|  |
| --- |
| TRACK Team Project Mid-Term Design Report |
| Distributed Database Query Engine Service (DDQES) |
| Frederic Colin [2012400565] Henry Loharja [2012280102] Ng Yi Ying [2012280098] |

Table of Content

[List of Table 1](#_Toc354591197)

[List of Figure 1](#_Toc354591198)

[Introduction 2](#_Toc354591199)

[Project Scope 2](#_Toc354591200)

[System Environments 3](#_Toc354591201)

[Database System and Engine 4](#_Toc354591202)

[Global Table Structure 5](#_Toc354591203)

[Table Fragmentation 5](#_Toc354591204)

[Database System and Engines Communications 6](#_Toc354591205)

[Web Services 7](#_Toc354591206)

[Project Timelines 8](#_Toc354591207)

[Current Progression 8](#_Toc354591208)

[Summary 9](#_Toc354591209)

[Appendices 10](#_Toc354591210)

[Appendix I – Meeting Minutes 1 10](#_Toc354591211)

[Appendix II – Meeting Minutes 2 11](#_Toc354591212)

[Appendix III – Meeting Minutes 3 12](#_Toc354591213)

[Appendix IV – Meeting Minutes 4 13](#_Toc354591214)

# List of Table

[Table 1 Global Table Structure of Publisher, Book, Customer and Orders 5](#_Toc354433307)

[Table 2 Fragmentation of Publisher Table 5](#_Toc354433308)

[Table 3 Fragmentation of Book Table 5](#_Toc354433309)

[Table 4 Fragmentation of Customer Table 6](#_Toc354433310)

[Table 5 Fragmentation of Orders Table 6](#_Toc354433311)

[Table 6 Table of Project Timelines 8](#_Toc354433312)

# List of Figure

[Figure 1 Distributed Database Query Engine Service (DDQES) Architecture Paradigm 4](#_Toc354433337)

# Introduction

Distributed Database Query Engine Service (DDQES) is a project system which is assigned to promote a better and deep insight on the knowledge of distributed database system through a hands-on software design and implementation experiment. In term of scope, the project system is covering a distributed system environment which involved four distributed sites and four database engines. In short, the entire distributed environment is designed to be capable of handling distributed query processing that involved distributed query requests, query analysis for decomposition and localization. With the concern of data communication cost, query optimization is within the considerations of the project scope for effective database retrievals and optimized returned result set. In addition, to ease crossed platforms data communication; DDQES is expected to be published as a web service to support remote service call from all available sites for further processing, which is described with Web Service Description Language (WSDL). The entire project is handled by a group of three team members (with TRACK as the team name), namely Frederic Colin, Henry Loharja and Ng Yi Ying as the team leader. Throughout the entire midterm report, elaborations on the topic of designed system environments, databases systems and engines communication, master site strategy and dictionary storage and the implementation of web services and identified project timelines are covered; to serve as a design and project proposal for review.

# Project Scope

Distributed Database Query Engine Service (DDQES) is designed to handle multiple remote calls from involved sites in the entire system network architecture; through a published web service. It is within the project scope to include four physical sites which involved four distributed databases which consisted of four tables: Publisher, Book, Customer and Order. The final submitting DDQES is expected to provide:

* Structure Query Language (SQL) Support: Capable to handle various SELECT statements through the submission of query request to the published web service
* High Availability (HA) Web Service: Provide a web-based access point for query request submission from various sites through remote call. Mainly to promote a high availability rate, the HA Web Service is scoped to have the capability of immediate handle by another on-line site (based on the service dictionary) to take over the web service providing responsibility if the current handling service site is down.
* Distributed Query Processing Application (DQPA): Upon the HA Web Service is being called by the site, an instance of DQPA which is online and already linked to the web service owner to analyze the submitted query, generate query retrieval plan with query decomposition and finally data location. Once the data is retrieval, merging the result set for return to caller site is also within the project scope. Specifically mentioned on the result of the query, the output will include the display of the following:
  + The size of query result set
  + The optimized query tree
  + The time cost of query
  + The communication cost of query
* Query Optimization: To reduce the data communication cost, various optimizations are expected to be included in the project design and implementations, including:
  + Query tree optimization and result set merging
  + Query tree reduction by fragmentation
  + Web service availability optimization

In term of project deliverables, it is expected to finally complete a reliable Distributed Database Query Engine Service (DDQES) with the mentioned scope and functionality, together with a final report and system operation manual for final submission. In the following section, the system environment for DDQES is documented which included the details of database system and engines, global tables structure and DDQES development and implementation environment.

