Rainbow: Distributed Database System for Classroom Education and Experimental Research

Abdelsalam (Sumi) Helal and Hua Li

Computer & Information Science & Eng. Department
University of Florida
Gainesville, FL 32611-6120, USA
{helal,huli}@cise.ufl.edu

1. Overview

Rainbow is a Java and web-based distributed database system designed for academic purposes, and serves as an exercise to understand concepts of distributed transactions and transaction management including concurrency control, atomic actions, data replication, and fault tolerance.

The motivation for the Rainbow project is twofold. First, Rainbow can be used as a teaching tool. Graduate students taking Distributed Database (or Database Implementation courses) can gain a better understanding and experience of issues involved in transaction processing in a distributed environment [1][2][4]. The second motivation is to use Rainbow as a research tool to conduct scientific experiments on distributed database and transaction processing. To achieve these goals, Rainbow allows the user to configure and program the distributed environment, transactions, and transaction management protocols, and to observe local as well as global executions (history and measured behavior and performance). Rainbow also lends itself as an open system that can be easily changed and extended by students and researchers.

2. Rainbow Architecture and Design

Rainbow consists of three tiers: the graphical user interface (GUI), the Web middle tier, and the Rainbow core (Figure 1).

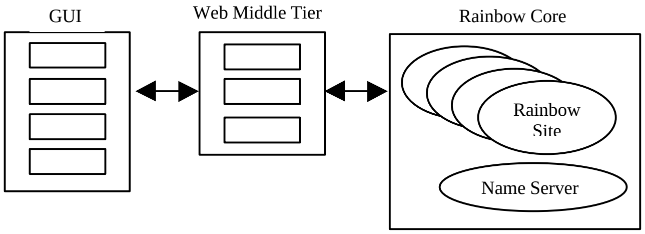


Figure 1. Rainbow architecture with functional mapping

Rainbow GUI allows the user to: a) configure a network simulation; b) configure a name server, Rainbow sites, database items, and select and configure protocols; c) manually or automatically configure and dispatch transactions; d) inject network and site failures and recoveries; and d) monitor the execution state, progress, and measure the performance resulting from executing a Rainbow instance.

The Rainbow core is comprised of the name server and a number of Rainbow sites. The name server stores metadata of all Rainbow sites, such as the id and end point specifications. Also maintained in the name server are the database fragmentation, replication and distribution schema. Any site can query the name server to get pertinent information. Each site can freely communicate with each other. Any site has the capability to concurrently process multiple transactions.

The Web middle tier serves as a fast and reliable bridge between Rainbow GUI and the Rainbow core. This tier consists of Java servlets, which are located at two levels, with level one being at the Rainbow home host and level two at the hosts where the Rainbow core resides. Even though Rainbow GUI applet can only communicate with the Rainbow home host, servlets do not have such restrictions. This means that the servlets residing at the Rainbow home can communicate freely with any other

Proceedings of the 26th International Conference on Very Large Databases, Cairo, Egypt, 2000

host in the network. Thus, the two-level arrangement of servlets provides Rainbow GUI applet the freedom of reaching any host in a Rainbow domain.

The detailed architecture is presented in Figure 2.

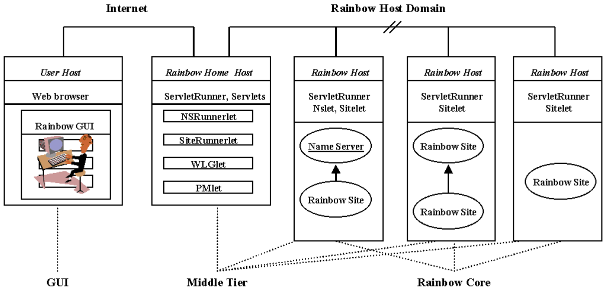


Figure 2. Rainbow architecture with physical mapping.

In this architecture, the user interacts with the Rainbow GUI via a Web browser in the user host. Rainbow GUI is downloaded to the user host as a Java applet when the user clicks a Web universal resource locator (URL) link to the Rainbow home. This URL has the following form:

http://RainbowHomeHost:8080/RainbowDemo.html

where RainbowHomeHost is the IP of the Rainbow home host, 8080 is the listening port of the Web server in this host. The Web page *RainbowDemo.html* contains the Rainbow access authorization and Rainbow GUI applet.

