

**A MULTI-LEVEL BIOMEDICAL ONTOLOGY-ENABLED BROKER:  
DYNAMIC SERVICE-BASED DATA SOURCE INTEGRATION**

by  
SHENG-CHIEH JACK FU

Presented to the Faculty of the Graduate School of  
The University of Texas at Arlington in Partial Fulfillment  
of the Requirements  
for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON

May 2008

Copyright © by SHENG-CHIEH JACK FU 2008

All Rights Reserved

To my wife Jeanne, my parents Tsun-nan Fu, Jung Kuo,  
and parents in law Robert and Judy  
for their love and fully support.

## ACKNOWLEDGEMENTS

I am forever grateful to my advisor Dr. Ramez Elmasri for his expertise, constant guidance, encouragement and support towards the successful completion of my doctoral degree. I also want to thank my committee members, Dr. Nikola Stojanovic, Dr. Jean Gao, Dr. Lynn Peterson, and Dr. Chien-Pai Han for their time to participate in my committee and providing additional insightful comments into my research.

I thank my fellow labmates at BioCom Lab, Feng Ji, Weimin He, Qing Li, Yiming Zhang, and Kyungseo Park for creating an interesting and enjoyable environment to work where I have spent endless hours of my time, and for the research and publications we have worked together. The discussions and team work have played important parts in my studies.

I would like to thank my family with deepest gratitude, especially my parents for being a role model and brought me up to the person who I am today. Their unconditional love and support has helped me through my studies. I also want to thank my sisters for their support and encouragement, especially my sister, Dr. Sheng-Chen Denise Fu, who has inspired me in the Mathematics area. Lastly, and most importantly, I want to thank my wife Jeanne and son Winston for their sacrifice and patience. I couldn't have done it without their support and understanding.

May 2, 2008

## ABSTRACT

### A MULTI-LEVEL BIOMEDICAL ONTOLOGY-ENABLED BROKER: DYNAMIC SERVICE-BASED DATA SOURCE INTEGRATION

SHENG-CHIEH JACK FU, Ph.D.

The University of Texas at Arlington, 2008

Supervising Professor: Ramez Elmasri

Web services have recently become a new trend for gathering biomedical information. However, it is not easy to integrate and obtain a concise/complete query result among thousands of service operations. In this dissertation, we propose a multi-level ontology-enabled service broker system for dynamically integrating web services in the biomedical domain (BioServiceBroker)[1, 2]. By introducing multi-level modeling concepts and intra/inter level relationships [3], our approach facilitates more accurate modeling of biomedical ontologies, which leads to a better understanding of the data stored in various biological data sources as well as the services provided by the data sources. In addition, we incorporate temporal concepts with new enhanced QoS measures [2], which allow service requesters to control more querying factors in order to precisely invoke corresponding service operations. We also define a Unified Biomedical Service Interface (UBSI) as a proposed service deployment standard.

The multi-level ontology-enabled service broker system can be combined with other mediator systems [4] for cross referencing the diverse bioinformatics sources and can also be utilized in other application domains. Our ultimate goal is to construct a public, scal-

able, and interoperable biomedical service platform based on UBSI to benefit scientists in data searching and publishing.

## TABLE OF CONTENTS

ACKNOWLEDGEMENTS . . . . .	iv
ABSTRACT . . . . .	v
LIST OF FIGURES . . . . .	x
LIST OF TABLES . . . . .	xii
Chapter	
1. INTRODUCTION . . . . .	1
1.1 Motivation . . . . .	1
1.2 Contributions . . . . .	3
1.2.1 Multi-level Conceptual Data Modeling . . . . .	3
1.2.2 Web Services Integration . . . . .	4
1.3 Organization of the Dissertation . . . . .	6
2. BACKGROUND . . . . .	7
2.1 Overview of Biological Data Sources . . . . .	7
2.1.1 Data example – PDB (Protein Data Bank) . . . . .	8
2.1.2 Characteristics of Biological Data Sources . . . . .	13
2.1.3 Summary . . . . .	15
2.2 Biological Data Source Integration . . . . .	15
2.2.1 Data Warehouse Approach . . . . .	16
2.2.2 Mediator-based Approach . . . . .	17
2.2.3 Ontology-based Approach . . . . .	18
2.2.4 Service-based Approach . . . . .	20
2.2.5 Comparison . . . . .	21

3. MULTI-LEVEL CONCEPTUAL DATA MODELING . . . . .	24
3.1 What is Conceptual Data Modeling? . . . . .	25
3.2 Multi-level Concept . . . . .	25
3.3 Multi-level Relationships . . . . .	28
3.3.1 Formal Definitions . . . . .	28
3.3.2 Properties Summary . . . . .	32
3.4 Multi-level Integration . . . . .	33
4. BIOINFORMATICS WEB SERVICES . . . . .	36
4.1 Introduction to Web Services . . . . .	36
4.2 Overview of Bioinformatics Web Services . . . . .	39
4.3 Classification of Bioinformatics Web Services . . . . .	41
4.4 QoS for Web Services . . . . .	42
5. MULTI-LEVEL BIOMEDICAL SERVICE BROKER . . . . .	45
5.1 Web Service Ontology . . . . .	46
5.1.1 Service Schema . . . . .	48
5.1.2 Relationships of Service Operations . . . . .	50
5.1.3 Properties of Service Relationships . . . . .	53
5.2 Reference Database . . . . .	54
5.3 Web Service Query Planner . . . . .	54
5.3.1 Service Pre-Selector . . . . .	54
5.3.2 QoS Service Analyzer . . . . .	55
5.3.3 Formal Definitions of QoS Parameters . . . . .	61
5.4 Dynamic Invocation Engine . . . . .	65
5.5 Web Service Result Fusioner . . . . .	65
5.5.1 Single Result Processing . . . . .	67
5.5.2 Batch Results Processing . . . . .	68



5.6	UBSI (Unified Biomedical Service Interface) . . . . .	68
5.7	Experimental Results . . . . .	70
5.7.1	Service Analysis . . . . .	70
5.7.2	Events Handling . . . . .	71
6.	APPLICATIONS . . . . .	80
6.1	Mediated Taxonomy System . . . . .	80
6.2	Biomedical Service Workflows . . . . .	82
7.	CONCLUSIONS AND FUTURE DIRECTIONS . . . . .	87
7.1	Conclusions . . . . .	87
7.2	Future Research Directions . . . . .	89
	REFERENCES . . . . .	90
	BIOGRAPHICAL STATEMENT . . . . .	99

## LIST OF FIGURES

Figure	Page
1.1 Three Integration Layers . . . . .	2
2.1 PDB REMARK in 1992(A) and 1996(B). Figure is from [5] . . . . .	9
2.2 The STAR/CIF data representation. Figure is from [6] . . . . .	11
2.3 Sample PDB data in mmCIF format. Figure is from [7] . . . . .	12
2.4 Data Warehouse Integration. Figure is from [8] . . . . .	16
2.5 Mediator-based Integration. Figure is from [8] . . . . .	18
2.6 Static Service-based Integration . . . . .	20
2.7 Dynamic Service-based Integration . . . . .	21
3.1 Multi-level Biological System . . . . .	27
3.2 Horizontal and Vertical Integration . . . . .	34
3.3 Hybrid Integration . . . . .	35
4.1 Web Service Frameworks . . . . .	37
5.1 BioServiceBroker System Architecture . . . . .	46
5.2 BioServiceBroker System . . . . .	47
5.3 BioServiceBroker – Web Service Schema Diagram . . . . .	49
5.4 BioServiceBroker – Service Relationships . . . . .	51
5.5 BioServiceBroker – QoS Service Analyzer (Line Chart) . . . . .	56
5.6 BioServiceBroker – QoS Service Analyzer (High-Low Chart) . . . . .	57
5.7 Response Time in Case 1 . . . . .	58
5.8 Response Time in Case 2 . . . . .	60
5.9 Response Time in Case 3 . . . . .	61

5.10	EMBL dbfetch service operation Result (partial) . . . . .	66
5.11	PharmGKB Original Return Map (partial) . . . . .	74
5.12	UBSI getRelatedPathwayByGene Operation Return Query Report . . . . .	75
5.13	UBSI Batch Operations Query Form . . . . .	76
5.14	UBSI Batch Operations Return Query Report . . . . .	77
5.15	Service Operations 36 and 128 – Grouping by HOUR_OF_DAY . . . . .	78
5.16	Service Operations 36 and 128 – Grouping by DAY_OF_WEEK . . . . .	79
6.1	Mediated Taxonomy System . . . . .	81
6.2	Web Service Type Workflow . . . . .	82
6.3	Web Service Instance Workflow . . . . .	85
6.4	Web Service Instance Graph . . . . .	86

## LIST OF TABLES

Table	Page
2.1 Integration Approaches Comparison. . . . .	22
2.2 Integration System Comparison. Table is from [9]. . . . .	23
3.1 Properties of Relationships in Multi-level Data Model. . . . .	33
5.1 Properties of Service Relationships. . . . .	54
5.2 Average Response Time ( $\overline{RT}_{tf_i}$ ) in Case 1 . . . . .	59
5.3 Stability ( $STA_{tf_i}$ ) in Case 1 . . . . .	59
5.4 Average Response Time ( $\overline{RT}_{tf_i}$ ) in Case 2 . . . . .	59
5.5 Stability ( $STA_{tf_i}$ ) in Case 2 . . . . .	60
5.6 Decision Table. . . . .	65
5.7 Unified Biomedical Service Operation (partial). . . . .	69

## CHAPTER 1

### INTRODUCTION

This dissertation presents a web service broker system to dynamically extract biomedical information from multiple relevant data sources. Biological related software developers can utilize our broker system to construct a more flexible, scalable and reliable application for researchers who intensively use these applications to answer different research questions or to solve various tasks [10]. From the services providers' point of view, access to their services will become more stable and reliable as well since our broker system practically provides load balancing among registered services.

In this chapter, we give the main motivations behind the design and developments of BioServiceBroker system in section 1.1 and summarize the contributions of the dissertation in section 1.2.

#### 1.1 Motivation

Biological and medical research creates large amounts of data spread over diverse databases such as GenBank, PDB (Protein Data Bank) [11, 12], etc., which need to be processed, integrated and organized in order to query them efficiently. Traditionally, computer scientists pursue a schema layer integration on relation/attribute names among various database schemas, while life scientists focus on the instance layer by seeking a universal agreement on identification of biogenetic entities such as International Protein Index (IPI) [13] and Life Science Identifier (LSID) [14] standards in order to achieve the goal of integration. Some other integration work also has been done as mentioned in [9, 15, 16, 17, 18], and most of this work focuses on integration based on the instance and

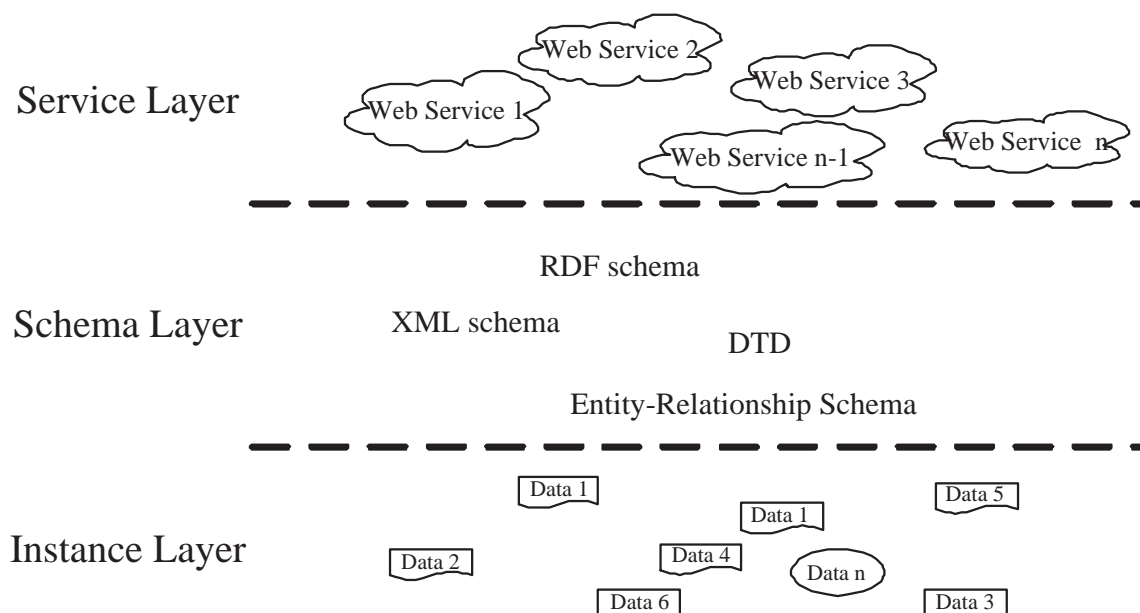


Figure 1.1. Three Integration Layers.

schema layers (see Figure 1.1). However, there are some inevitable drawbacks in instance and schema layer integrations.

For instance layer integration, the problems include determining if different instances coming from different data sources represent the same real-world entity and how to select a source when contradictory information is found in different data sources. Since most of the biological databases have different schema designs and their own identifiers for the same biological entities [19], the identifier of each biological entity needs to be updated and maintained frequently in order to ensure data consistency and integrity. For schema layer integration, two data sources may use different names to represent the same concept (for example, "car" and "vehicle"), or the same name to represent different concepts, or different ways for conveying the same information. In addition, each data source may employ different modeling techniques to create their schema such as Entity Relationship (ER) [20], XML [21], RDFS [22], etc. Schema layer integration requires the detailed schema information of each data source and relies on schema matching and

reasoning techniques [23]. These increase the difficulty of integrating work at the schema layer.

Due to the nature of life science data (highly distributed, dynamic, complex, incomplete, heterogeneous)[24], we focus on service layer integration instead of directly dealing with schemas and data. In this dissertation, we propose a multi-level biomedical ontology-enabled service broker, which incorporates multi-level data modeling concepts [3] with web service interoperation. Some similar and related work has been proposed in other genome research such as SPDBSW [25], FUSION [26], MOBY-S [27] and DAS [28]. Our approach is more flexible, versatile, and platform independent as it can provide loosely coupled integration across different application domains through some related industrial and academic standards such as XML, SOAP [29], WSDL [30], and UDDI [31].

## **1.2 Contributions**

The contributions of our work can be divided into two major parts: In section 1.2.1, we overview our contributions to multi-level conceptual data modeling, which can be used for describing properties and behaviors of the data between different abstraction levels. In section 1.2.2, we overview our contributions to web services integration, which provides a QoS ensured web service platform to reduce overall network traffic and achieve high availability when many users attempt to access bioinformatics and biomedical data sources via their provided services.

### **1.2.1 Multi-level Conceptual Data Modeling**

In the biomedical/biological research fields, some semantic data (experiments, evidence, annotation, etc.) can be classified into different abstraction levels (from the DNA/RNA level to higher levels such as cells, tissues, organs, and biological systems)

based on different degrees of abstraction. Hence, it is important to represent and integrate the data across these different levels. We address the multi-level concept, one of the essential characteristics of biological data, and develop a multi-level conceptual data model. Our approach facilitates more accurate modeling of biomedical ontologies, and a better understanding of the data stored in various biological data sources. We propose intra-level and inter-level relationships and give formal definitions illustrated with examples to precisely describe relationship/behavior among concepts at multiple levels. In chapter 3, we will give a more comprehensive description, and show the need for this modeling technique in data source integration.

### 1.2.2 Web Services Integration

Web services technology not only increases the flexibility and interoperability of business software development, but also provides a new way for scientists to retrieve data and utilize tools more efficiently. Currently, many bioinformatics data and tools can be accessed/invoked through varied service operations provided by different web services. Our BioServiceBroker system is an integrating platform that can dynamically coordinate these web services. Some key advantages are listed and discussed below. The detailed system architecture description will be given in chapter 5.

**Flexibility** – Unlike static web service providers, where the number of services provided to the end users is limited and the services are specified at design time. Our BioServiceBroker can dynamically invoke corresponding service operations that confronts users' QoS (Quality of Service) query requirements. By utilizing our system, the application is no longer restricted to the original set of service operations that were specified and hard-coded at the design or compile time. In contrast, the capabilities of the application can be extended at runtime.



