching is applied on static content only
• Single development technology per service
To Wrap Up the Background

• We have enumerated all ranges of web application servers (From a simple web server to the super architecture of a Grid Supper computer)
• We have defined the different components of web application servers
• We have defined the essential building blocks that need to exist in a web application server
• We have defined nice features as well as drawbacks.

Now, we need to introduce the desired architectural changes whose effect on performance will propagate through all the layers of a Web Application Server, and all the different ranges of Web Application Servers we discussed earlier.
Proposed Solution

It can be easily observed that the TCP communication protocol used as a communication layer for Traditional Enterprise Web Application Servers is the main source behind the limitations and drawbacks defined earlier.

This thesis poses a performance based research question which is “What is the effect, on performance, of changing the communication layer of web environments from the normal stateful TCP to the stateless UDP communication protocol, and how will this change help the emergence of new mechanisms and features that will add to performance and to already established characteristics of web environments such as expandability, extendibility, fault tolerance, high availability, ... etc.?"

This change should be done in a delicate way so that all the communication and integration standards from the web clients interfaces perspective are kept unchanged.
Muti-Channel Setup

- Clustered back-end application servers environment is required.
- A single web transaction will be virtually served over multiple concurrent network connections.
- A single transaction can be served physically by different machines
- Multi-threading constructs inside web services scripts to define different threads of execution
- Messages and variables exchange between different threads within the same service
- Service execution can continue by a different cluster node from a specific position through service state migration mechanisms
- The application servers in the new environment will be called Containers
New Architecture

Container

- A Container is the application deployment engine in the new environment
- The new environment is built of a cluster of containers
- It will be able to serve partial requests of a service
- The client of the container is the HPA (High Performance Agent) which will be described in the next slide
- Each Container type supports a programming language, so we will have Java containers, C++ containers, PHP containers, ... etc
- Different types of containers can coexist in the same cluster
New Architecture
High Performance Agent (HPA)

- HPA is a **small agent program** that is installed on the client machine
- HPA acts as a **reverse proxy**
- HPA talks to the web client over **TCP** and to the container over **UDP**
- A web transaction is split by the HPA and served over multiple requests
- HPA provides web caching
- The ideal setup is that the HPA is integrated into the web client
New Architecture

Container - Multi-Channel Over UDP

October, 2008

Multi-Channel Clustered Web Application Servers
New Architecture

Container – UDP & TCP

• UDP is much more lighter, thus larger number of channels can be handled by the container at any point of time.
• It is easy to change the recipient of traffic without recreation of a UDP socket, thus Service State Migration can be implemented.
• In the case of the public WAN (Internet):
  ➢ UDP Containers are transparently deployable
  ➢ Less Control over communication parameters
  ➢ Upper Bound of MTU (Maximum Transmission Unit) will degrade performance
  ➢ High Rates of UDP is considered as attacks by most IDSs which needs a special setup for the deployment environment security components
New Architecture
Case Of The Public WAN

PCs with normal Web Clients

HTTP Traffic over TCP

HTTP Traffic over UDP Channels

MultiChannel Containers Cluster

Inter-cluster Communication Switch

C++ Container

Inter-Cluster Interface

Java Container

Inter-Cluster Interface

Server Side HPA

HTTP Dispatcher

LAN Switch
A cluster is built up of a number of application servers that all collaborate to serve requests coming from the HPA. Multiple HPA can initiate requests to the cluster concurrently.

Different OS/Container Middle ware can coexist in the same cluster, collaborating all together to serve portions of a single request.