Rainbow GUI applet communicates with Java servlets living in the Rainbow home host. Java servlets are server-side programs (Java threads) living inside a servlet enabling Web server process. The Web server at the Rainbow home host is such a servlet enabling Web server. Rainbow chooses the Java™ Servlet Development Kit Version 2.1 (JSDK) Web server for its lightweight and ease of use. For convenience, we call this Web server the "ServletRunner", which in fact is its former name. Obviously every host in the Rainbow host domain needs a ServletRunner because it requires servlets to function properly. It is for this reason that light-weight of ServletRunner becomes very desirable.

The middle tier consists of a number of servlets, i.e. server side threads living in the ServletRunner. The servlets are: NSRunnerlet, NSlet, SiteRunnerlet, Sitelet, WLGlet, and PMlet. NSRunnerlet and NSlet are responsible for starting the Rainbow name server in the Rainbow host domain and later transferring user request to it. Similarly, SiteRunnerlet and Sitelet are responsible for creating Rainbow sites in the Rainbow host domain and later bringing user requests to them.

WLGlet transfers transaction processing related requests (WLG refers to workload generator) to Rainbow sites in cooperation with Sitelets. PMlet brings progress related requests (PM stands for progress monitor) to and results back from both the name server and the Rainbow sites by working closely with NSlet and Sitelet.

The system requires that the Rainbow home host must have the following servlets: NSRunnerlet, SiteRunnerlet, WLGlet, and PMlet. The reason is that the Rainbow GUI Applet can only communicate with the host it is downloaded from, i.e. the Rainbow home host. The four resident servlets serve as jump-off points for Rainbow GUI to the other hosts in the Rainbow host domain.

NSlet is only needed in the host that the name server lives in. There is only one name server per a Rainbow instance. If there is only one Rainbow instance running, one NSlet is all that is needed. If there are multiple instances running and the name servers are in different hosts, then one NSlet for each distinct host. Similarly, a Sitelet is needed in every host that Rainbow sites live. If multiple sites are created in one host, they share the same Sitelet. It may not be apparent tier from Figure 3 that the middle tier actually involves all Rainbow domain hosts, not just the Rainbow home host.

The Rainbow core consists of the name server and the Rainbow sites, which can live anywhere in the Rainbow host domain including the Rainbow home host. Thus like the middle tier, the Rainbow core may also involve all hosts in the Rainbow domain.

2.1 Transaction Processing Protocols

Rainbow supports 1) replication control protocols (RCP) including Read one write all (ROWA) and Quorum consensus (QC); 2) Concurrency Control Protocols (CCP) including Two-phase locking (2PL) and Timestamp ordering and 3) Two-phase commit (2PC) as the Atomic Commit Protocol (ACP). Rainbow protocols are implemented with minimum interdependencies and assumptions in order to facilitate their replacement (e.g., by students) with minimum system-wide modifications.

The following scenario demonstrates the RCP, ACP, and CCP interactions during the course of transaction processing, in a given Rainbow configuration. When a new transaction arrives at a Rainbow site, the site dedicates one thread to process it. The thread immediately invokes the RCP. The default protocol for RCP in Rainbow is QC. QC starts by building a quorum (read or write) for the first operation of the transaction. To do this, QC needs first to find a set of sites from whom the quorum can be built. QC then sends each site in the set a request for that site's local copies. At that site, copies are read (returning their current value) or pre-written (returning their current version number) through CCP. When a quorum is built for an operation, the next operation is considered. When all operations of a transaction are processed by the RCP, the home site initiates a two-phase commit session, the default ACP in Rainbow. When commitment terminates, the transaction is complete and the thread finishes.

3. Use of Rainbow in Experimental Research

Rainbow is an effective tool to conduct scientific experiments. It has been successfully used in studying the quorum consensus behavior and message traffic in quorum-based systems [3]. It could be used by other researchers when it is made publicly accessible. Rainbow supports experimentation through the following facilities:

- A workload generator component
- A network simulator and fault/recovery injector
- A GUI support in automating experiments and visual rendering of the results.