**Availability** – Availability represents the probability that a service is available or ready for immediate use. Our system can dynamically determine the best service operations chosen from candidates that can provide equivalent/homogeneous query result based on their history and current time frame. However, it is still possible that the selected "best" service accidentally fails to gather data. If this is the case, the system will switch to invoke the next candidate with the purpose of fulfilling user/client program's request. That is why we can improve the availability of services in our broker system.

**Load balancing** – Our BioServiceBroker system utilizes enhanced QoS measures as an essential factor to determine the best service provider among service operation candidates selected from the service ontology. If a registered service provider receives high volume of service requests within a certain time frame, then our system reflects its performance in terms of QoS measures and switches new incoming service requests to others. Each incoming service request (work load) is dynamically distributed in an efficient manner, and not only can the average response time be minimized but also the overall throughput can be increased.

**Reduce network traffic/Save bandwidth** – Most service providers do not classify their service operations based on the granularity of returned data. For instance, a user/client program is only interested in a certain gene sequence, but an available related service operation is "getGeneInfo" that returns not only gene sequence but also a bunch of terms or sections, which are not really needed by the current request. This extra information occupies bandwidth, and increases network traffic. By utilizing our system, the web service result fusioner module is responsible to filter out the unnecessary data section, and construct a more exchangeable format before sending the result back to the requester. Our approach can also reduce or eliminate the post-processing time at user/client program sides.

**Building block of web service workflows** – With the increasing number of bioinformatics data sources and computational analysis programs being made available as web services, a service workflow system is an important tool for scientists to fully utilize these web services. Our service broker system can be used for the creation and performing of scientific web service workflows since we maintain the semantic and syntactic information for each service operation and define service relationships that can express the possible relationships between service operations. This work can be used as a building block for establishing complex web service workflows.

**Unified service interface** – UBSI (Unified Biomedical Service Interface) is designed for achieving interoperability, reusability, and scalability across a varied base of underlying and changing services. A client side (programmer, application, etc.) can remotely invoke a service operation through UBSI, and is not required to deal with complex/inconsistent interfaces of each web service operation.

### 1.3 Organization of the Dissertation

The organization of the dissertation is as follows. Chapter 2 describes the characteristics of biological data sources, and discusses four main integration approaches. In chapter 3, we present multi-level conceptual data modeling, and show the need for this concept. Chapter 4 introduces web services technology and its applications in the Bioinformatics domain. QoS issues for web services are also addressed in this chapter. Chapter 5 proposes a multi-level biomedical service broker system (BioServiceBroker) architecture and gives detailed description for each component. Some examples are also included for demonstrating the system functions. Chapters 6 describes two applications that utilizes our proposed service broker to query multiple biomedical web services. We conclude our work and discuss possible future directions in chapter 7.

## CHAPTER 2

### BACKGROUND

In last decade, we have witnessed the sequencing of the human genome, and the vast growth in the quantity of information stored in biological and bioinformatics data sources. Many approaches have been tried to analyze, classify and integrate these data over the years so that scientists and researchers can locate the information that is relevant to their research. The current issue of *Nucleic Acids Research* [32] also points out that the total count of molecular biology databases is more than 1000 in 2008. As most data sources are developed and maintained independently, they are highly heterogeneous in the types of the stored data and the data format. Hence, the user must decide which data sources to access and in which order. In addition, they need to know how to retrieve information from each data source, and how to combine the query results. The integration of data in these molecular biology databases is becoming more and more critical as scientists cannot always find all relevant data using a single source of data.

This chapter will describe the characteristics of biological data sources by examining PDB (Protein Data Bank) [11, 12] data formats in section 2.1 and discussing four major integration approaches that are presently employed in several integration systems in section 2.2.

#### 2.1 Overview of Biological Data Sources

Unlike data types and properties in other application domains, biological data is more complex, heterogeneous, distributed and incomplete, and thus requires extra techniques to process it. In section 2.1.1, we explore three different data formats (PDB file,

mmCIF, and XML) employed in PDB as an example, and review the characteristics of biological data/data sources that have been pointed out [7, 33, 34, 35] in section 2.1.2. A summary of the key characteristics will be given in section 2.1.3.

### **2.1.1 Data example – PDB (Protein Data Bank)**

When the protein data bank was established in 1971, it held only seven structures and very few were added there till 1980, with a limited number of user groups who were experts involved in structural research. But things changed in the 1980's and 90's as the number of deposited structures grew dramatically and a diverse group of researchers and students in biology, chemistry and computer science started using this data bank [11, 12].

Initially, the structural data was very limited, so it was stored as single text records. But since then, the level of detail in specifying refinement and data collection information has grown at a very fast rate. Apart from this, authors of these structures and new researchers have been adding or changing existing records or making new related entities based on new experimental results. Because of all this, the PDB format has been evolving continuously resulting in records existing in several formats in PDB. An example of this evolution can be illustrated from the example in Figure 2.1.

The data format with which PDB came up right from its establishment in 1970's, which provided a standard representation for macromolecular structure data derived from X-ray diffraction and NMR studies was known as PDB format. Since then it has been continuously evolving and has served the community of researchers well [6]. Many software tools have been built on this format which helped users for depositing and retrieving data from this huge data bank. Many validation programs have also been written to help author's double check their new entries before finalizing and storing them in the database. With the growing demand and popularity of PDB, the number and complexity of the queries on it keeps increasing. It was not hard to realize that with

**A**

```

REMARK      3
REMARK      3 REFINEMENT. MOLECULAR DYNAMICS REFINEMENT BY THE METHOD OF
REMARK      3 A. BRUNGER, J. KURIYAN, AND M. KARPLUS (PROGRAM *XPLOR*) .
REMARK      3 THE R VALUE IS 0.186 FOR ALL 42851 REFLECTIONS IN THE
REMARK      3 RESOLUTION RANGE 10 TO 1.8 ANGSTROMS.

```

**B**

```

REMARK      3 DATA USED IN REFINEMENT .
REMARK      3 RESOLUTION RANGE HIGH ( ANGSTROMS ) : 1.8
REMARK      3 RESOLUTION RANGE LOW ( ANGSTROMS ) : 10.0
REMARK      3 DATA CUTOFF (SIGMA ( F ) ) : 0.0
REMARK      3 DATA CUTOFF HIGH
REMARK      3 DATA CUTOFF LOW ( ABS ( F ) ) : NULL
REMARK      3 COMPLETENESS ( WORKING + TEST ) (%) : NULL
REMARK      3 NUMBER OF REFLECTIONS : 42851

```

Figure 2.1. PDB REMARK in 1992(A) and 1996(B). Figure is from [5].

this growth rate of PDB, the PDB current data storage format, which is based on a set of collections of record types stored as plain text files, will not be able to maintain the required level of consistency, accuracy and reproducibility for such a large data bank.

The gave rise to the 'Data Uniformity Project' by RCSB after it took the full responsibility of management of the PDB in July 1999 [36]. The main goal of this project was to validate all the data in the PDB archive and to come up with a uniform archive for the community, which will have much better querying capabilities and would be flexible and stable to growing and varying user needs. Many approaches were based on Crystallographic Information File (CIF), a standard means of information exchange in crystallography. They came up with an extended from of CIF called macromolecular CIF (mmCIF). We describe both of these data formats used by PDB in more detail below.

PDB Format is in use since the 1970's and has gone through a number of modifications since then and has benefited the research community in a number of ways. A PDB file is a collection of lines terminated by end-of-line indicator, which should be the last character of each PDB entry. Each PDB record entry has 80 columns. Each line

has a record name as the first six columns of the line, which identifies the line and has to be one of the standard pre-defined record types. A record type for a particular record can be described in more than one line. The record types can be classified into section numbers; some of the important ones are described below [7].

**Title Section** – This section consists of record types like HEADER, OBSLTE, TITLE, CAVEAT, COMPND, SOURCE, REMARKS etc, which are used to describe the experiment and biological macromolecules present in the entry.

**Primary Structure Section** – It contains the sequence of residues in each chain of macromolecule. They contain chain ids and sequence numbers to let other records link to them. This section consists of record types like DBREF, SEQRES and MODRES etc.

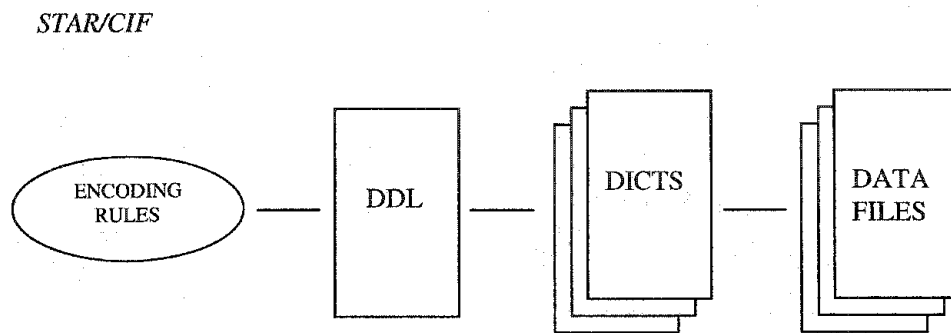
**Heterogen Section** – This section contains the complete description of non-standard residues in the entries. It has record types like HET, HETNAM and FORMUL etc.

**Secondary Structure Section** – It encloses information describing the helix, sheet and turn structures of protein and polypeptide structures. It has a record type for all the three types of structures that are HELIX, TURN and SHEET.

**Connectivity Annotation Section** – It specifies the existence and location of disulfide bonds and other linkage. It includes the record types SSBOND, LINK, HYDBOND etc.

**Coordinate Section** – This section contains the collection of atomic coordinates, and has record types MODEL, ATOM, SIGATM, TER, HETATM etc.

**Connectivity Section** – It provides the information on chemical connectivity and the only record type that comes in this section is CONNECT.



*English language Analogy*

There are 26 letters in	Words are allowable	Words are defined	This document
the alphabet. I before E	groups of letters	in a dictionary.	use only words
after C is a rule.			In dictionary.

Figure 2.2. The STAR/CIF data representation. Figure is from [6].

Macromolecular Crystallographic Information File (mmCIF) Format was developed under the support of International Union of Crystallography (IUCr) and is based on CIF data representation, which was developed to describe small molecule organic structures and the crystallographic-experiment by IUCr [6]. CIF is a subset of STAR (Self-defining Text Archive and Retrieval Format), which defines a set of encoding rules [37]. A Dictionary Definition Language (DDL) is defined, which uses these rules and also provides a framework from which a dictionary of the terms is defined. In 1990, this core dictionary developed for CIF was extended to include data items relevant to the macromolecular crystallographic experiment which currently follows DDL v2.1.1. Figure 2.2 shows components of STAR/CIF data representation.

A mmCIF file consists of a series of name-value pairs (a data item) where data name is preceded by a leading underscore(\_) to distinguish it from the data value, as compared to a PDB file, which has a series of records identified by record name of up to

```

data_PDR025
_symmetry.space_group_name_H-M 'C 2 2 21'
_cell.length_a 137.290
_cell.length_b 153.100
_cell.length_c 76.200
_cell.angle_alpha 90.00
_cell.angle_beta 90.00
_cell.angle_gamma 90.00
#
loop_
_atom_site.id
_atom_site.label_atom_id
_atom_site.label_alt_id
_atom_site.label_comp_id
_atom_site.label_asym_id
_atom_site.label_seq_id
_atom_site.Cartn_x
_atom_site.Cartn_y
_atom_site.Cartn_z
1 O5* ? G A 1 -3.897 61.994 -24.841
2 C5* ? G A 1 -5.016 62.932 -24.760
3 C4* ? G A 1 -4.695 64.310 -25.301
4 O4* ? G A 1 -4.459 65.240 -24.222
5 C3* ? G A 1 -3.413 64.316 -26.115
6 O3* ? G A 1 -3.497 65.336 -27.093
7 C2* ? G A 1 -2.354 64.634 -25.084
8 C1* ? G A 1 -3.091 65.629 -24.212
9 N9 ? G A 1 -2.631 65.588 -22.836
10 C8 ? G A 1 -2.125 64.484 -22.203
11 N7 ? G A 1 -1.781 64.713 -20.965
12 C5 ? G A 1 -2.077 66.050 -20.769

```

Figure 2.3. Sample PDB data in mmCIF format. Figure is from [7].

6 characters. Another important difference between these two formats is that it is easy to convert a mmCIF data file to a PDB file without losing any information, but it is not possible to automatically convert a PDB file to a mmCIF file because some of the mmCIF data is either not present in PDB file or is taken from the REMARK records of PDB file. The application program that runs in UNIX called CIFTr is used to translate files in mmCIF format to files in PDB format. Figure 2.3 shows a few lines of data in mmCIF format.

On June 3<sup>rd</sup> 2003, PDB officially released XML data files for beta test. All the released PDB entries were now made available in XML format. These XML data files



were created by software translation of the mmCIF data files that were created as part of the PDB Data Uniformity Project. The mmCIF data files used the data items defined in the PDB Exchange Dictionary. The XML data files conform to an XSD style XML schema derived from the PDB Exchange Dictionary. Because of this, the element and attribute names in the XML data files directly correspond to the item names defined in PDB Exchange Dictionary. PDB maintains data in all these formats. As a result a user has the option of viewing the protein data in any format with the help of various searching tools provided by PDB.

We have seen the complexity and evolution of data in PDB (not even mentioning the variability of biological data stored and processed in other data sources), which shows the need of establishing a platform to integrate multiple data sources. In the following section, we will review the key characteristics of biological data before we propose our solution.

### **2.1.2 Characteristics of Biological Data Sources**

Biological data is very complex in nature as compared to traditional data so we need to represent and define it properly without losing any information. For example, some attributes may have a high range of possible values, so we need an appropriate set of data types to cover all types of data that could exist. This may even require having a data type for a particular instance of data, as we cannot lose any information that piece of data might be carrying by excluding it. Another problem is that the schema in biological databases changes at a vary rapid rate. Even the very popular databanks like GenBank and PDB come up with new releases involving different schemas from time to time. This puts additional load on systems that try to integrate these various data sources because the system has to keep track of all these changes.

Many of the existing biological data banks contain the same data deposited by various biologists that may be represented differently. Even if they are submitting the data to the same depository, it may vary based on the representation used by a biologist. Because of the complexity of the biological data, an entity might be interpreted differently by different users. This makes things more difficult when the same entity is presented in various data banks or when they are interlinked. So we need to have a storage system for standardized formats that would help in maintaining uniformity while representing such data from various depositors.

The read queries on biological data are greater in number as compared to write queries (updates and insertions) but the query pattern of users is very different and unexpected as compared to ones on traditional databases. This makes it very difficult to come up with an efficient way of indexing such data. Most of the time users are biologists who only know what data they need but have little or no knowledge of the database schemas used to store such data and hence are not able to query these data in an efficient manner. Moreover, quite complicated queries are usually required to get the desired data making things more difficult for the biologist users. So a bioinformatics database system needs to provide some user friendly tools to hide all these complexities and provide a simple and efficient interface to the users.

Apart from updating the frequent modifications made to a biological data record in the database, we also need to keep track of earlier data before it was modified. Old data cannot be discarded because there might be various biologists using this data for their research work. This leads to the problems associated with maintaining multiple versions of data objects.

### 2.1.3 Summary

- Biological data is more complex than data in other domains. It needs complex data structures and relationships to ensure there is no information lost during the modeling process.
- Unlike traditional business data sources, most users only generate read-access patterns into biological data sources since only curators or system administrators have the privilege to update it.
- Users of biological data sources may want to access old versions of data for analysis and comparison. The system should support multi-version data stores and queries.
- The amount and range of variability of biological data is extremely high. Biological data systems should be flexible to extend or change their data types and values.
- Each biological data source has its own schema (ER, XML, RDFS,...etc) so schema matching and evolution techniques should be employed for data exchanging and data migration with others.
- Due to the complexity of the biological data, biological data source system must support complex query formulation and provide predefined query templates.