# System Environments

Under this section, it aimed to document the descriptions of relevant system environment of Distributed Database Query Engine Service (DDQES). Mainly to ease the integration efforts of having multiple developers (three team members) for the project, Java is the identified programming language for building DDQES. Another reason for the decision for the selected programming language is due to its capability to provide web service with Servlet and Java Server Page which fulfilled the project development and implementation needs. Hence, it is reasonable to visualize that all four sites within the DDQES are running the same Java application instance on their physical location for handling all necessary operations. To simulate the real world environment and to own a closer estimations for the communication cost, four physical personal computers will be used to each represents a site rather than using a single virtual computer for demonstration. Regarding the data communication protocol, socket communication is preferred which relies on the personal computer open ports. The overall Distributed Database Query Engine Service (DDQES) architecture paradigm can be viewed as the following:

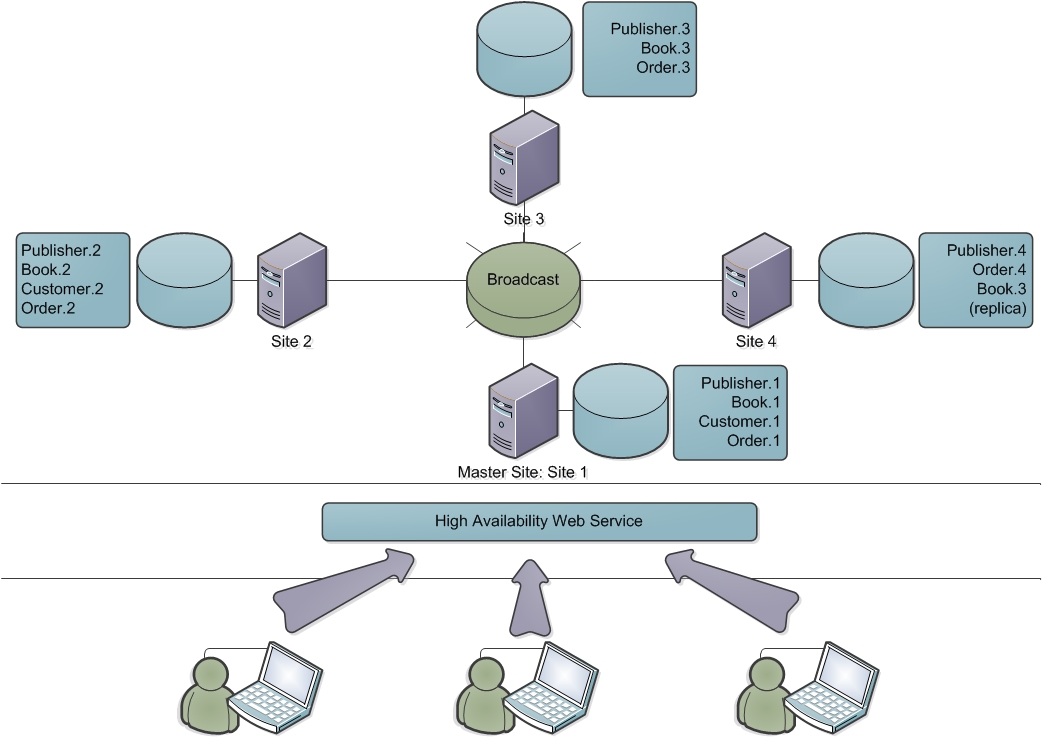


Figure 1 Distributed Database Query Engine Service (DDQES) Architecture Paradigm

Based on the above illustration, it can be further explained as the users who intended to grab information from the databases own no direct knowledge of the physical storage location of the intended data. The users simply submit their query requests to the published web service through an URL. The High Availability (HA) Web Service is responsible of further branching of proceeding operations which will be discussed in the coming section [Section: Web Services].

## Database System and Engine

For times it is relatively hard to standardize all the distributed systems to be equipped the same database systems and/or engines for data storage and handling; the project is designed to handle the communication of multiple heterogeneous database systems, including PostgreSQL, MySQL and SQLite; where each database system can be handled equally by each team member. It is believed that such implementation may higher increase the practicality of the entire project.