The performance of transaction processing and several dynamics of the distributed database system can be monitored and measured. Rainbow offers an extensible set of output statistics including:

- Number of committed transactions
- Number of aborted transactions (and rate) due to RCP, ACP, and CCP
- Transaction commit rate
- Transaction abort rates for each type of aborts
- Total number of messages generated per time unit
- Transaction throughput and response time measures
- Other parameters such as number of orphan transactions, round trip messages and other load balance/imbalance indicators.

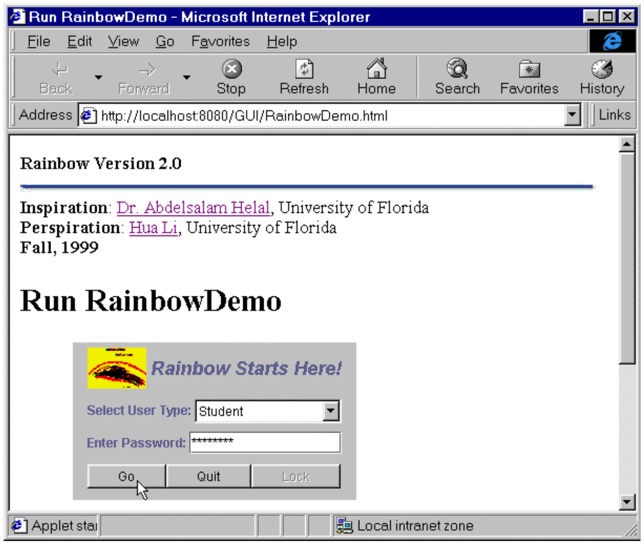
4. A Brief Tour of the Rainbow Demo

4.1 Setting Up

From any Web browser, the user first downloads the Rainbow GUI downloading applet using a well-known URL[1], where *RainbowHomeHost* is the Rainbow home host, which is controlled by the Rainbow system administrator (Figure 3). Upon successful login, the Rainbow GUI is downloaded to the user machine and is activated. If the user is the administrator, he is able to access Administrator menus to configure the Rainbow host domain and the Rainbow name server. Otherwise, the user is ready to start a Rainbow session.

4.2 Running Rainbow

When a new session starts, the user should first configure Rainbow and then submit a workload. Rainbow configuration includes Rainbow sites, transaction processing protocols, database items, and database replication scheme, in that order (Figure 4). If networking simulation is desired, then it should be configured first. The configuration data can be saved for reuse in another session.



http://localhost:8080/GUI/RainbowDemo.html

Rainbow Version 2.0

Inspiration: Dr. Abdelsalam Helal, University of Florida
Perspiration: Hua Li, University of Florida
Fall, 1999

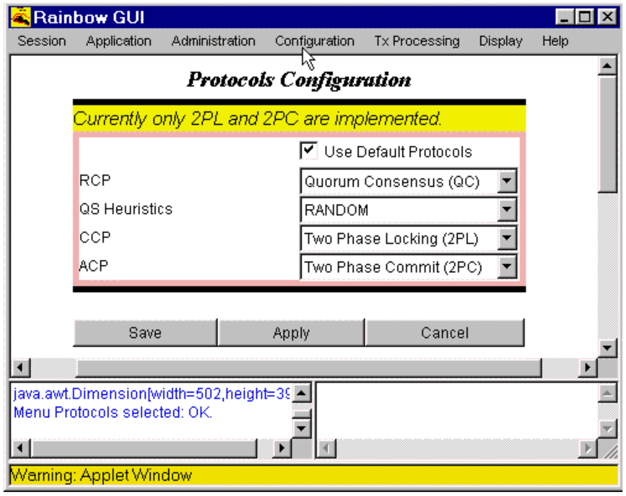
Run RainbowDemo

Rainbow Starts Here!

Select User Type: Student

Enter Password: *********

Figure 3. A screenshot of Rainbow GUI downloading applet.



Protocols Configuration

Currently only 2PL and 2PC are implemented.

☑ Use Default Protocols

RCP: Quorum Consensus (QC)
QS Heuristics: RANDOM
CCP: Two Phase Locking (2PL)
ACP: Two Phase Commit (2PC)

java.awt.Dimension[width=502,height=3
Menu Protocols selected: OK.