## 2.2 Biological Data Source Integration

The biological data sources store a wide range of subjects, objects and data types such as gene sequences, gene expression data, protein sequences, protein structures, and pathways information. Each data source may include several different data formats as we have presented in section 2.1, and the data stored inside each data source may be semantically interrelated with others even though they are often distributed and can be heterogeneous [38]. Furthermore, biologists/scientists usually want to query multiple data sources by using a consistent, friendly, and integrated interface to gather a query re-

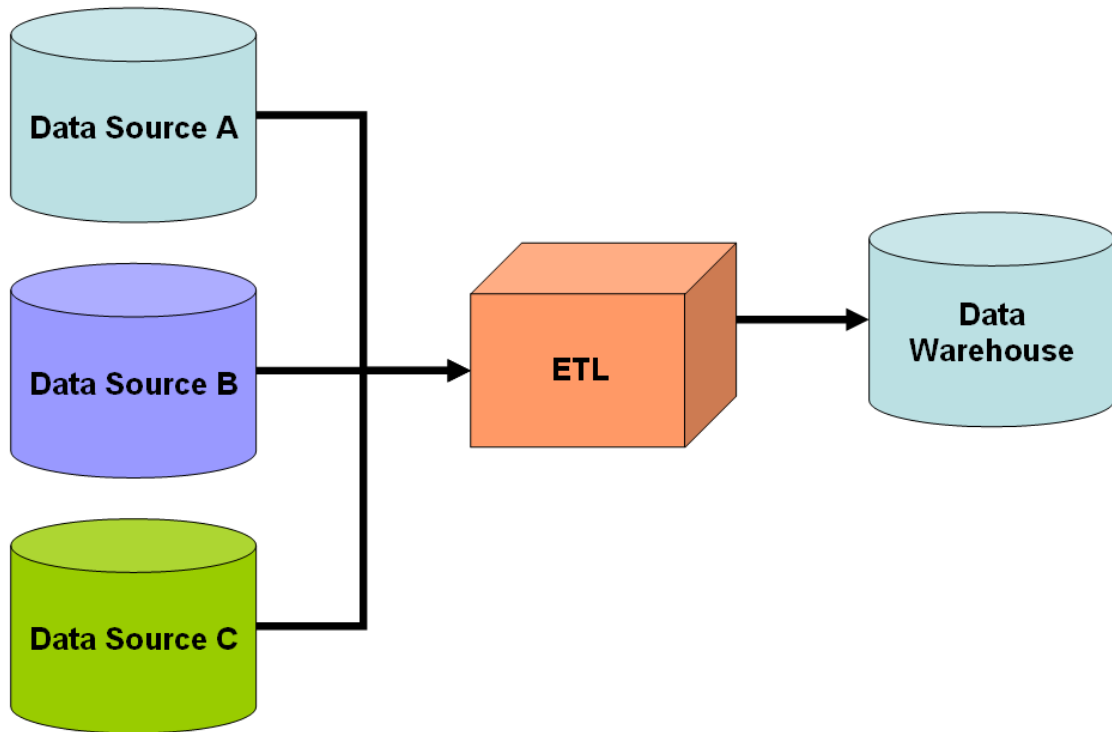


Figure 2.4. Data Warehouse Integration. Figure is from [8].

port. To facilitate this goal, researchers proposed several different integration approaches. In section 2.2.1, we introduce the idea of data warehouse integration. In section 2.2.2, we describe the concept of mediator-based integration and show the differences with the data warehouse approach. The power of Ontology-based integration is discussed in section 2.2.3. In section 2.2.4, we show the service-based approach that is utilized as the primary scheme of our BioServiceBroker system. A comparison of these approaches will be given in section 2.2.5

### 2.2.1 Data Warehouse Approach

Data warehouse integration collects the data from selected data sources of the specified domain, and stores it into a local data warehouse based on a pre-unified data

schema. As shown in Figure 2.4, it requires a ETL (Extract, Transform, Loading) process that is the essential way to facilitate the parsing, converting, and physically loading of the source data sets into their respective local data warehouse tables through data mapping techniques.

Extracting the data from the data sources is the first stage of an ETL process. A data warehouse system may collect data from several different sources, and each separate data source may use a different data structures and formats. Extraction procedure is responsible for parsing source data into a expected structure for transformation processing. The transform stage may involve some selection, cleaning, translation and joining operations to meet the business and technical specifications of the end target. The last stage is the loading phase that determines the timing and scope to update or insert data into the data warehouse [8].

Since all data is stored locally and queries are executed at the same place rather than at the original data sources, overall network transmissions and response time can be reduced. In addition, query plans are easier to be formed and executed more efficiently. However, this approach needs to constantly monitor and update the changes of the data sources in order to maintain the integrity, consistency, and reliability of the data [39, 40, 41].

### **2.2.2 Mediator-based Approach**

Unlike in the data warehouse approach that converts the data format of the original sources and stores everything locally, there is no data physically transferred/converted in mediator-based integration systems. However, each data source needs a wrapper/adaptor to transform the local query results into an easily processed form for the data integration system (see Figure 2.5). The mediator-based approach focuses on query translation that needs to establish the mapping between each data source schema and the single mediator

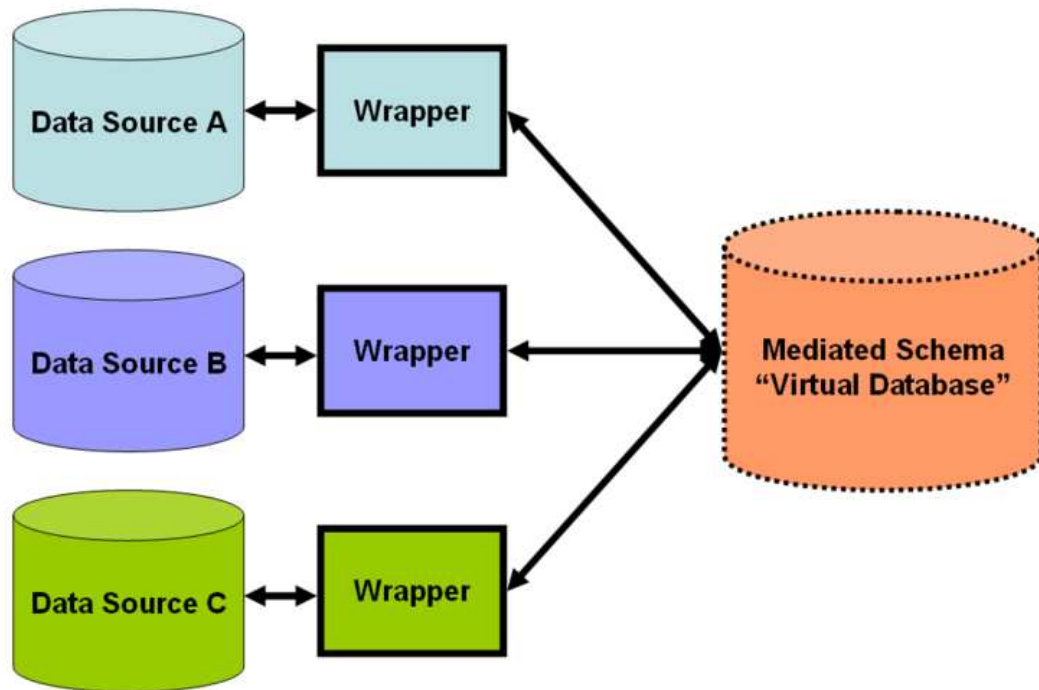


Figure 2.5. Mediator-based Integration. Figure is from [8].

schemas. A mediator is responsible for reformulating a query given by a user on a single mediated schema into several sub-queries on the underlying data source schema at run time. This process can also be called as view based query answering because we can consider each of the data sources to be a view over the mediated schema. The obvious drawback is the need to rewrite the view for mediated schema once a new source is to be integrated and/or an existing source changes its schema [8].

### 2.2.3 Ontology-based Approach

The term "*ontology*" was introduced by Gruber [42] as an explicit specification of a conceptualization, and originally used in philosophy where it is concerned with the objects

of knowledge. A conceptualization refers to an abstract model of how people commonly think about a real thing in the world, and explicit specification means that concepts and relationships of an abstract model receive explicit names and definitions. An ontology gives the name and the description of the domain specific entities by using predicates that represent relationships between these entities. The ontology provides a vocabulary to represent and communicate domain knowledge along with a set of relationships containing the vocabulary terms at a conceptual level. Therefore, because of its potential to describe the semantics of information sources and to solve the heterogeneity problems, an ontology might be used for data integration tasks.

In the bioinformatics field, a large number of biological ontologies that describe related domain knowledge have become available in recent years covering genomics and proteomics domain knowledge. Other ontologies describe other knowledge domains in bioinformatics and medicine. For instance, Gene Ontology(GO) [43] describes cellular components, molecular function and biological process of genes and proteins; TAMBIS ontology covers a wide range of biological concepts and is used as an unified schema to support queries over multiple data sources [44]; MeSH is used for indexing, classifying and searching for biomedical related information [45]. These ontologies can be employed to support the integration of multiple biological data sources since the user query may involve several different types of biological knowledge that is spread over many ontologies. Based on their domain knowledge, we can establish ontological annotations of data sources and mappings between data values and schemas, which is utilized to formulate subqueries of the selected data sources. In short, the primary role of ontologies is to support query processing.

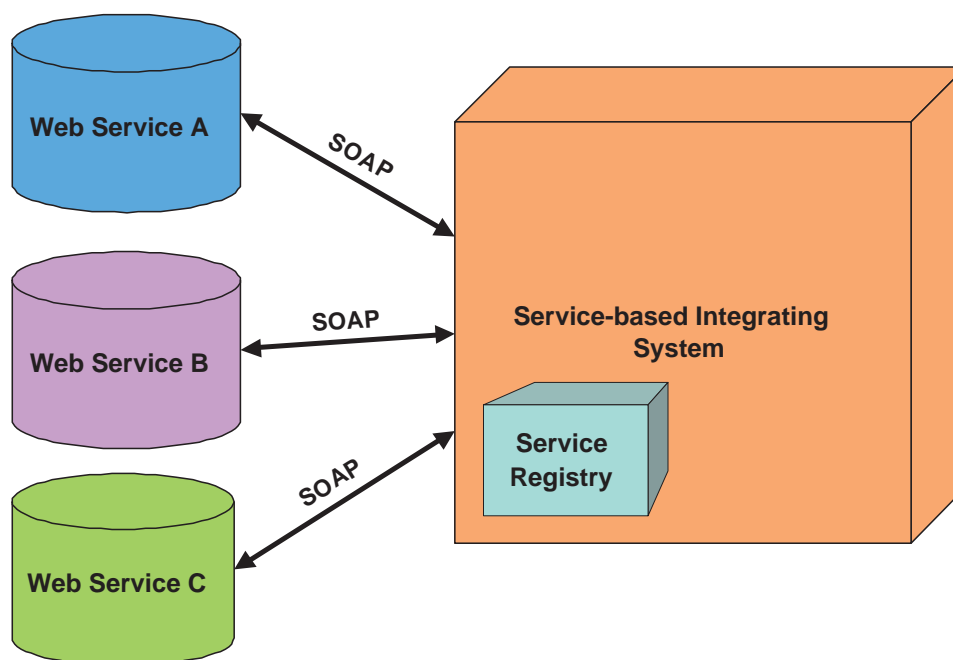


Figure 2.6. Static Service-based Integration.

#### 2.2.4 Service-based Approach

Web services technology is the basis of SOA (Service Oriented Architecture), which not only increases the flexibility and interoperability of business software development, but also provides a new way for scientists to retrieve data and utilize tools more efficiently. Especially, it makes it possible to exchange information and integrate data between the heterogeneous/distributed systems through SOAP messages.

Currently, many bioinformatics data and tools can be accessed/invoked through varied service operations provided by different web services. Hence the need of finding solutions to integrate the existing web services. Jung and cho [25] proposed a service-based integration system (SPDBSW) to combine a local data source (SPDBS) with external web services such as OLS, KEGG, and NCBI. However, this static service-based approach (see Figure 2.6) is not enough to handle numerous unpredictable issues (such as low availability, service under maintenance, and long response time) that might oc-



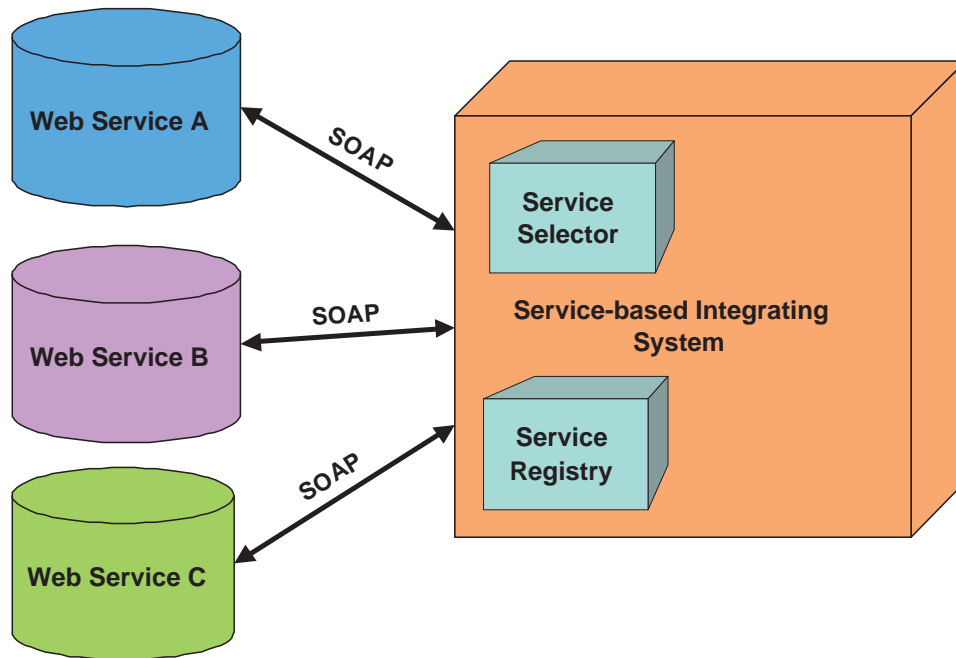


Figure 2.7. Dynamic Service-based Integration.

cur during service invocations since the number of services provided is limited and the services are specified at the system development stage.

In contrast, our BioServiceBroker system is a dynamic service-based approach (see Figure 2.7), which provides an integrating platform that can determine and coordinate web services at run time based on QoS (Quality of Service) query factors specified by client requests. The detailed system architecture description will be given in chapter 5.

### 2.2.5 Comparison

Each approach has its own advantages and disadvantages as shown in Table 2.1, system developers need to determine the essential goal and properties of their integration system and choose the appropriate framework. Table 2.2 lists some existing integration work in terms of their approaches, data models and level of transparency.

Table 2.1. Integration Approaches Comparison.

Approach	Advantage	Disadvantage
<i>DataWarehouse</i>	<ol style="list-style-type: none"> <li>1. Query plans are easier to be formed</li> <li>2. Better Query efficiency</li> <li>3. Reduce overall network traffic</li> </ol>	<ol style="list-style-type: none"> <li>1. High maintenance costs</li> <li>2. Data integrity issues</li> <li>3. Consistency issues</li> </ol>
<i>Mediator – based</i>	<ol style="list-style-type: none"> <li>1. Query plans are flexible</li> <li>2. Dynamic data retrieval</li> <li>3. No data maintenance issues</li> </ol>	<ol style="list-style-type: none"> <li>1. Query plans are difficult to be formed</li> <li>2. Maintain source schemas</li> </ol>
<i>Ontology – based</i>	<ol style="list-style-type: none"> <li>1. Query plans are flexible</li> <li>2. Dynamic data retrieval</li> <li>3. No data maintenance issues</li> <li>4. Cross-references are easier to be established</li> <li>5. Eliminate the mapping conflict problems</li> </ol>	<ol style="list-style-type: none"> <li>1. Query plans are difficult to be formed</li> <li>2. Maintain source schemas</li> </ol>
<i>Service – based</i>	<ol style="list-style-type: none"> <li>1. Query plans are flexible</li> <li>2. Dynamic data retrieval</li> <li>3. No data maintenance issues</li> <li>4. Cross-references are easier to be established</li> <li>5. Eliminate the mapping conflict problems</li> <li>6. High scalability</li> <li>7. Low maintenance cost</li> </ol>	<ol style="list-style-type: none"> <li>1. Maintain web service descriptions</li> <li>2. Determine equivalent web services</li> </ol>

Table 2.2. Integration System Comparison. Table is from [9].