## Global Table Structure

In details, each database system in the sites are based on the following table structure as mentioned previously; including Publisher, Book, Customer and Orders table:

|  |
| --- |
| Table Structure |
| Publisher (id integer PRIMARY KEY, name char(100), nation char(3))  Book (id integer PRIMARY KEY, title char(100), authors char(200), publisher\_id int, copies int)  Customer (id integer PRIMARY KEY, name char (25), rank integer)  Orders (customer\_id integer, book\_id integer, quantity integer) |

Table 1 Global Table Structure of Publisher, Book, Customer and Orders

## Table Fragmentation

As can be noticed from the Distributed Database Query Engine Service (DDQES) architecture paradigm [Figure 1], the number of tables and ID are different from one site to another. It is in fact drawn based on the fragmentation rule which aim to optimize data communication cost and performance. The fragmentation details can be summarized as below, grouped by table name:

* Publisher

|  |  |  |
| --- | --- | --- |
| Fragmentation | Fragmentation Condition | Site |
| Publisher.1 | id < 104000 AND nation=’PRC’ | Database 1 [Site 1] |
| Publisher.2 | id < 104000 AND nation=’USA’ | Database 2 [Site 2] |
| Publisher.3 | id >= 104000 AND nation=’PRC’ | Database 3 [Site 3] |
| Publisher.4 | id >= 104000 AND nation=’USA’ | Database 4 [Site 4] |

Table 2 Fragmentation of Publisher Table

* Book

|  |  |  |
| --- | --- | --- |
| Fragmentation | Fragmentation Condition | Site |
| Book.1 | id < 205000 | Database 1 [Site 1] |
| Book.2 | id >= 205000 AND id < 210000 | Database 2 [Site 2] |
| Book.3 | id >= 210000 | Database 3, 4 (R) [Site 3, 4] |

Table 3 Fragmentation of Book Table

* Customer

|  |  |  |
| --- | --- | --- |
| Fragmentation | Fragmentation Condition | Site |
| Customer.1 | (id, name) | Database 1 [Site 1] |
| Customer.2 | (id, rank) | Database 2 [Site 2] |

Table 4 Fragmentation of Customer Table

* Orders

|  |  |  |
| --- | --- | --- |
| Fragmentation | Fragmentation Condition | Site |
| Order.1 | customer\_id < 307000 and book\_id < 215000 | Database 1 [Site 1] |
| Order.2 | customer\_id < 307000 and book\_id >= 215000 | Database 2 [Site 2] |
| Order.3 | customer\_id >= 307000 and book\_id < 215000 | Database 3 [Site 3] |
| Order.4 | customer\_id >= 307000 and book\_id >= 215000 | Database 4 [Site 4] |

Table 5 Fragmentation of Orders Table

On top of the basic database configurations to support the implementation of Distributed Database Query Engine Service (DDQES), Hibernate Framework will be used which is an object relational mapping library for mapping an object-oriented domain model to a traditional relational database. In the following section, detailed discussion on the framework will be described.

# Database System and Engines Communications

[Fred’s Part]

Under this section, write on:

* Brief description regarding Hibernate Framework
* The database communication strategy as what we have discussed last Friday. For instance, there is Distributed Query Processing Application (DQPA) sitting on the master site for query decomposition (you can refer to what I have written in page 3) and how it communicate with the local Hibernate. Followed by the query is further distributed to corresponding and/or involving sites for data retrieval and the other Hibernate in the sites coordinate and grab the data and send it back to the caller. Should provide a clear process flow, such as DQPA -> Send Query out to other sites -> Hibernate in other site grab the data from database -> send back to the site…
* I think we did mention on timestamp each of the query to avoid query request collision or something like that?
* Backup and recovery? I think we just write the basic ones first so I think we can limit the scope to have a backup database which will copy all data from all the sites to one database. So, in case one site is down, the application will grab the data from the backup database and provide alert message of that the site(s) is down currently.

# Web Services

[Henry’s Part]

Under this section, write on:

* Master Site Strategy and Dictionary Storage: Elaboration on the purpose of master site (each of the sites may potentially be the master site). And, the discussed method of always ping the current web service providing site to ensure its availability. I think my explanation concept here is that it’s like a domain name and hosting service. We have a domain of [www.abc.com](http://www.abc.com) for example which is an access point for the user to submit query. If the current master site is down (failed to ping it), then immediately another site will take over to host [www.abc.com](http://www.abc.com). I am not quite sure whether is it going to work. If it is not possible, I think we can write that the “ping” operation is done by the web service/ or an application interface. For instance, when the user submit the SQL statement to an URL (where the web service is located), there will be an application sitting there to check if the current master site is online or not. If not, point to next one based on the dictionary storage. You can write the best solution based on your preference but just ensure your writing matched my introduction and system environment. Let me know if I need to change anything else.

Not to forget mention where do we plan to store the dictionary storage, I think we previously agreed to store the dictionary on each site. But if you choose to write on the 2nd method, then you can identify another way/method.