Multi-Channel Clustered Web Application Servers
New Architecture

Container – Container Main Subsystems

- Multi-threaded Communication Sub-System
- Service Manager
- Service Discovery Manager
- Configuration Manager
- Administration Manager
- Deployment Manager
- Cluster Management Sub-System
New Architecture
HPA Main Subsystems

- Reverse Proxy Multi Protocol Communication Layer
- Skeleton Cache
- Discovery Client
New Architecture
Service Types

Single Channel Services:
- Smallest Component for deployment
- Called by name
- Cannot be broken down for distribution

Multi-Channel Services: (Server Page)
- Composed of multiple single channel services
- Services within a **server page** can be served concurrently or sequentially based on in-line code tag construct type
- A server page will usually have the extension “.sp”
- Cache-able **skeleton maps** are generated for Serve Pages with the extension “.skel”
- Server Pages utilizes the in-line scripting techniques of normal web server pages with the addition of new constructs for distribution and concurrency
New Architecture
Single Channel Service Scenario

A user opens Firefox and directs its URL location to http://www.domain.com/servicename

It is very important to note that the web browser proxy settings should be set to localhost.

Multi-Channel Clustered Web Application Servers
New Architecture
Multi-Channel Service Scenario

It is very important to note that the web browser proxy settings should be set to localhost.
New Architecture
Service State Migration

A user opens Firefox and directs its URL location to http://www.domain.com/servicename

It is very important to note that the web browser proxy settings should be set to localhost

Web Client
HPA
Container (Active)
File Spooler Service (File.so)

Service State Migration (State)
Cluster Management SubSystem (Active Container)
Cluster Management SubSystem (Standby Container)

Call serialize
Call migrate (State)
Call deserialize (State)

October, 2008
Multi-Channel Clustered Web Application Servers

27
New Architecture

Features

- A **single transaction** can be broken down and its fragments can be **executed concurrently** on different hardware nodes.
- In-line script container sections can be implemented in **different technologies** provided that the cluster has nodes supporting different technologies within the deployed server pages.
- Ability to **Parallelize Delays** of server pages' components that are characterized by heavy processing needs which will **boost performance**.
- Ability to exchange **messages** between components running on different nodes.
- Ability to **cache** more **static content** existing in **dynamic server pages**.
- Make more use of available **hardware** existing in the cluster if available.
- **Fault Tolerance** can be implemented in a way that does not require full re-execution of the requested server side script through **Service State Migration**.
What is the effect, on performance, of changing the communication layer of web environments from the normal stateful TCP to the stateless UDP communication protocol, and how will this change help the emergence of new mechanisms and features that will add to performance and to already established characteristics of web environments such as expandability, extendibility, fault tolerance, high availability, ... etc.?
Deliverables

Performance Deliverables
  Measured experimental results for different architectures comparison. (TCP With dispatchers versus multi-channel over UDP)

Feature Deliverables
  Additional features that are gained due to the architectural change of the network communication layer, and may add to performance in some situations.
Deliverables

Performance Deliverables

• Comparing end-to-end speed (Browser-Back End Web Environment-Browser) of the containers with traditional web environments (Apache and Tomcat), without traffic load. (We are targeting 50% increase in speed)

• Comparing the maximum number of connections within a specific time duration that could be handled by the Containers environment compared to Apache Cluster based on transaction traffic dispatching (Enterprise II or III versus Enterprise I) (We are targeting 20% increase in the number of requests per a specific duration of time)

• Compare both Performance and maximum number of connections that could be handled together. (We are targeting the same values of point 1 & 2)

Very Important: The kind of applications targeted in those tests are those applications that are decomposable by nature and have a considerable amount of processing compared to data transfer; Thus the above results will be relative to the amount of processing needed by the tested services.
Deliverables

Functional Deliverables

- Transparency from the Web Client point of view
- Expandability of the web environment on the hardware, OS, and development environments levels
- Service State Migration between services written with different programming languages (e.g. C++ and Java)
- Easy single point of service deployment
- Hybrid programming language feasibility within a single server page
- Server Page Skeleton caching on the client side

HPA

October, 2008
Approach

The Approach that will be followed to reach the new architecture will be:

- Design
- Prototype for main and basic functionalities
- Preliminary experiments to probe the potential of the idea
- Full system functionality implementation
- Final Comprehensive testing
Work Done So Far

Some work has been done so far on the following different fronts:

- Design
- Prototyping
- Preliminary Experiments
Design

The main and important use cases, class diagrams, sequence diagrams, and deployment diagrams are designed and ready for implementation.