	<i>Aim of integration</i>	<i>Data model</i>	<i>Source model</i>	<i>User model</i>	<i>Level of transparency</i>	<i>Overall integration approach</i>
<b>SRS</b>	Portal, browsing-based	Linked text records	Mostly complementary, some overlap	No critical expertise	Sources specified by user	Navigational, (with local index)
<b>K2/ Bio-Kleisli</b>	Query-oriented	Semi-structured, object-oriented	Mostly complementary	Expertise in query language	Sources specified by user	Mediator-based
<b>TAMBIS</b>	Query-oriented	Structured, object-relational	Mostly complementary	Interactive query formulation	Sources hard-wired by system	Mediator-based
<b>Discovery-Link</b>	Query-oriented middleware	Structured, object-relational	Mostly complementary, some overlap	Expertise in query language	Sources selected by system	Mediator-based
<b>BACIS</b>	Query-oriented	Structured, object-relational	Mostly complementary, some overlap	Interactive query formulation	Sources selected by system	Mediator-based
<b>Bio-Navigator</b>	Portal, browsing-based	Text model	Mostly complementary	No critical expertise	Sources specified by user	Navigational
<b>GUS</b>	Query-oriented	Structured, relational	Mostly complementary	Expertise in query language	N/A <sup>3</sup>	Warehouse
<b>KIND</b>	Query-oriented	Semi-structured, object-oriented	Mostly complementary	Expertise in query language	Sources specified by user	Mediator-based
<b>Entrez</b>	Portal, browsing-based	Linked text records	Mostly complementary, some overlap	No critical expertise	Sources specified by user	Navigational

## CHAPTER 3

### MULTI-LEVEL CONCEPTUAL DATA MODELING

Biological data such as protein structure and function, DNA sequences, and metabolic pathways require conceptual modeling characteristics that are not available in many conceptual data models, including the widely used ER (Entity-Relationship) model and its variant the EER (enhanced-ER) model. Many researchers have proposed extensions on existing conceptual data models to represent biological and bioinformatics data. For example, Keet [33] discusses the characteristics of biological data and its effect on ER, OO and Object Role Modeling(ORM) methodologies. Chen and Carlis [46] present a genomic schema element data model to capture the basic biological sequence notion. Ram [47] also proposes a semantic model for 3D protein structure and DNA sequences. Our extensions differ from these previous works because we focus on classifying and modeling the semantic concepts and relationships between them. In the biomedical/biological research fields, some semantic data (experiments, evidence, annotation, etc.) can be classified into different levels (from the DNA/RNA level to higher levels such as cells, tissues, organs, and biological systems) based on different degrees of abstraction. Hence, it is important to represent and integrate them across different levels.

In this chapter, we first review the basic concepts of conceptual data modeling and how are they used to create database schemas in section 3.1. We then describe multi-level concepts, one of the essential characteristics of biological data, and present a multi-level conceptual data model in section 3.2 . Our approach facilitates more accurate modeling of biomedical ontologies, and a better understanding of the data stored in various biological data sources. We propose intra-level and inter-level relationships and give formal

definitions illustrated with examples in order to precisely describe relationship/behavior among concepts at multiple levels in section 3.3. In section 3.4, we will demonstrate the power of the multi-level approach for data source integration by distinguishing horizontal, vertical and hybrid integrations.

### **3.1 What is Conceptual Data Modeling?**

Analyzing and describing various data objects (or entities) and the relationships they share among themselves is known as Conceptual Data Modeling and it is usually the first process in designing a database or for describing the data requirements for a particular domain of knowledge. The first step is to create a conceptual schema, which is a conceptual representation of the needed data structures for an application. This step is used to identify entity types; associate data attributes to them, and identify relationships among the entity types. This helps to remove redundancy of data and in getting data in standard normalized form. It is not dependent on which DBMS (Database Management System) we use to implement the database system. Although many conceptual data models have been proposed [20, 33], two of the most commonly used models are the Entity-Relationship approach and the Object Model approach. Many variations of these two main approaches also exist. The main objective for conceptual data modeling is to make sure that all data objects and relationships required by the database are completely and correctly specified, and we do not leave out any data or any dependencies between various data objects.

### **3.2 Multi-level Concept**

In this section, we introduce multi-level concepts by briefly describing some functions and relationships between elements inside a human body and applying multi-level

concepts on them to illustrate the need for multi-level modeling. In addition, we give biological examples and present the associated EER conceptual modeling notations to show that multi-level modeling may be useful in biological/biomedical domains. The EER model has a diagrammatic notation known as EER diagrams. We follow the notation used in [20](Chapters 3 and 4). We now present a high-level summary of the characteristics of biological data.

Cell theory states that all living things are composed of cells, and cells are the basic units of structure and function in living things [48]. In a multi-cellular organism such as a human, different types of cells perform different tasks. Some cells can be responsible for extracting nutrients, and others can be functioning as receptors or responsible for other functions. Tissues are groups of similar cells specialized for a single function, such as epithelial tissue that line the chambers of the heart to prevent leakage of blood and nervous tissue that receive messages from the body's external and internal environment, analyzes the signals, and directs the response. An organ is a group of tissues that work together to perform a complex function. For instance, the heart is the most important organ inside the human body, which is mainly composed of epithelial tissues, connective tissues, and nervous tissues. Biologists classify the human body into eleven organ systems (nervous system, circulatory system, skeletal system, muscular system, and so on), and each system is a group of organs that work together to perform closely related functions. For example, the digestive system involves a series of processes (Ingestion, Mechanical digestion, Chemical digestion, Absorption, Elimination) within several organs, which break food down into small molecules that can be passed through the cells that need them. Based on this very brief description of the human body, we may define six different data abstraction levels for human biology: molecule, sub-cell, cell, tissue, organ and system.

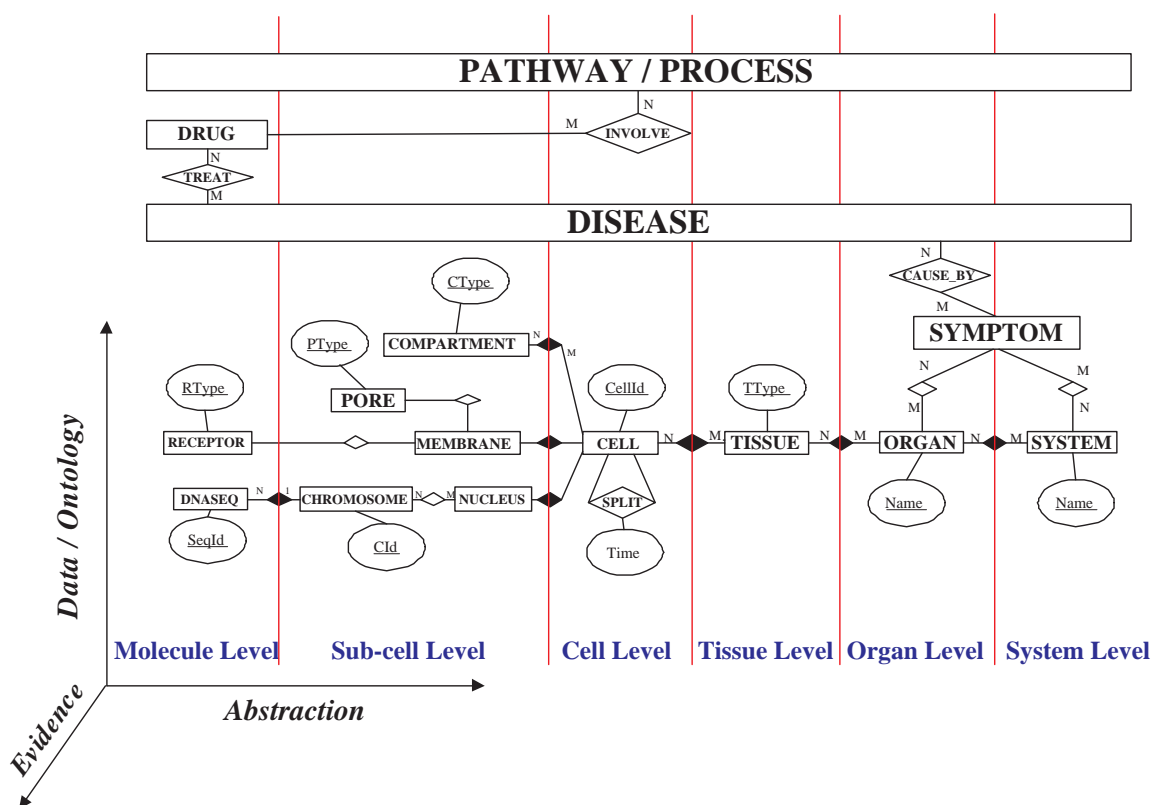


Figure 3.1. Multi-level Biological System.

**EXAMPLE 1.** Figure 3.1 shows the EER conceptual modeling of a cell system [49]. Usually a cell is surrounded by a plasma membrane (Entity type Membrane). There are channels (Entity type Pore) embedded within this membrane that allow different molecules to pass through the membrane. Cell surface membranes also contain receptor (Entity type Receptor) proteins that allow cells to detect external signalling molecules such as hormones. Cells also have a set of little organs, called organelles (Entity type Compartment), specialized for carrying out one or more vital functions. In every second there are millions of biochemical reactions that happen in a cell, integrating into diverse types of biological processes (Entity type Pathway). Genetic materials such as DNA are packaged into chromosomes (Entity type Chromosome), which are stored in the nucleus (Entity type Nucleus) of a cell. Cells grow through successive cell divisions. Relationship Split

denotes this cellular metabolism. The same type of cells can be assembled into a group of cells (*Entity type Tissue*) executing special functions.

Due to the complexity and huge amount of data stored and processed in a single cell (not mentioning the even higher information density in tissue, organ, and organism system), it will be more efficient and easy to integrate if we can classify and relate data at different abstraction levels. Each level can be defined by the degree of abstraction of the data, which determines the amount of detail of information that is represented. The higher abstraction level data contains less detailed information than in lower abstraction levels [50].

### 3.3 Multi-level Relationships

Furthermore, events occurring at one level can effect and be affected by events at different levels of abstraction, scale or time [51]. Between the levels, there may be some transitions and connections, which need what we call a *vertical approach* to integrate these data sources to explore further information. We now define more formally the concepts of our multi-level model.

#### 3.3.1 Formal Definitions

- Level - is defined by the degree of abstraction, which is subjective. Hence, we may define different numbers of levels for specific applications based on the domain knowledge. In this chapter, we define six levels (molecule, sub-cell, cell, tissue, organ and system) based on human biology knowledge.
- Class and instance - In this chapter, we define a class as a concept that can contain a collection of individual instances (or objects) assigned to the same level.



- Binary relationship - is a concept that specifies an association between two classes, two instances, or a class and an instance.

For each binary relationship, there are two participating entities E (operands/objects), which can be continuants(C)[52], processes(P) or functionalities(F).

- Continuant(C) - Biological entities such as molecule, cell, membrane, and organ.
- Process(P) - Each process may have input, output, catalyst, and factory. The detailed discussion of process concepts has been proposed in our previous work [53]. We have added the concept of factory to our previous work to specify the location of the process.
- Functionality(F)- Describes the role, function, and propose of continuants or processes.

A large number of relationship types have been defined [20, 52]. In a multi-level model, we focus on differentiating intra-level versus inter-level relationships, since these will be crucial in multi-level conceptual modeling. Also, it is important to note that a particular concept may be represented as a class at one level of abstraction, and an instance at another level.

**Definition 1.** Intra-level relationship

We say that  $R$  is an intra-level relationship between  $E_1$  and  $E_2$ , if  $E_1$  and  $E_2$  are classes or instances at the same level of abstraction. We call these *horizontal* or *H relationships*.

**Example 1:** Esophagus (organ) "*attach\_to\_H*" stomach (organ)

**Example 2:** Yellow marrow (tissue) "*is\_a\_H*" bone marrow (tissue)

**Definition 2.** Inter-level relationship

We say that  $R$  is an inter-level relationship between  $E_1$  and  $E_2$ , if  $E_1$  and  $E_2$  are classes or instances at different levels. We call these *Vertical* or *V relationships*.

**Example 1:** Red cells (cell) are "*produced\_by\_V*" red marrow (tissue)

**Example 2:** Smooth muscle tissue (tissue) is "*part\_of\_V*" Stomach(organ)

**Definition 3.** is\_a (superclass/subclass) relationship

We say that  $R$  is an is\_a relationship between  $E_1$  and  $E_2$ , where  $E_1$  is the subclass and  $E_2$  is the superclass, if  $\forall e \in E_1$ , then  $e \in E_2$ , where  $E_1$  and  $E_2$  are classes,  $e$  is an instance of the class, and  $E_1, E_2$  need to be the same type of operands (C, P or F) at the same level.

**Example 1:** Yellow marrow (C)(tissue) "*is\_a*" bone marrow (C)(tissue).

**Example 2:** Intracellular signaling cascade (F) "*is\_a*" signal transduction.

**Example 3:** Catabolic process (P) "*is\_a*" metabolic process (P).

**Example 4:** Rods (C)(cell) "*is\_a*" photoreceptor (C)(cell).

**Definition 4.** part\_of-H relationship

We say that  $R$  is a part\_of-H relationship between  $E_1$  and  $E_2$ , if given  $e_i \in E_1$  and  $e_j \in E_2$ , then  $e_i$  is a sub-component of  $e_j$ , where  $E_1$  and  $E_2$  need to be the same type of operands (C, P or F) at the same level.

**Definition 5.** part\_of-V relationship

We say that  $R$  is a part\_of-V relationship between  $E_1$  and  $E_2$ , if given  $e_i \in E_1$  and  $e_j \in E_2$ , then  $e_i$  is a sub-component of  $e_j$ , where  $E_1$  and  $E_2$  need to be the same type of

operands (C, P or F) at different levels of abstraction.

**Definition 6.** produced\_by relationship

We say that  $R$  is a produced\_by relationship between  $E_1$  and  $E_2$  through a process type  $P$ , if  $\exists p \in P$ , such that  $E_1$  can be produced through  $p$  with participating operand  $E_2$ , where  $E_1$  and  $E_2$  are continuants, and  $P$  is a process set.

**Example:** Red cells (cell) are "produced\_by" red marrow (tissue)

**Definition 7.** covered\_by relationship

We say that  $R$  is a covered\_by relationship between  $E_1$  and  $E_2$ , if  $SpatialShape(E_1)$  around  $SpatialShape(E_2)$  where  $E_1$  and  $E_2$  are continuants, and *around* means encapsulates the shape but may not surround it completely.

**Example:** Spinal cord (tissue/component) is "covered\_by" meninges (tissue/component).

**Definition 8.** has\_functionality relationship

We say that  $R$  is a has\_functionality relationship between  $E_1$  and  $E_2$ , if  $E_1$  supports certain biological function of  $E_2$ , where  $E_1$  can be a continuant(C) or process(P) and  $E_2$  is a functionality (F).

**Example:** Axial skeleton (sub-system) "has\_functionality" supporting the central axis of the body.

**Definition 9.** preceded\_by relationship

We say that  $R$  is a preceded\_by relationship between  $E_1$  and  $E_2$ , if  $TemporalStamp(E_1) < TemporalStamp(E_2)$ , where  $E_1$  and  $E_2$  are processes (P).

**Example 1:** Digestion (P) "preceded\_by" ingestion (P).

**Example 2:** Translation (P) "*preceded\_by*" transcription (P).

In Figure 3.1, intra-level relationships are shown as clear diamond shapes, whereas inter-level relationships are shown as filled diamond shapes.

### 3.3.2 Properties Summary

In mathematics, a binary relation is an association of elements of one set/class with elements of another (perhaps the same) set/class. It has been used widely in conceptual modeling in mathematics and computer science. In this section, we review the mathematical definitions of transitive, symmetric, asymmetric and reflexive properties that help us to describe the characteristics of the modeling relationships that we introduced in section 3.3.1, and we demonstrate these properties that can be applied to our proposed relationships in Table 3.1.

- *Transitive*

We say that  $R$  is a transitive relation over a set  $S$ ,  $\forall x, y, z \in S$ , if  $xRy$  and  $yRz$  hold, then  $xRz$  must hold.

**Examples:** Photoreceptors (cell) are "part\_of\_V" Retina (tissue) and Retina (tissue) is "part\_of\_V" eye (Organ), so we can conclude that photoreceptors (cell) are "part\_of\_V" eye (organ).

- *Symmetric*

We say that  $R$  is a symmetric relation over a set  $S$ ,  $\forall x, y \in S$ , if  $xRy$  holds, then  $yRx$  must also hold.

**Examples:** Esophagus (organ) "attach\_to\_H" stomach (organ).

- *Asymmetric*

We say that  $R$  is an asymmetric relation over a set  $S$ ,  $\forall x, y \in S$ , if  $xRy$  holds, then  $yRx$  must not hold.