* The web service interface like how does it look like, just a draft will do. If you don’t have the time to draw, at least mentioned that we are using JSP/Servlet which is in HTML format. Also mentioned on the display element, such as the interface will display the initial query tree and optimized tree to show the optimization result [refer to the project description from TA] and the return result set will contain query result set and etc…

# Project Timelines

Since the project was initiated, the team was having a weekly meeting to plan and follow up the project progression [Appendix I - III]. Good project management starts with proper scheduling of a project which may increase the project success rate. As indicated the deadline submission of the project is mid of June, below is the created milestones for the team proper progression and monitoring after the last group discussion. While the person responsible simply means the person who owns a higher knowledge towards one particular subject, the entire team will be paying full attention and coordination to contribute to the project success:

|  |  |  |
| --- | --- | --- |
| Milestone Title | Date | Person Responsible |
| Mid-Point Project Review  Finalize Launcher and Hibernate Connectivity | 29th April, 2013 | Ying  Fred |
| Perform Query and Data Communication Cost  Finalize Optimization Strategy Plan | 6th May, 2013 | Fred, Henry, Ying  Ying |
| Database Building and Hibernate Linkage  Final Documentation Template Creation | 13th May, 2013 | Fred  Ying |
| Individual Application Development (Java) | 20th May, 2013 | Henry |
| Application and Connection Integration | 27th May, 2013 | Fred, Henry, Ying |
| Application Wrap and Documentation | 3rd June, 2013 | Fred, Henry, Ying |
| Final Review and Presentation Wrap Up | 10th June, 2013 | Henry |

Table 6 Table of Project Timelines

## Current Progression

Apart from the project milestones, this section documents the current progression of TRACK team for the project. Up to current moment, after several group discussions which were held previously the system environment and scenario is created as documented above. Apart from those documented details, a launcher prototype program was created, using Java programming language. The main intention of such an experiment was to identify the possibilities of automating the creation of database table structure based on an XML file. Without exploring further, the launcher prototype is capable in creating physical database directly into MySQL database based on a given XML file and some made assumptions. With the belief that the entire prototyping application is a good basic start for the project, TRACK team is looking forward to work well together to complete the entire project specifications.

# Summary

Though TRACK team had decided to adopt the default scenario for implementing Distributed Database System Query Engine Service (DDQES), the following are the strengths and/or challenging points which worth a note:

* Four physical computers: The team is increasing the project difficulty level by pushing the DDQES environment to be run from four different physical computers rather than running on a single computer with four virtual sites.
* Heterogeneous database engines: Instead of putting all four databases into a single computer which most likely to be the same database system, the team is challenging the scope by using heterogeneous database system; including MySQL, PostgreSQL and SQLite by using Hibernate framework library.
* High Availability (HA) Web Service: To reduce the limitation of the nature of distributed database system, the publishing web service is aimed to provide a high availability rates which is capable of service recovery to reduce the possibility of having an offline service. [If Fred included the points by saying there is a replicated backup database, it should be added here.]

# Appendices

## Appendix I – Meeting Minutes 1

Date : 11th March, 2013

Time : 10:00 am – 12:30 pm

Venue : 8-304

Attendees : Frederic Colin, Henry Loharja, Ng Yi Ying

**Intended Discussions**:

* Identify team leader
* Identify as to create new scenario or use the default one
* Identify development environment
  + Number of sites
    - Using same application or different sites uses different application environment, such as site A for finance and site B for warehousing
  + Programming language
  + Data communication and transmission medium
  + Database management system
* Gather all team members’ student ID
* Desirable project task distributions and responsibilities
* Identify schedule for proper progression tracking of the project
* Next step to proceed after discussion

**Meeting Minutes**:

* The student ID of the team members were noted to be filled up to the discussion board in the website and Yi Ying was selected as the team leader.
* The team decided to use back the default scenario and add features to the existing one after the basic requirements are met.
* The following are the discussion results regarding the development environment
  + Number of sites: The team decided to stick back to the default 4 sites and expand it if there’s a time.
  + Programming language: The team decided to start with the same programming language first and JAVA is preferred.
  + Data communication and transmission medium: It is preferable to use socket and TCP/IP or UDP which able to expose the data through web services.
  + Database Management System: The team had shortlisted Hybernate and MySQL.
* The team decided to work together for all tasks and will distribute other unknown tasks in future for better project management.
* Every Monday will be the weekly meeting for the team and during the meeting, the team members will review the progression since last week, discuss on the tasks for this week and how additional working hours are required for this week.
* Fred suggested storing the scenario condition into an XML, further creating an application to generate and launch the sites by referencing the XML “dictionary”.