Warning: Applet Window

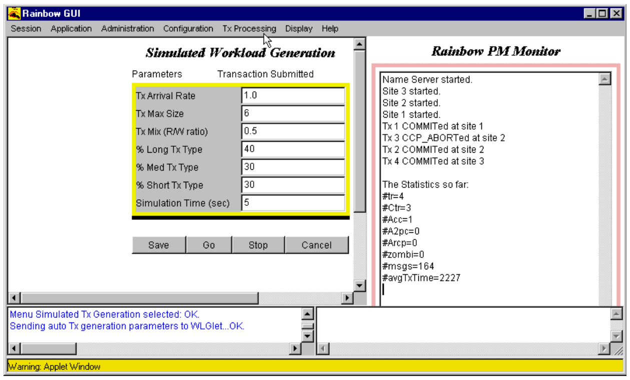
Figure 4. A screenshot of Protocols Configuration window.

Once the configuration is complete, the user can use either the manual or the simulated workload generation panel to compose and submit transactions to Rainbow. The results of transaction processing are feeding back to the user in real time and displayed in the Rainbow GUI. The user can view the final outputs graphically via the Display menu and Tx processing statistics via the Tx Processing menu (Figure 5).

5. The use of Rainbow in the Classroom

The Teaching Assistant (TA) or the Rainbow Administrator is responsible for the following tasks:

[1] "*http://RainbowHomeHost:8080/RainbowDemo.html*


Rainbow GUI
Session Application Administration Configuration Tx Processing Display Help
Simulated Workload Generation
Parameters
Transaction Submitted
Save
Go
Stop
Cancel
Rainbow PM Monitor
Warning: Applet Window

Tx Arrival Rate 1.0
Tx Max Size 6
Tx Mix (R/W ratio) 0.5
% Long Tx Type 40
% Med Tx Type 30
% Short Tx Type 30
Simulation Time (sec) 5

```
Name Server started.
Site 3 started.
Site 2 started.
Site 1 started.
Tx 1 COMMITed at site 1
Tx 3 CCP_ABORTed at site 2
Tx 2 COMMITed at site 2
Tx 4 COMMITed at site 3

The Statistics so far:
#tr=4
#Ctr=3
#Acc=1
#A2pc=0
#Arcp=0
#zombi=0
#msgs=164
#avgTxTime=2227
```

```
Menu Simulated Tx Generation selected: OK.
Sending auto Tx generation parameters to WLGlet...OK.
```

Figure 5. Transaction processing output in a Rainbow session.

- The installation of the Rainbow software on potential Rainbow domain hosts.
- Startup of the dedicated Web server, ServletRunner, on each Rainbow domain host. It is essential that the Rainbow home host must have the ServletRunner running at all times.
- Configuration of the Rainbow Domain. This is done by logining into Rainbow as the administrator and then use the Administration menu.
- Rainbow Name Server Configuration. The administrator can access the submenu "Name Server Configuration", under the Administration menu to configure the parameters and the location of the Rainbow name server.

As soon as the TA has done her work setting up Rainbow, students can download Rainbow and login as Student. The students need to do two basic things. First, they need to configure Rainbow sites, transaction processing protocols (RCP, ACP, and CCP), and the database. Second, they need to generate transactions either manually or using the simulation mode. They can then sit back and examine the results of the execution of the submitted transactions. Homework and lab assignments can be designed around Rainbow. The code can be distributed to students so they can gain hands-on experience on how to implement transaction processing and distributed database concepts. Term projects can be based on modifying Rainbow by adding a protocol (e.g. replacing two phase commit by three-phase commit, basic timestamp ordering by multi-versioning TSO, etc.)

6. References

[1] Bell, D., Grimson, J., *Distribtued Database Systems*, Addison-Wesley, Reading, MA, USA, 1992.