**Examples:** Smooth muscle tissue (tissue) is "part\_of\_V" Stomach (organ), but Stomach (organ) is not "part\_of\_V" smooth muscle tissue (tissue).

- *Reflexive*

We say that  $R$  is a reflexive relation over a set  $S$ ,  $\forall x \in S$ ,  $xRx$  always holds.

**Examples:** Retina (tissue) is "part\_of\_H" Retina (tissue).

Table 3.1. Properties of Relationships in Multi-level Data Model.

Relationship	Transitive	Symmetric	Asymmetric	Reflexive
is_a	+	-	+	+
part_of-H	+	-	+	+
part_of-V	+	-	+	+
produced_by	n/a	-	+	-
covered_by	n/a	-	+	-
attach_to	+	+	-	+
has_functionality	n/a	-	+	-
preceded_by	+	-	+	-

Table 3.1 shows the properties of the relationships that we introduced in section 3.3.1. These can be used to infer additional relationships.

### 3.4 Multi-level Integration

Biological data sources include many types of data. One type includes many ontologies as well as data for various biological concepts. A second dimension is to characterize the data/ontology based on the different abstraction levels. A third type can be categorized as evidence data, such as experiment results, publications, and other results of studies. In Figure 3.1, we categorized three different dimensions to characterize biological data: the abstraction level dimension, the data/ontology dimension, and the experimental result/evidences dimension. Each entity can have several experimental results or

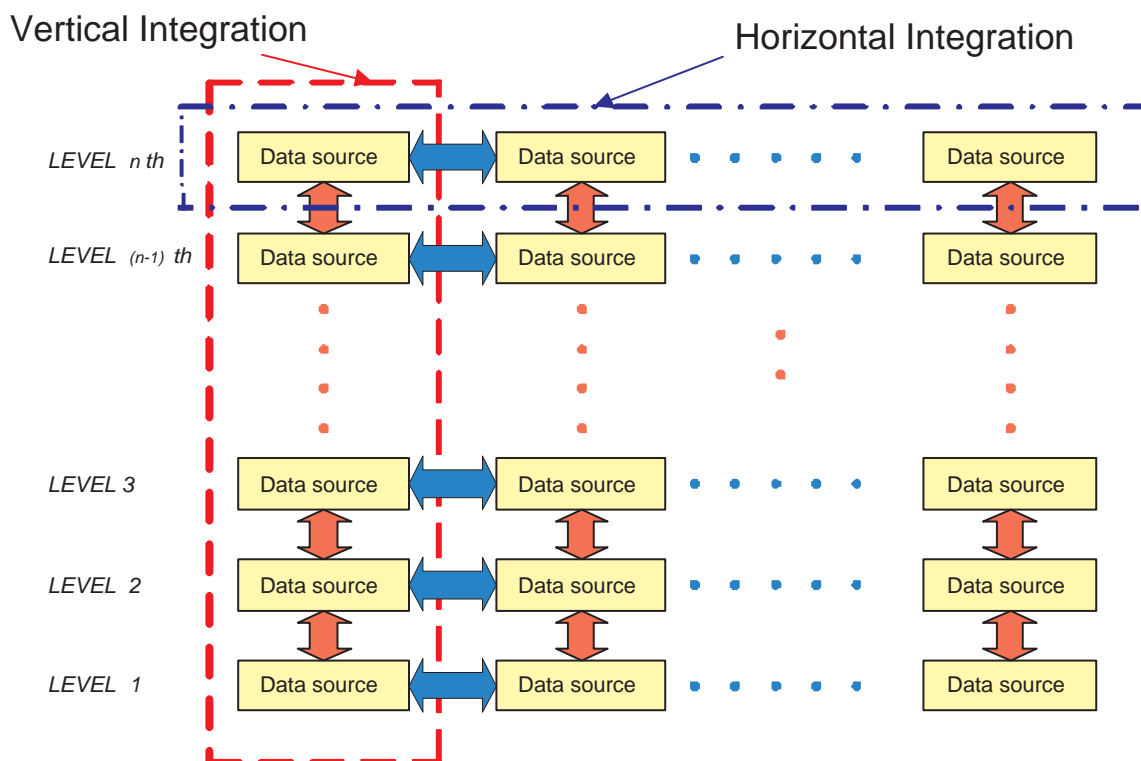


Figure 3.2. Horizontal and Vertical Integration.

evidences related to it, such as how it was produced or discovered. At the data/ontology dimension, various ontologies can be defined, such as anatomy ontologies, disease ontologies, drug ontologies, and so on. The concepts within these ontologies can then be characterized based on their abstraction level dimension using our multi-level framework.

Figures 3.2 and 3.3 demonstrates the multi-level approach for data source integration by distinguishing horizontal, vertical and hybrid integration. The horizontal approach integrates data sources at the same abstraction level. In contrast, the vertical approach integrates data sources from different abstraction levels. However, each data source may store data fragments at more than one level of abstraction as shown in Figure 3.3. Through the hybrid approach, we can integrate data fragments at the same level between different data sources and achieve the vertical integration between them at a

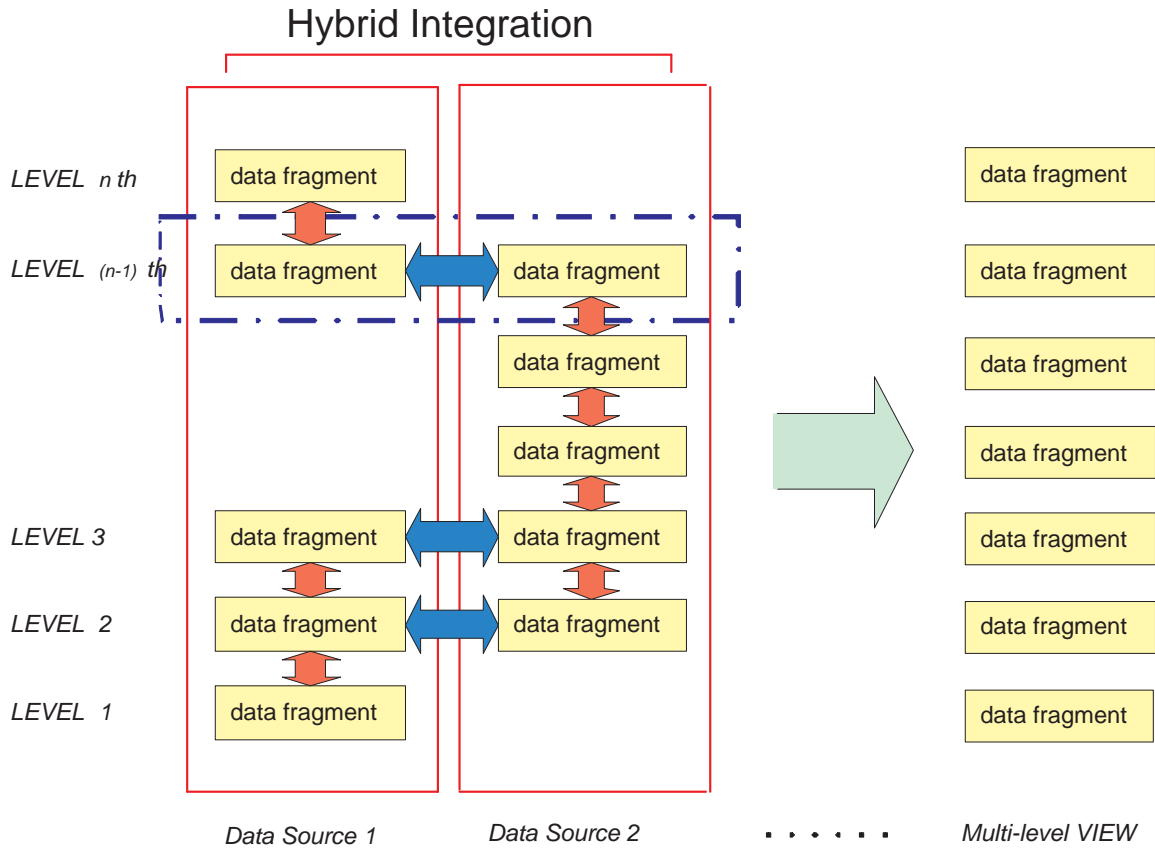


Figure 3.3. Hybrid Integration.

later stage. The multi-level view can generate data at multiple levels of abstraction for further processing by means of hybrid integration.

Furthermore, the service-based integration introduced in section 2.2.4 can employ multi-level conceptual data modeling techniques to establish service models for each service operation provided by register web service sites.

## CHAPTER 4

### BIOINFORMATICS WEB SERVICES

Web services technologies provide a language-neutral, world-wide, and platform-independent programming model that accelerates software development and resource integration over the Internet. Web service providers use the standard Web Services Description Language (WSDL) to describe location, transport protocols, operations, and messages of their services and how to invoke the service operations inside, which allows programs submit requests to service providers over the Internet through the XML-based simple object access protocol (SOAP) for data processing or gathering.

Furthermore, application integration through web services yield flexible loosely coupled systems and can be deployed quickly and recomposed to new services since web services technologies are easily to incorporate with existing applications. These features make web services a good choice for deploying bioinformatics applications. In this chapter, we will briefly introduce the web services technology and related industrial standards in section 4.1. In section 4.2, we give a overview of key bioinformatics web service providers. In section 4.3, we classify the existing bioinformatics web services into three categories based on their functionalities. The QoS measures for web services will be reviewed in section 4.4.

#### 4.1 Introduction to Web Services

The web service is defined by W3C (World Wide Web Consortium) as a software interface standard to provide interoperable machine-to-machine interaction over a network. Its definition covers many different system service frameworks (see Figure 4.1),



Name	Platform	Destination	Specifications	Protocols
ActionWebService	Ruby (on Rails)	Client/Server	?	SOAP, XML-RPC, WSDL
Apache Axis	Java/C++	Client/Server	WS-ReliableMessaging, WS-Coordination, WS-Security, WS-AtomicTransaction, WS-Addressing	SOAP, WSDL
Apache Axis2	Java/C	Client/Server/Asyn Sypport	WS-ReliableMessaging, WS-Security, WS-AtomicTransaction, WS-Addressing, MTOM, WS-Policy, WS-MetadataExchange	SOAP, MTOM, WSDL 2.0, WSDL
AlchemySOAP	C++	Client/Server	WS-Addressing	SOAP
csoap	C	Client/Server	?	SOAP
JSON-RPC-Java	Java	Server	???	JSON-RPC
JSON-RPC-Lua	Lua	Server	???	JSON-RPC
Java Web Services Development Pack	Java	Client/Server	WS-Addressing, WS-Security, ???	SOAP, WSDL, ???
NuSOAP	PHP	Client/Server	?	SOAP, WSDL
Web Services Interoperability Technology	Java	Client/Server	WS-Addressing, WS-ReliableMessaging, WS-Coordination, WS-AtomicTransaction, WS-Security, WS-Security Policy, WS-Trust, WS-SecureConversation, WS-Policy, WS-MetadataExchange	SOAP, WSDL, MTOM
Web Services Invocation Framework	Java	Client	???	SOAP, WSDL
Windows Communication Foundation	.Net	Client/Server ?	WS-Addressing, WS-ReliableMessaging, WS-Security	SOAP, WSDL
XFire	Java	Client/Server	WS-Addressing, WS-Security	SOAP, WSDL
XML Interface for Network Services	Java	Server ?	??	SOAP, XML-RPC
gSOAP	C/C++	Client/Server	WS-Addressing, WS-Discovery, WS-Enumeration, WS-Security	SOAP, XML-RPC, WSDL
Zolera SOAP Infrastructure (ZSI)	Python	Client/Server	???	SOAP, WSDL

Figure 4.1. Web Service Frameworks.

but in common usage the term refers to clients and servers that communicate using XML messages that follow the SOAP (Simple Object Access Protocol) standard [54]. Web services are not the only technology to solve the problem of distributed resource invocation. CORBA (Common Object Request Broker Architecture) is the first one to deal with this issue, which is defined by the Object Management Group (OMG). DCOM (Distributed Component Object Model) and RMI (Remote Method Invocation) proposed by Microsoft and Sun microsystems respectively are also two available solutions to provide an extensible way for distributed applications to communication each other. Even though these tools have had a major impact on large scale environments, they have several critical problems that have make them unfeasible for current complex operating environment. First, because of their proprietary nature, they can not provide effective cross-vendor communication defeating the point of distributed computing. Second, all

the data passed between hosts assumed that both client and server knew exactly what inputs and outputs to expect. There is no SOAP-style object description available. Third, binary-encoded objects are far more difficult to decode and analyze than XML-style used in web services. The importance of web services is that they raise the distributed computing abstraction to a higher level by removing a large part of intercommunications difficulties. The essential standards are as follows:

- **XML (Extensible Markup Language)**

The XML is originally inspired by HTML, but by its simplicity and fragmentation. HTML is a very generic and structurally flat document type, which describes the basic components of a Web page. However, it is too simple to describe documents outside the web area and is becoming inadequate at describing documents within specific area. Document designers and authors are motivated to use XML because it provides a means of describing documents, independent of medium; XML documents can be used for print, the web or any other document medium. This flexibility promotes information system designers to use XML, as they can adopt one set of standards, tools and methods for processing documents, regardless of their various distribution targets. To create web services that can easily be used, XML-based standards have been developed to describe data, services and the communication between these services. In addition, there is a growing suite of tools based on XML can be utilized to establish a data integration framework [21].

- **SOAP (Simple Object Access Protocol)**

SOAP is a XML-based and lightweight protocol used for exchanging structured information in a distributed environment. The SOAP standard is developed by Web Services Activity group of the W3C and uses XML to create an extensible

messaging framework that can exchange data over underlying protocols such as HTTP, FTP, SMTP, and POP3 [29].

- **WSDL (Web Services Description Language)**

WSDL is an XML-style document designed by W3C for describing network services as a set of end-points operating on messages containing either document-oriented or procedure-oriented information. The operations and messages are described abstractly, and then bound to a network protocol and message format to define an end-point. Related end-points are combined into services. WSDL is extensible to allow description of end-points and their messages regardless of what message formats or network protocols are used to communicate [30].

## 4.2 Overview of Bioinformatics Web Services

There is a wide range of bioinformatics web services available worldwide, with those officially provided by the NCBI (National Center for Biotechnology Information) Entrez Programming Utilities [55], the EBI (European Bioinformatics Institute) [56], the the DNA Databank of Japan (DDBJ) [57] and the Kyoto Encyclopedia of Genes and Genomes (KEGG) [58] being the most commonly used in bioinformatics and biomedical fields.

- **National Center for Biotechnology Information**

NCBI is established in 1988 and creates public databases, conducts research in computational biology, develops software tools for analyzing genome data, and disseminates biomedical information. NCBI provides some web service utilities such as EGQuery, EFetch, EInfo, ELink, ESearch, ESpell, and ESummary to access the GenBankR DNA sequence database, the human genome, more than 40 related molecular biology database services, and the scientific literature [55].

- European Bioinformatics Institute

EBI has employed web service technology to enhance and ease the use of the bioinformatics resources it provides. The European Bioinformatics Institute supports SOAP services for both database information retrieval and sequence analysis. For example, WSDbfetch, ChEBI, InterProScan, Emboss, WU-Blast, NCBI Blast, ClustalW, T-Coffee, DaliLite, etc. Currently, EBI provides access to more than 200 databases and to about 150 bioinformatics application [56].

- DNA Databank of Japan

DDBJ has been working in collaboration with EBI and NCBI to catalogue nucleotide sequences, and it provides a number of SOAP services through individual WSDL files: for example, ARSA, Blast, ClustalW, DDBJ, Ensembl, Fasta, GIBEnv, GIBV, Go, GTPS, GetEntry, NCBIGenomeAnnotation, OMIM, RefSeq, and TxSearch (26 services and 214 service operations at the time of writing). Some composed bioinformatics web service workflows, such as DDBJ-UniProt WorkFlow, Blast-ClustalW WorkFlow, Ensembl-PDB WorkFlow, SNP workflow, and OMIM WorkFlow, are also offering in DDBJ to simplify information searching procedures.

- Kyoto Encyclopedia of Genes and Genomes

KEGG is a set of databases including GENES, SSDB, PATHWAY, LIGAND, LinkDB, etc. for genome research and related research areas in molecular and cellular biology [59]. It also offers client application to access all the resources stored in KEGG as a batch processing style through KEGG API (a SOAP web service interface), without any limitations that the interactive CGI programs will suffer.