UML2 Specification is used as a Design Modeling Language.

Visual-Paradigm case tool is used as a design case tool.

October, 2008
A prototype for a C++ container is implemented with the following features:
- Service Loader and Factory
- UDP multi-threaded Communication Layer
- Service and communication channels Garbage Collector
- XML Configuration Reader

A Prototype for HPA is implemented with the following features:
- Multi-threaded UDP Communication Layer
- Multi-threaded TCP Communication Layer
- Server Page parser
- XML Configuration Reader
Preliminary Experiments- Single Channel

- Pure Throughput test through single channel file serving.
- Experiment was applied on an HPA/C++ Container in comparison with the Apache/PHP
- Network Speed is 100 Mb/Sec
- The experiment is performed over several runs with different workloads.
Work Done So Far
Preliminary Experiments- Single Channel Results

October, 2008
Multi-Channel Clustered Web Application Servers
Work Done So Far

Experiments- Multi-Channel

- 2 Channel Container Service running on a cluster of 2 nodes and one HPA.
- Apache Running on a cluster of 2 nodes and one dispatcher.
- Network Speed is 100 Mb/Sec
- The experiment is performed over several runs with different workloads and different processing delays.
Work Done So Far
Results – Multi-Channel

% in Bandwidth Gain Per Client

2 Clients

% in Bandwidth Gain Per Client

4 Clients

% in Bandwidth Gain Per Client

6 Clients

% in Bandwidth Gain Per Client

8 Clients

% in Bandwidth Gain Per Client

10 Clients

% in Bandwidth Gain Per Client

12 Clients

% in Bandwidth Gain Per Client

14 Clients

Multi-Channel Clustered Web Application Servers

October, 2008
Related Work

Enterprise Web Application Servers

- Apache Web Server loadable module mechanism, and redirect engine.
- Tomcat Application Server
- ANT for components deployment
- WebSphere Application Server with EJB container.
- GLOBUS Toolkit and Grid Computing (OGSA and OGSI)
- Jboss
- Weblogic
- Jrun
- MS IIS
- Zope
- Oracle Application Server
- Iplanet
Related Work

Dispatching

- IBM Edge Server (Layer-4)
- The Linux Virtual Server Project. (Layer-4)
- IPTABLES Firewall round robin dispatching (Layer-4)
- Squid Reverse Proxy (Layer-7)
- BIND DNS domain name cluster rotation with PERL integration (Layer-7)
- Apache URL redirect feature (Layer-7)
Related Work
Load Balancing and Fault Tolerance

- Cyclone Streaming Server with Socket Cloning
- Helix Streaming Server for UDP based streaming
- WebSphere Application Server clustering features
- The Linux Virtual Server Project
- HA Linux Project for Heartbeats and network take-over mechanisms
- Open Mosix process migration
Related Work

Communication Protocols

• Cyclone Streaming Server with Socket Cloning
• UDT: UDP-based Reliable Data Transfer for High-Speed Wide Area Networks.
• HTTP over UDP Project
• MTU Discovery
Related Work

Web Programming Tools and Interfaces

- CGIs
- SOAP
- XML-RPC
- RMI
- CORBA
- EJBs
- In-Line scripting
- Web2 tools such as AJAX
- CURL For web traffic simulation
Future Work

- Build Containers for more development environments such as **PHP, Python, Perl, ...** etc.
- Integrate the HPA with useful **WEB2** technologies such as **AJAX**
- Integrate the HPA into one of the open source web browsers rendering engines, e.g. **Safari**
- Automatic **Load balancing** mechanisms
- Automatic **Check Pointing** mechanisms to provide more reliable **Fault Tolerance**
- Explore the effect of using **IPV6** on performance
- Implement a UDP-based **Grid Deployment Container** and **Integrate** it to one of the famous **Grid Toolkits** such as **GLOBUS** Toolkit
Questions