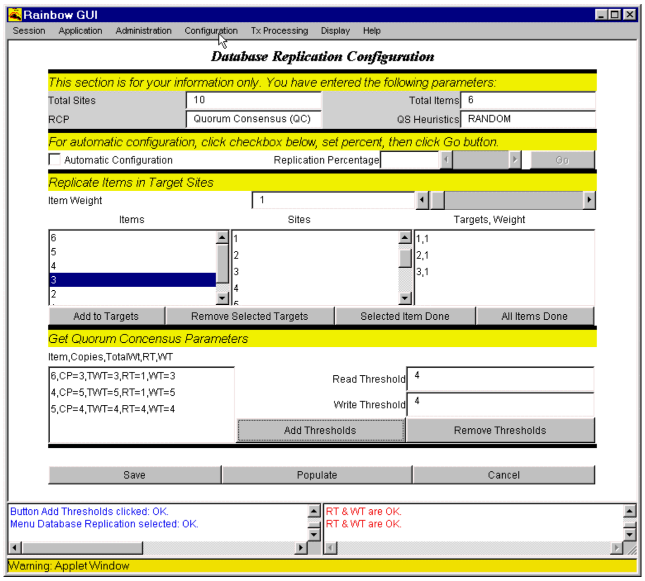
[2] Ceri, S., Pelagatti, T., *Distributed Database: Principle and Systems*, McGraw-Hill Book Company, New York, NY, USA, 1984.

[3] Helal, A., Srinivasan, J., Bhargava, B., *SETH: a quorum-based database system for experimentation with failures*, Proceedings of the 5th IEEE International Conference on Data Engineering, Los Angeles, CA, USA, 1989.

[4] Özsu, M.T., Valduriez, P., *Principles of Distributed Database Systems*, 2nd edition, Prentice Hall, Upper Saddle River, NJ, USA, 1999.

7. Appendix

In the following, we include screen dumps of additional Rainbow GUI windows and dialogue boxes.



This section is for your information only. You have entered the following parameters:

Total Sites 10
Total Items 6
RCP Quorum Consensus (QC)
QS Heuristics RANDOM

For automatic configuration, click checkbox below, set percent, then click Go button.

□ Automatic Configuration

Replicate Items in Target Sites

Item Weight 1

Items	Sites	Targets, Weight
6	1	1,1
5	2	2,1
4	3	3,1
3	4	
2		

Get Quorum Concensus Parameters

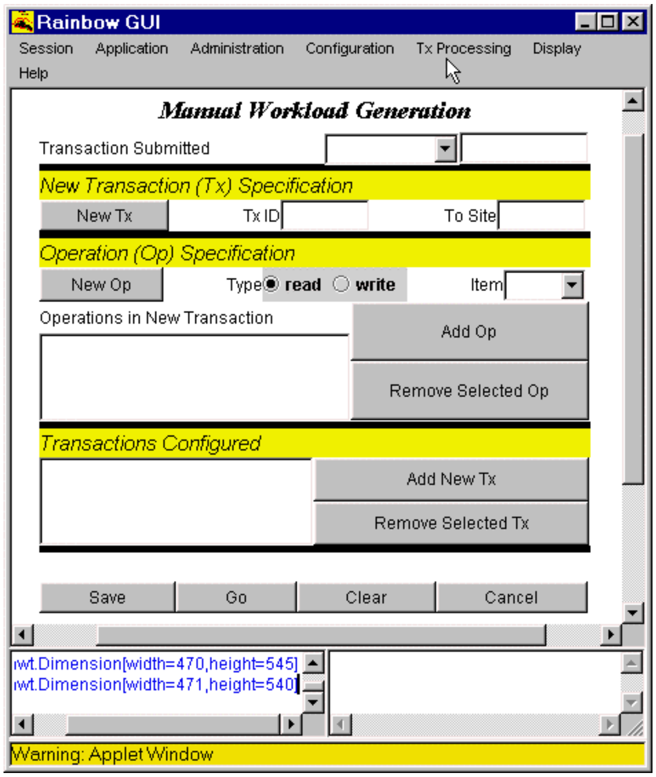
```
6,CP=3,TWT=3,RT=1,WT=3
4,CP=5,TWT=5,RT=1,WT=5
5,CP=4,TWT=4,RT=4,WT=4
```

Read Threshold 4
Write Threshold 4

```
Button Add Thresholds clicked: OK.
Menu Database Replication selected: OK.
```

```
RT & WT are OK.
RT & WT are OK.
```

Figure A-1: Database Replication Configuration panel.



```
iwt.Dimension[width=470,height=545]
iwt.Dimension[width=471,height=540]
```

Figure A-2: Manual Workload Generation panel.