Beside the above bioinformatics service providers, there are many bioinformatics/biomedical web services available listed in [60].

The beauty of these bioinformatics web services is that researchers/biologists can construct workflows and pipelines combining two or more Web Services to solve complex biological tasks such as protein function prediction, genome annotation, microarray analysis, etc. Users can customize any analytical protocol by combining services available from different locations. Services, thus become building blocks that can be exchanged, allowing flexibility and robustness. Workflow protocols can be created as either simple scripts or using graphical workflow tools such as Taverna [61] or Triana [62, 56].

### 4.3 Classification of Bioinformatics Web Services

Web services technology not only increases the flexibility and interoperability of business software development, but also provides a new way for scientists to retrieve data and utilize tools more efficiently. It may play a role of middle layer between bioinformatics data sources and the user/client program interface. Currently, many bioinformatics data and tools can be accessed/invoked through varied service operations provided by different web services. For instances, WSFasta and WSWUBlast can be used to compare a DNA or protein sequence; InterProScan can perform protein function analysis; MSDFold can analyze protein structure. Microarray identifiers can be annotated by Genecruiser. NCBI Entrez utilities (such as ESearch, Einfo, and Esummary) provide varied data/tools via SOAP for genome research and related research areas in molecular and cellular biology. In addition, PharmGKB web services developed by Stanford University provide genomic, molecular, cellular phenotype, clinical, pharmacokinetic and pharmacogenomic information.

These biomedical/biological web service operations can be classified into three categories (annotating, analysis, and data searching) according to their functionalities. Some examples are given below.

1. Data searching Service Operations

Data searching service operations are mainly used for retrieving varied data and publications for genome research and related research areas in molecular and cellular biology.

**Examples.**

- KEGG : get\_pathways\_by\_genes (pathway)
- NCBI : run\_eFetch, run\_eInfo, run\_eSearch
- PharmGKB : searchGene, searchDrug, searchDisease

2. Analysis Service Operations

Analysis service operations are developed to compare a DNA or protein sequence, analyze protein structure, perform protein function analysis, generate statistical results, etc [56].

**Examples.**

- WSFasta : runFasta (sequence)
- WSWUBlast : runWUBlast (sequence)
- WSDaliLite : runDaliLite (sequence)
- PML : getFrequency

3. Annotating Service Operations

Annotating service operations allow users to annotate their genomic data such as mapping microarray feature identifiers to gene identifiers [63].

**Example.**

- GeneCruiser : annotateProbes, idsToProbes, keywordsToProbes

#### 4.4 QoS for Web Services

With the widespread explosion of Web services, quality of service (QoS) becomes a significant factor in distinguishing the success of service providers. QoS measures determine the service usability and utility, both of which influence the popularity of the

service. In this section, we review major QoS measures used for evaluating the quality of service in the web services area [64, 65, 66] as follows:

- Availability –

Availability is the quality aspect of whether the Web service is present or ready for immediate use. Availability represents the probability that a service is available. Larger values represent that the service is always ready to use while smaller values indicate unpredictability of whether the service will be available at a particular time.

- Accessibility –

Accessibility is the quality aspect of a service that represents the degree it is capable of serving a Web service request. It may be expressed as a probability measure denoting the success rate or chance of a successful service instantiation at a point in time. There could be situations when a Web service is available but not accessible. High accessibility of Web services can be achieved by building highly scalable systems. Scalability refers to the ability to consistently serve the requests despite variations in the volume of requests.

- Integrity –

Integrity is the quality aspect of how the Web service maintains the correctness of the interaction in respect to the source. Proper execution of Web service transactions will provide the correctness of interaction. A transaction refers to a sequence of activities to be treated as a single unit of work. All the activities have to be completed to make the transaction successful. When a transaction does not complete, all the changes made are rolled back.

- Performance –

Performance is the quality aspect of Web service, which is measured in terms of throughput and latency. Higher throughput and lower latency values represent

good performance of a Web service. Throughput represents the number of Web service requests served at a given time period. Latency is the round-trip time between sending a request and receiving the response.

- Reliability –

Reliability is the quality aspect of a Web service that represents the degree of being capable of maintaining the service and service quality. The number of failures per month or year represents a measure of reliability of a Web service. In another sense, reliability refers to the assured and ordered delivery for messages being sent and received by service requestors and service providers.

- Regulatory –

Regulatory is the quality aspect of the Web service in conformance with the rules, the law, compliance with standards, and the established service level agreement. Web services use a lot of standards such as SOAP, UDDI, and WSDL. Strict adherence to correct versions of standards (for example, SOAP version 1.2) by service providers is necessary for proper invocation of Web services by service requestors.

- Security –

Security is the quality aspect of the Web service of providing confidentiality and non-repudiation by authenticating the parties involved, encrypting messages, and providing access control. Security has added importance because Web service invocation occurs over the public Internet. The service provider can have different approaches and levels of providing security depending on the service requestor.

In our approach, we propose enhanced QoS measures by considering temporal dimension to increase the accuracy. The detailed description will be given in section 5.3.2, and the experimental results demonstrate in section 5.7



## CHAPTER 5

### MULTI-LEVEL BIOMEDICAL SERVICE BROKER

Scientists and biologists usually face the problem of selecting appropriate service operations to get more relevant genomic information from hundreds of web services that provide thousands of operations. In order to utilize the existing web services as data sources, we need to model them as available data sources and create rules to relate the existing web services with various concepts in the domain [67]. Furthermore, since there is currently no globally agreed model for describing the structure, type and semantics of data passed between services, it becomes a barrier for them to cooperate with each other. For example, as the body of the SOAP message typically contains weakly or implicitly typed data with many complex legacy and flat-file formats, it increases the difficulty of processing the result at client sides [68]. The above issues give us a motivation to propose a multi-level biomedical service broker system (see Figure 5.2) to solve these problems.

In this chapter, we demonstrate the architecture of our web service broker (see Figure 5.1) and describe the function and interaction among main components. We first introduce the most important component, Web Service Ontology, which maintains description and semantic information of each web service in section 5.1. In section 5.2, we demonstrate how to establish cross-reference among distinct data sources. In section 5.3, we explain how to generate a QoS-ensured query plan for answering user/client program requests. In section 5.4, we show DIE (Dynamic Invocation Engine) that is responsible for dynamically invoking corresponding service operations. In section 5.5, we describe the procedures for constructing the final query results. We also propose a unified programming interface, UBSI (Unified Biomedical Service Interface), which is designed for

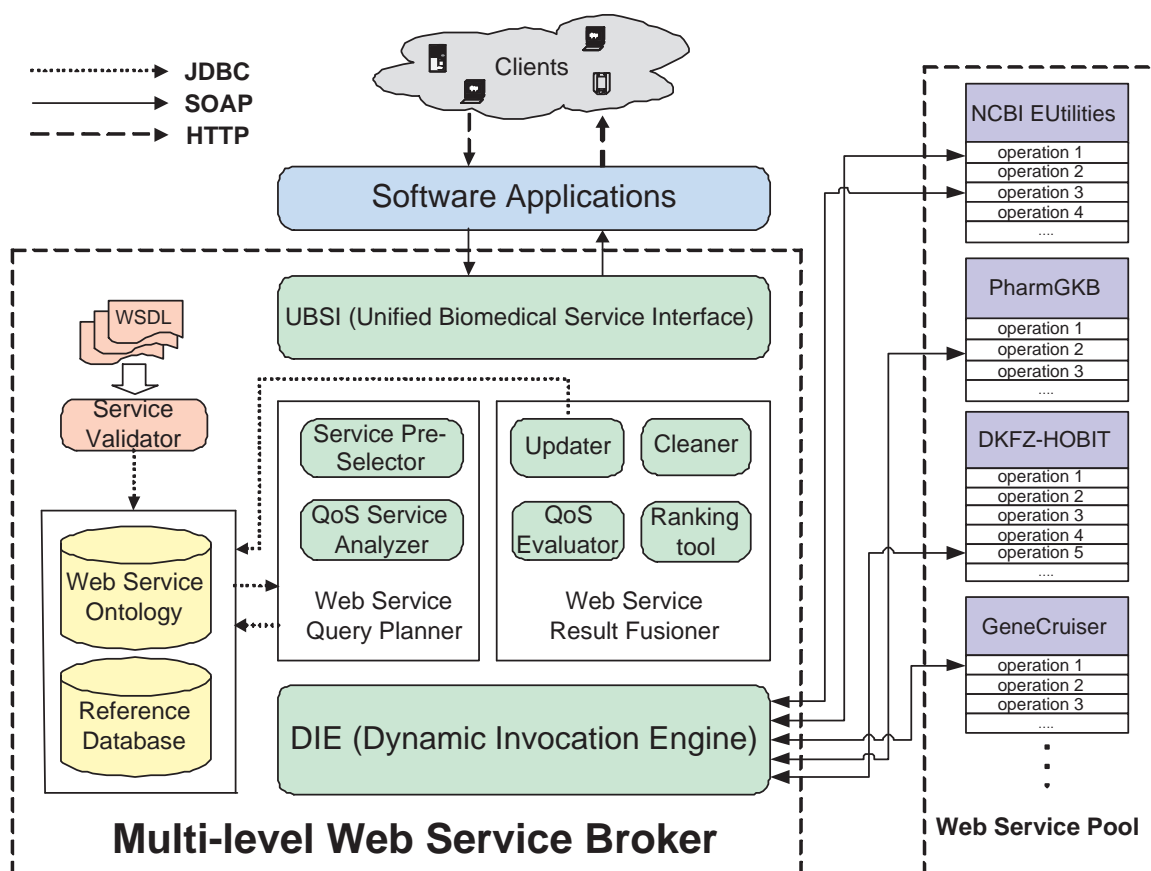


Figure 5.1. BioServiceBroker System Architecture.

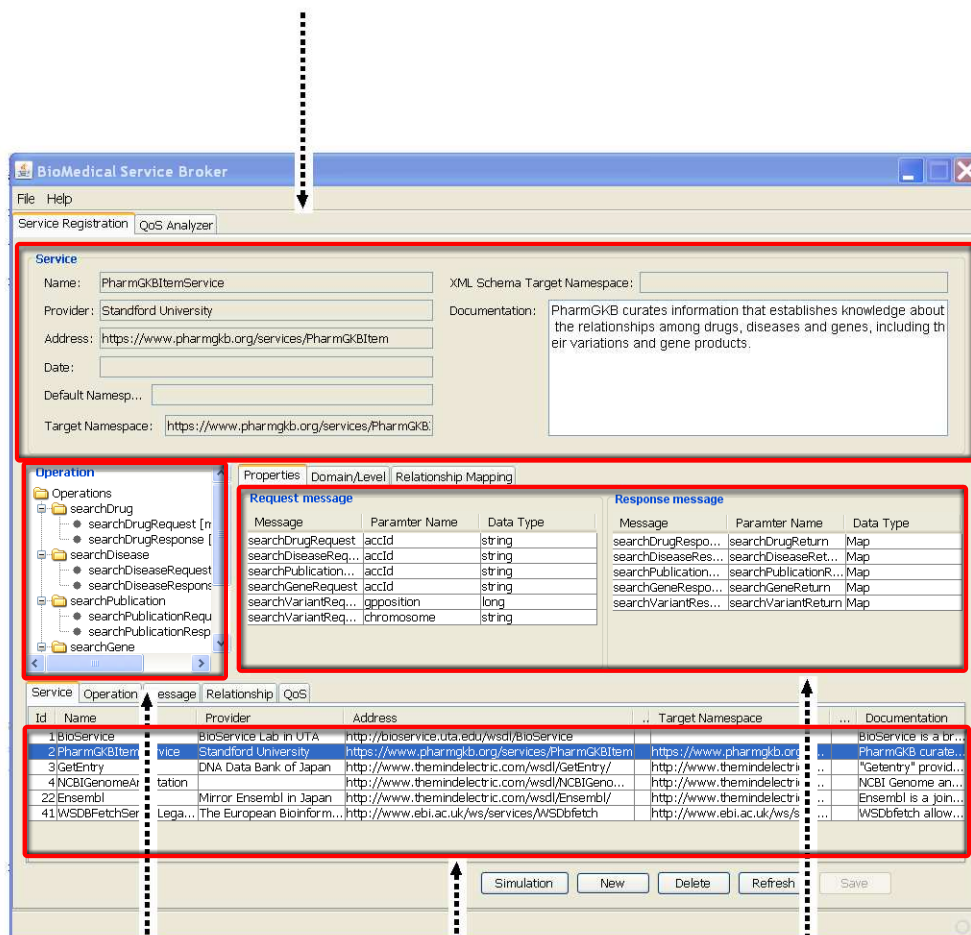
achieving interoperability, reusability, and scalability across a varied base of underlying and changing services in section 5.6. The last section will give the experimental results of the real data collecting from some on-line bioinformatics web services.

## 5.1 Web Service Ontology

"*Ontology*" is the study of the kinds of real things that exist in the world. In computer science, ontology is a representation vocabulary, often specified to some domain or subject matter for representing a conceptualization of terms (or concepts) and relationships among them in a specific application domain [69, 70]. In our approach, we consider each type of service operation as a concept and capture the semantic inter-

<< Web service description section >>

Contains service name, provider, address, default namespace, target namespace, documentation, and so on.



<< Web service operation section >>

Lists all available operations and messages for the corresponding web service.

<< Web service message section >>

Maintains parameter names, input data types, and return data type for each service operation

<< Web Service Ontology browsing section >>

Allows system administrator to browse or update stored data.

Figure 5.2. BioServiceBroker System.

relation between them. The WSO (Web Service Ontology) is the key component in our framework with the purpose of providing service description and semantic information to other components, which not only keeps the WSDL (Web Services Description Language) [30] description for each web service, but also records classification (level/domain), QoS measures (response time, pre-processing time, post-processing time), and binary relationships between service operations. It provides essential functions/data needed by other components in order to complete the query process successfully. In section 5.1.1, we present the schema diagram of the Web Service Ontology and explain each entity and relationship inside. In section 5.1.2, we propose seven service relationships to describe the semantic interrelations between service operations, and the properties of these service relationships will be discussed in section 5.1.3.