**To-do List**:

* Submit group to discussion board in <http://learn.tsinghua.edu.cn>
* The student ID for team members are as follow:
  + Frederic Colin – 2012400565
  + Henry Loharja - 2012280102
  + Ng Yi Ying – 2012280098 [Team Leader]
* Register for GitHub and download and install Git
* Create a scenario for the assignment by specifying the limitation and general picture of the entire project
* Create a prototype launcher based on the XML
* (Optional) Draw the relationship among sites and possible working/communicating logic

## Appendix II – Meeting Minutes 2

Date : 18th March, 2013

Time : 10:00 am – 12:30 pm

Venue : 8-304

Attendees : Frederic Colin, Henry Loharja, Ng Yi Ying

**Intended Discussions**:

* Brief discussion on last week lecturing materials
* Review on the draft of the created first scenario
  + Architecture and drew diagram
  + XML for environment variables
  + Scalability
  + Potential modifications
* Review on the created launcher prototype
* Shortlisting database systems to further prototyping
* Distribution of tasks
* Scheduling and milestones for better project monitoring
* Next step to proceed after discussion

**Meeting Minutes**:

* A brief review of last week lecturing was carried out by the team.
* The initial draft of the case scenario was finalized to be just fine for the starting and will definitely enhance the model when the project goes on. The potential solution is to make each and every site a master site of its own so that when one of the sites was down, the other may still progress while the other will perform the recovery.
* Fred and Henry were agreed to check out on Hybernate and its connectivity to first launcher prototyping while Yi Ying will be working on MySQL to further explore the database connectivity.
* After checking the schedule for mid-term presentation, it was agreed by the team that there will be sufficient time to prepare and work on this project to achieve desire goals. For now, weekly meeting and report are agreed to be kept going as more prototyping can increase the completeness of the project.

**To-do List**:

* Fred and Henry: Try out launcher prototype and test out the connectivity between the application and Hybernate.
* Yi Ying: Try out the connectivity between the application and MySQL and try to expand the launcher prototyping.

## Appendix III – Meeting Minutes 3

Date : 25th March, 2013

Time : 10:00 am – 12:30 pm

Venue : 8-304

Attendees : Frederic Colin, Henry Loharja, Ng Yi Ying

**Intended Discussions**:

* Brief discussion on last week lecturing materials
* Review on the draft and progression of the created second scenario
  + Connection from Launcher to MySQL
  + Connection from Launcher to Hibernate
* Discussion on the concept of web services (Since one of the requirements expects the distributed database query engine is capsuled into a web service) and WSDL
* Review on the required content of mid-term design report and presentation
  + Design of the distributed database query engine
  + Timeline for project work plan
* Next step to proceed after discussion

**Special Notes:**

* Mid-term presentation will be on 26 April, 2013

**Meeting Minutes**:

* A short discussion on the homework assignment was carried by linking to the project needs in the future
* The use and publish of service to the web may need further verification with the teaching assistant before proceeding
* It is agreed to ready the launcher before next coming meeting for evaluation and preparation for later mid-point presentation

**To-do List**:

* Meet up the teaching assistant for concept verification
* Build up the launcher connectivity to MySQL, Hibernate and Postgre

## Appendix IV – Meeting Minutes 4

Date : 19th April, 2013

Time : 15:00 pm – 18:00 pm

Venue : 8-304

Attendees : Frederic Colin, Henry Loharja, Ng Yi Ying

**Intended Discussions**:

* Review on the draft and progression of the created second scenario
  + Connection from Launcher to MySQL
  + Connection from Launcher to Hibernate
* Review on the required content of mid-term design report and presentation
  + Design of the distributed database query engine
  + Timeline for project work plan
* Preparation of midterm presentation and report writing section distribution
* Next step to proceed after discussion

**Meeting Minutes**:

* A short discussion on last class of distributed database system.
* The current progression of the Launcher was reviewed by specifying existing limitation of not yet including Hibernate as part of the prototype.
* However, the connection from Launcher to database system is relatively stable for now and should plot over to include Hibernate right after midterm presentation.
* The discussion of the entire scenario was carried out to double confirm the discussed details in the past month.
* The content of the midterm report was discussed and section writing was distributed and agreed to complete before Thursday for integration.
* The demonstration of Launcher was agreed not to push up to the midterm presentation since it is not quite complete yet.
* Presentation slide will be prepared by Yi Ying and content will be added by Fred and Henry.
* Another meeting was expected to be carried out before the midterm presentation for rehearsal of having better presentation performance.

**To-do List**:

* Write on midterm report
* Create midterm presentation slides and content