### 5.1.1 Service Schema

Without conceptual structure of underlying data, it is not possible to understand the relationship among them in a particular domain [70]. In Figure 5.3, we show the conceptual ER (Entity-Relationship) [20] model of the web service ontology. Entity **SERVICE** stores the general information of each service. The attribute *targetNamespace* denotes the namespace [30] that will be used throughout the service document. For example, the service **PharmGKBItemService** has the *targetNamespace* value <http://www.pharmgkb.org/services/PharmGKBItem>. The attributes *address*, *defaultNamespace*, and *documentation* describe the endpoint, namespaces, and purpose of the services, plus the meanings/constraints on their use respectively [30]. Service operation signatures will be stored in **OPERATION** and **PARAMETER** entities, which keep the name, input/output data structures, type, and transmission protocol of each service operation. **RELATIONSHIP** stores all possible binary relationships (described in section 5.1.2) between two service operations. Also, each service operation will be clas-

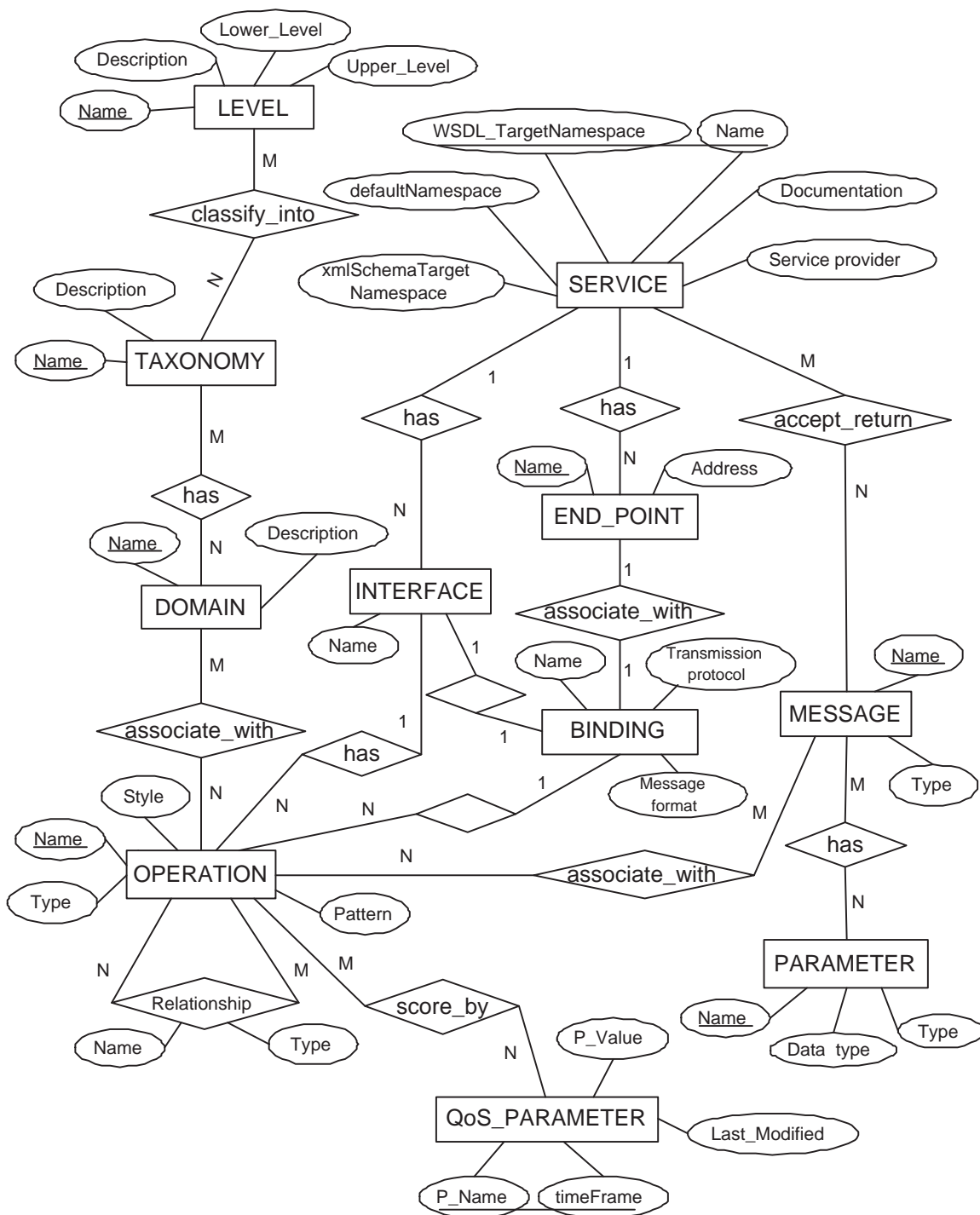


Figure 5.3. BioServiceBroker – Web Service Schema Diagram.

sified into one or more domains and assigned categories with levels stored in DOMAIN, TAXONOMY, and LEVEL entities [3]. Furthermore, QoS\_PARAMETER table will be updated by calculating the scores of QoS parameters based on the definitions of new QoS measures proposed in section 5.3.3.

### 5.1.2 Relationships of Service Operations

We formally define seven binary service relationships that are used for describing semantic interrelations between service operations in this section. The BioServiceBroker system allows service administrator to assign/update relevant service relationships (see Figure 5.4) during service registration procedure.

**Definition 1.** *Equivalent Relationship*

We say that  $R$  is an equivalent relationship between  $op_1$  and  $op_2$  (denoted as  $op_1 \equiv op_2$ ) if  $execute(op_1)$  is identical to  $execute(op_2)$ , where  $op_1$  and  $op_2$  are service operations, and  $execute(op)$  is the data sets returned by service operation "op".

**Definition 2.** *Homogeneous Relationship*

We say that  $R$  is a homogeneous relationship between  $op_1$  and  $op_2$  (denoted as  $op_1 \simeq op_2$ ) if  $execute(op_1)$  is identical or semantically similar to  $execute(op_2)$ <sup>1</sup>, where  $op_1$  and  $op_2$  are service operations, and  $execute(op)$  is the data sets returned by service operation "op".

**Example:**

op1 - NCBI : runNCBIBlast  
 op2 - WU-Blast : runWUBlast  
 $\implies op_1 \simeq op_2$

---

<sup>1</sup> $op_1 \equiv op_2 \Rightarrow op_1 \simeq op_2$

<< Web service relationship tree >>  
Shows the defined service relationships.

The screenshot shows the BioMedical Service Broker interface. The top section displays service registration details for 'PharmGKBItemService'. Below this, there are tabs for 'Properties', 'Domain/Level', and 'Relationship Mapping'. The 'Relationship Mapping' tab is active, showing a tree view of relationships (Equivalent, Homogeneous, Containing, Overlapping, Derived, Successor (direct), Successor (indirect)) and a list of UBSI service operations. A table at the bottom lists the identified relationships between service operations.

Service Operation	UBSI Relationship	UBSI Service Operation
36.searchGene	Containing	1.getGeneInfo
36.searchGene	Containing	4.getRelatedGeneByGene
36.searchGene	Containing	14.getRelatedPathwayByGene
36.searchGene	Containing	21.getRelatedDiseaseByGene
36.searchGene	Containing	26.getRelatedDrugByGene
58.getXML_DDB.Entry	Containing	1.getGeneInfo
58.getXML_DDB.Entry	Containing	3.getGeneSequence
128.getGeneInfo	Containing	1.getGeneInfo
150.getGeneInfo	Containing	3.getGeneSequence

<< Web service relationship table >>  
Displays all identified relationships between service operations.

<< UBSI service operation list >>  
Shows the UBSI service operations that have specified service relationship with selected operation.

Figure 5.4. BioServiceBroker – Service Relationships.

**Definition 3.** *Containing Relationship*

We say that  $R$  is a containing relationship between  $op_1$  and  $op_2$  (denoted as  $op_1 \odot op_2$ ), if  $\forall d \in execute(op_1)$ , then  $d \in execute(op_2)$ , where  $op_1$  and  $op_2$  are service operations,  $execute(op)$  is the data sets returned by service operation "op", and  $d$  is a part of  $execute(op)$ .

**Example:**

op1 - KEGG : get\_pathways\_by\_genes

op2 - PharmGKB : searchGene

$\implies op_1 \odot op_2$

**Definition 4.** *Overlapping Relationship*

We say that  $R$  is an overlapping relationship between  $op_1$  and  $op_2$  (denoted as  $op_1 \circ op_2$ ) if  $execute(op_1) \cap execute(op_2) \neq \phi$  and  $execute(op_1) \neq execute(op_2)$ <sup>2</sup>, where  $op_1$  and  $op_2$  are service operations, and  $execute(op)$  is the data sets returned by service operation "op".

**Example:**

op1 - NCBI : getGeneInfo

op2 - Ensembl : getGeneInfo

$\implies op_1 \circ op_2$

**Definition 5.** *Derived Relationship*

We say that  $R$  is a derived relationship between  $op_1$  and  $op_2$  (denoted as  $op_1 \rightarrow op_2$ ) if  $execute(op_2)$  can be derived from  $execute(op_1)$ , where  $op_1$  and  $op_2$  are service operations, and  $execute(op)$  is the data sets returned by service operation "op".

**Example:**


---

<sup>2</sup> $op_1 \odot op_2 \Rightarrow op_1 \circ op_2$



op1 - KEGG : get\_genes\_by\_organism

op2 - KEGG : get\_number\_of\_genes\_by\_organism

$\implies op_1 \twoheadrightarrow op_2$

**Definition 6.** *Successor<sub>direct</sub> Relationship*

We say that  $R$  is a *Successor<sub>direct</sub> Relationship* between  $op_1$  and  $op_2$  (denoted as  $op_1 \rightarrow op_2$ ) if  $execute(op_2)$  can be derived from  $execute(op_1)$ , where  $op_1$  and  $op_2$  are service operations, and  $execute(op)$  is the data sets returned by service operation "op".

**Definition 7.** *Successor<sub>indirect</sub> Relationship*

We say that  $R$  is a *Successor<sub>indirect</sub> Relationship* between  $op_1$  and  $op_n$  (denoted as  $op_1 \rightsquigarrow op_n$ ) if  $\exists$  a series of *Successor<sub>direct</sub> Relationships*  $\{(op_1 \rightarrow op_2), (op_2 \rightarrow op_3) \dots, (op_{n-1} \rightarrow op_n)\}$  such that  $execute(op_2)$  can be derived from  $execute(op_1)$ , where  $op_1$  and  $op_2$  are service operations, and  $execute(op)$  is the data sets returned by service operation "op".

### 5.1.3 Properties of Service Relationships

In mathematics, a binary relation is an association of elements of one set/class with elements of another (perhaps the same) set/class. It has been used widely in conceptual modeling in mathematics and computer science. We employ the mathematical definitions of transitive, symmetric, asymmetric and reflexive properties to describe the characteristics of each service relationship. Table 5.1 demonstrates these properties applied to our proposed relationships. By utilizing these properties, we can precisely select appropriate service operations, and establish possible instances of service workflows.

Table 5.1. Properties of Service Relationships.

Relationship	Transitive	Symmetric	Asymmetric	Reflexive
<i>Equivalent</i>	+	+	-	+
<i>Homogeneous</i>	+	+	-	+
<i>Containing</i>	+	-	+	n/a
<i>Overlapping</i>	-	+	-	n/a
<i>Derived</i>	+	-	+	n/a
<i>Successor<sub>direct</sub></i>	-	-	+	-
<i>Successor<sub>indirect</sub></i>	+	-	+	-

## 5.2 Reference Database

RD (Reference database) is a cross-reference depository, which keeps identifier mapping information among diverse databases such as Gene ontology, GeneBank, PDB, KEGG, PharmGKB, and so on. By referring to this component, our system can establish solid connections for the same biological entity among services. It allows our system to accept user queries with varied identifiers, and makes it more flexible and interoperable. Similar approaches are discussed in [19].

## 5.3 Web Service Query Planner

WSQP (Web Service Query Planner) is responsible for generating web service query plans based on the domain, level/classification, and other properties provided by user queries. It can be divided into two sub-components: *Service Pre-Selector* and *QoS Service Analyzer* (see Figure 5.1), which will be discussed in the following paragraphs.

### 5.3.1 Service Pre-Selector

After a user invokes a service operation through UBSI (discuss in section 5.6), the Service Pre-Selector will check the domain and level information of invoked service operation, then looks up web service ontology to find related service operations as can-

didates to be invoked. For example, when a user requests a UBSI service operation – ”getGeneInfo”, the Service Pre-Selector module starts a process to list candidates who have (equivalent, homogeneous, containing, overlapping, etc.) relationships with the ”getGeneInfo” operation in the Web Service Ontology. The candidate list will be passed to the QoS Service Analyzer for advanced analysis.

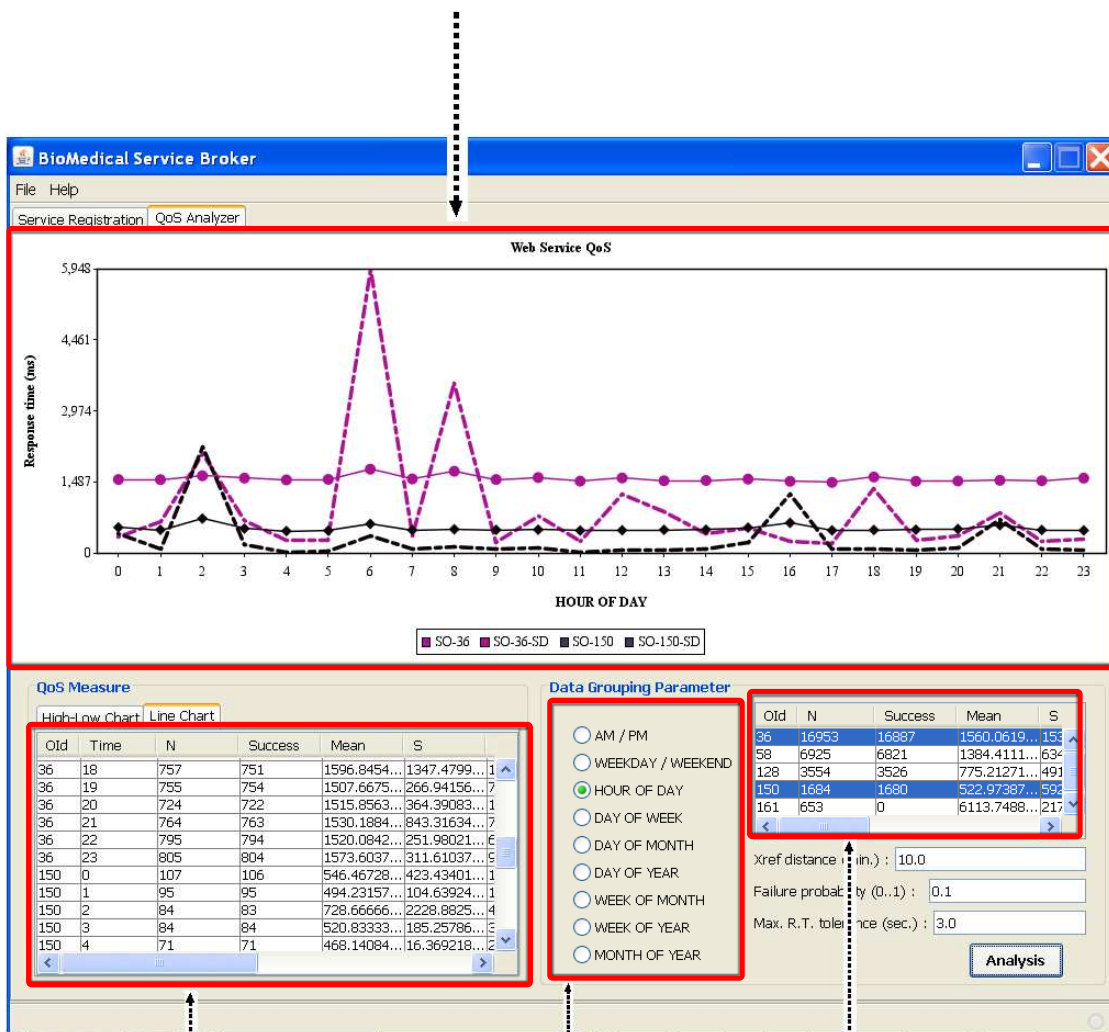
### 5.3.2 QoS Service Analyzer

QoS issues in web services have been discussed in previous work [64, 65, 66], and each of them proposed some QoS parameters such as average response time ( $\overline{RT} = \frac{\sum_{k=1}^n RT_k}{n}$ ), availability, reliability, security, throughput, and cost from different perspectives. The original definitions of these parameters do not consider possible patterns hidden behind the temporal dimension, which may lead us to conclude a fault or bias decision. We propose new QoS measures in section 5.3.3. The following two cases demonstrate potential problems with current measures and the needs for our enhanced QoS parameters. Figure 5.7 and 5.9 display the results of a simulation to measure response time over three time frames: 1, 2, 3 for two distinct service operations A and B but supplying homogeneous data sets.

**Case 1:** *In Figure 5.7,  $\overline{RT}$  of **service operation B** (2.19 sec) is smaller than that of **service operation A** (2.42 sec). If we only consider  $\overline{RT}$  as the most important QoS parameter for selecting a service operation, **service operation B** will be selected. However, by controlling time variable, **service operation B** should not be selected within time frame 1 or 3 since it has longer average response time during these periods (time frame 1 [(A)1.12 sec/(B)1.31 sec], time frame 3 [(A)1.10 sec/ (B)1.42 sec], see Table 5.2). Also, we get similar result in terms of Stability showing in Table 5.3. Without considering temporal dimension, this information may lead us to conclude that B is better*

<< Web Service QoS chart >>

*This figure shows the Line chart of service operations (36 and 150). Solid line represents the mean of response time, and dash line stands for standard deviation of response time.*



<< Web service operation QoS table >>  
*Shows the timestamp, Mean of response-time, Standard Deviation,*

<< Web service operation overall QoS table >>  
*Shows the overall QoS data of each stored operation.*

<< Web service data grouping constraints >>  
*Allows system administrator to analyze data based on different grouping constraints.*

Figure 5.5. BioServiceBroker – QoS Service Analyzer (Line Chart).

<< Web Service QoS chart >>

*This figure displays the High-Low chart of service operation 150. It shows the mean, standard deviation, and historical high and low of response time.*



<< Web Service data selected constraints >>

*Allows system administrator to analyze data based on Xref distance, Failure probability and Max. tolerance Response time.*

Figure 5.6. BioServiceBroker – QoS Service Analyzer (High-Low Chart).

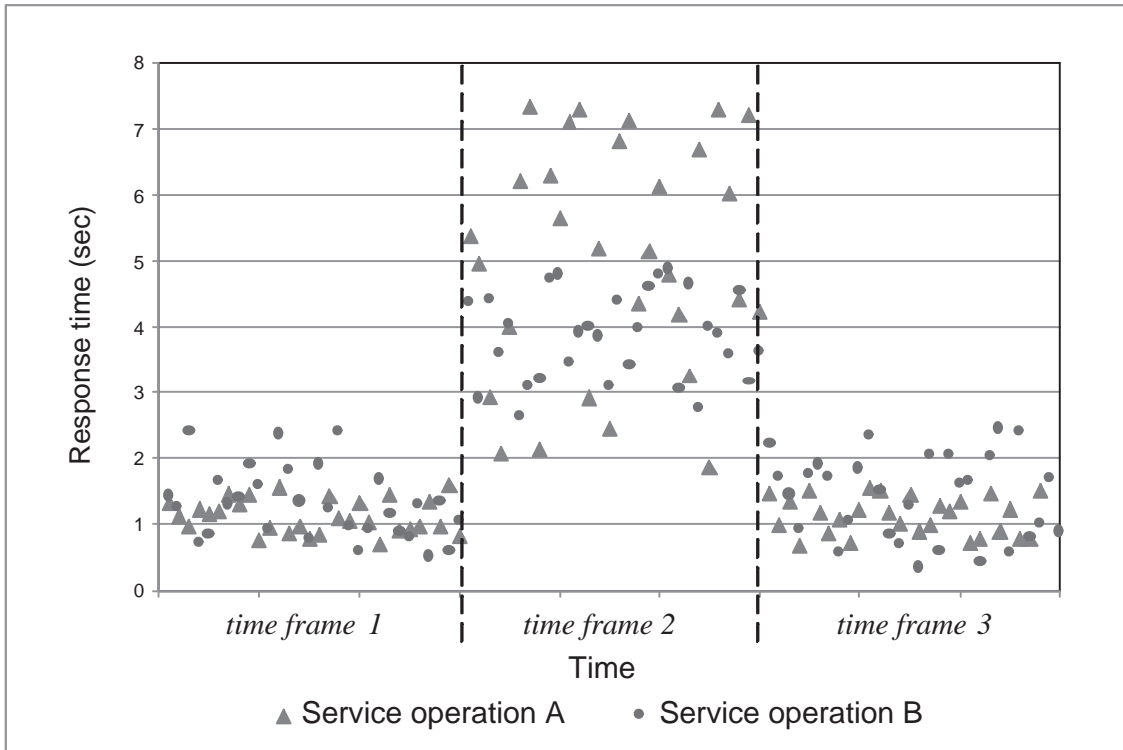


Figure 5.7. Response Time in Case 1.

than A both in terms of average response time and stability.

**Case 2:** In Figure 5.8 service operation B is generally a more stable system over all time frames. By controlling time variable and adjusting the weight of  $\overline{RT}_{tf_i}$ , service operation A may have chance to be selected even though service operation B shows better than service operation A both in overall average response time and stability.

**Case 3:** Furthermore, we may also make fault selections based on other original QoS parameters as service operations may not be accessible or has extremely low  $AVA_{tf_i}$  pattern within certain time frames due to system regular maintenance or other circumstances

Table 5.2. Average Response Time ( $\overline{RT}_{tf_i}$ ) in Case 1

Service operation $\overline{RT}_{tf_i}$ (sec)	Service Operation A	Service Operation B	Decision
Time frame 1	1.12	1.31	A
Time frame 2	5.05	3.85	B
Time frame 3	1.10	1.42	A
Total	2.42	2.19	B

Table 5.3. Stability ( $STA_{tf_i}$ ) in Case 1

Service operation $STA_{tf_i}$	Service Operation A	Service Operation B	Decision
Time frame 1	0.25	0.54	A
Time frame 2	1.76	0.67	B
Time frame 3	0.29	0.64	A
Total	2.13	1.33	B

Table 5.4. Average Response Time ( $\overline{RT}_{tf_i}$ ) in Case 2

Service operation $\overline{RT}_{tf_i}$ (sec)	Service Operation A	Service Operation B	Decision
Time frame 1	1.25	1.67	A
Time frame 2	3.79	3.01	B
Time frame 3	1.45	1.71	A
Total	2.17	2.13	B



Figure 5.8. Response Time in Case 2.

Table 5.5. Stability ( $STA_{tf_i}$ ) in Case 2

$STA_{tf_i}$ \ Service operation	Service Operation A	Service Operation B	Decision
Time frame 1	0.79	0.47	B
Time frame 2	0.81	0.50	B
Time frame 3	0.80	0.50	B
Total	1.40	0.79	B



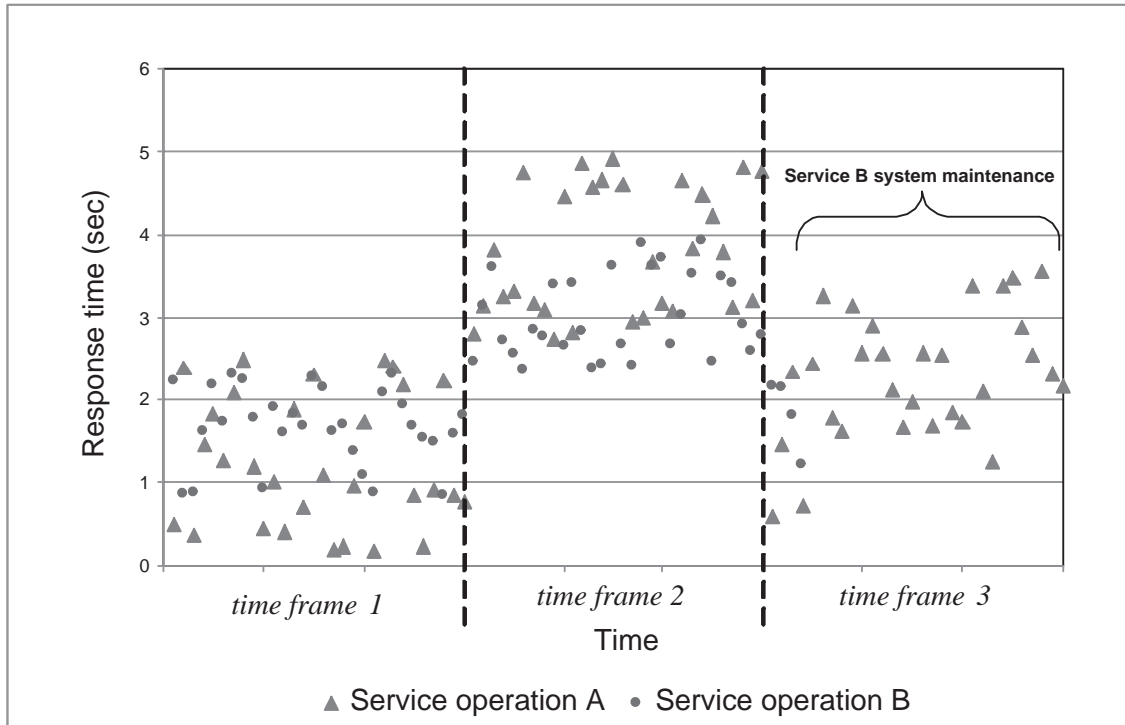


Figure 5.9. Response Time in Case 3.

(see Figure 5.9).

QoS service analyzer is in charge to deal with above issues in order to choose a more reliable and desirable service in a service pool.

### 5.3.3 Formal Definitions of QoS Parameters

In this section, we apply *temporal concept* on QoS parameters considered in QoS service analyzer, and give the formal definitions of them below. By adding *temporal concept*, our multi-level service broker can more precisely select and invoke corresponding external web service operations based on query time with weights of enhanced QoS parameters given by service requesters.

**Definition 1.** Time frame( $tf_i$ ).

We say that  $(tf_i)$  is a time frame of time axis  $T$ , if  $\forall tf_i \in T$  and  $\forall tf_j \in T$

$$tf_i \cap tf_j = \emptyset \quad (5.1)$$

where  $i \neq j$ , and

$$T = \bigcup_{i=1}^n tf_i, n \in \mathbf{N} \quad (5.2)$$

**Definition 2.** Average response time at time frame  $tf_i$  ( $\overline{RT}_{tf_i}$ ).

$$\overline{RT}_{tf_i} = \frac{\sum_{k=1}^{n_{tf_i}} RT_k}{n_{tf_i}} \quad (5.3)$$

where  $n_{tf_i}$  is the total number of invocations at  $tf_i$  and  $RT_k$  is the response time of  $k$ -th invocation at  $tf_i$

**Definition 3.** Stability at time frame  $tf_i$  ( $STA_{tf_i}$ ). It is measured by the dispersion of response time ( $RT_k$ ) around the ( $\overline{RT}_{tf_i}$ ).

$$STA_{tf_i} = \left( \frac{\sum_{k=1}^{n_{tf_i}} (RT_k - \overline{RT}_{tf_i})^2}{n_{tf_i} - 1} \right)^{\frac{1}{2}} \quad (5.4)$$

where  $n_{tf_i}$  is the total number of invocations at  $tf_i$  and  $RT_k$  is the response time of  $k$ -th invocation at  $tf_i$

**Definition 4.** Availability at time frame  $tf_i$  ( $AVA_{tf_i}$ ).

Availability is the probability (or percentage of time) that a service operation can be successfully invoked.

$$AVA_{tf_i} = \frac{n_{successtf_i}}{n_{tf_i}} \quad (5.5)$$

where  $n_{tf_i}$  is the total number of invocations at  $tf_i$  and  $n_{successtf_i}$  is the number of successful invocations at  $tf_i$

**Definition 5.** Completeness (*COMP*).

Completeness is the average of completeness percentages. It is fairly difficult to measure completeness automatically since it is subjective. We allow users to evaluate the degree of completeness (excellent, good, satisfactory, fair, unsatisfactory, etc.) when the query result shows on graphical user interface.

$$COMP = \frac{\sum_{i=1}^n \frac{rating_i}{rating_{max}}}{n} \quad (5.6)$$

where

$n$  is the total number of evaluations

$rating_i$  is rating value of  $i$ -th evaluation

$rating_{max}$  is the maximum rating value that can be given for each evaluation

**Definition 6.** Authority (*AUTH*).

Authority is a boolean parameter (0 or 1) assigned at service registration time, which indicates if the service operation is provided by official web site.

where

$AUTH=0$ , provided by official source

$AUTH=1$ , not provided by official source

**Definition 7.** Score at time frame  $tf_i$  ( $SCORE_{tf_i}$ ). Smaller score means better service.

$$SCORE_{tf_i} = \sum_{j=1}^m (\omega_j \times P_{jtf_i}) \quad (5.7)$$

where

$m$  is the total number of QoS parameters

$$\sum_{j=1}^m \omega_j = 1$$

$P_{jtf_i}$  is the value of parameter  $j$

$\omega_j$  is the weight of  $P_{jtf_i}$  assigned by user query

### EXAMPLE

We consider four QoS parameters  $j=\{1..4\}$  here, i.e.,  $(\overline{RT}_{tf_i}, STA_{tf_i}, AVA_{tf_i}, COMP)$

where

$$j=1, P_{1tf_i} = \text{Min}[\frac{\overline{RT}_{tf_i}}{RT_{max}}, 1], RT_{max} \text{ is max. tolerance time.}$$

$$j=2, P_{2tf_i} = \text{Min}[\frac{STA_{tf_i}}{RT_{max}}, 1], RT_{max} \text{ is max. tolerance time.}$$

$$j=3, P_{3tf_i} = (1-AVA_{tf_i})$$

$$j=4, P_{4tf_i} = (1-COMP)$$

Table 5.6 demonstrates the possible decisions of case 1 at different time frames by considering two more QoS parameters and assigned weight values on them. It shows the power of enhanced QoS parameters that allow service requesters to control more querying factors that may lead to varied decisions (Smaller score means better quality of service).

Figure 5.5 shows the comparison of two service operation (36 and 150) in the QoS Line chart based on "HOUR\_OF\_DAY" grouping constraint with other selected constraints (solid line represents the mean of response time, and dashed line stands for the standard deviation of response time), and the BioServiceBroker system also provides detailed statistical information in the QoS table. Additionally, the QoS High-Low chart

Table 5.6. Decision Table.

Parameter Time; Weight	Service A operation					Service B operation					Decision
	$P_{1tfi}$	$P_{2tfi}$	$P_{3tfi}$	$P_{4tfi}$	Total Score	$P_{1tfi}$	$P_{2tfi}$	$P_{3tfi}$	$P_{4tfi}$	Total Score	
T1	0.112	0.025	0.17	0.05	0.091	0.131	0.054	0.28	0.08	0.128	<b>A</b>
Weight	0.45	0.20	0.15	0.20		0.45	0.20	0.15	0.20		
Score	0.05	0.005	0.026	0.01		0.059	0.011	0.042	0.016		
T2	0.505	0.176	0.01	0.05	0.381	0.385	0.067	0.25	0.08	0.292	<b>B</b>
Weight	0.70	0.10	0.00	0.20		0.70	0.10	0.00	0.20		
Score	0.354	0.018	0	0.01		0.27	0.007	0	0.016		
T3	0.11	0.029	0.25	0.05	0.216	0.142	0.064	0.17	0.08	0.158	<b>B</b>
Weight	0.1	0	0.8	0.1		0.1	0	0.8	0.1		
Score	0.011	0	0.2	0.005		0.014	0	0.136	0.008		
T	0.242	0.213	0.19	0.05	0.069	0.219	0.133	0.06	0.08	0.094	<b>A</b>
Weight	0.1	0	0	0.9		0.1	0	0	0.9		
Score	0.024	0	0	0.045		0.022	0	0	0.072		

$$RT_{\max} = 10$$

shown in Figure 5.6 can provide the historical high and historical low information for a certain service operation.

#### 5.4 Dynamic Invocation Engine

DIE (Dynamic Invocation Engine) is responsible for composing signatures of remote service operations and invoking them based on the query plan generated by the WSQP module at runtime. Contrastingly, static invocations require that the signatures of service operations are provided while developing the application in order to compile the program, which will limit the flexibility and extensibility of the application [71]. Through DIE component, we can establish more flexible and distributed query environment without even changing source program.

#### 5.5 Web Service Result Fusioner

The majority of service providers do not classify their service operations based on the granularity of return data. For example, a user/client program is only interested

```

<?xml version="1.0" encoding="UTF-8" ?>
- <EMBL_Services xmlns:ebi="http://www.ebi.ac.uk/embl/schema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://www.ebi.ac.uk/embl/schema/EMBL_Services_V1.1.xsd">
- <entry accession="AC002457" version="13" dataClass="STD" taxonomicDivision="HUM" created="1997-
  08-26" lastUpdated="2007-10-16" releaseCreated="52" releaseLastUpdated="93">
  <description>Homo sapiens BAC clone CTB-60P12 from 7, complete sequence.</description>
  <keyword>HTG</keyword>
  + <reference>
- <reference>
  - <citation id="2" type="unpublished">
    <title>The sequence of Homo sapiens BAC clone CTB-60P12</title>
    <author>Beck C.</author>
    <author>Gibson A.</author>
    <author>Smith A.</author>
    </citation>
    <citationLocation begin="1" end="177380" />
    </reference>
  + <reference>
  + <reference>
  + <reference>
  + <reference>
  + <reference>
  + <reference>
    <dbreference db="EMBL-TPA" primary="BL000002" />
    <dbreference db="EPD" primary="EP35012" secondary="HS_ABCB1" />
    <dbreference db="GDB" primary="5241553" />
    <comment>On Mar 23, 2002 this sequence version replaced gi:3947433. ----- Genome
      Center Center: Washington University Genome Sequencing Center Center code: WUGSC Web
      site: http://genome.wustl.edu Contact: sapiens@watson.wustl.edu ----- Summary
      Statistics Center project name: H_RG060P12 ----- NOTICE: This sequence may not
      represent the entire insert of this clone. It may be shorter because we only sequence
      overlapping clone sections once, or longer because we provide a small overlap between
      neighboring data submissions. This sequence was finished as follows unless otherwise noted:
      all regions were double stranded, sequenced with an alternate chemistry, or covered by high
      quality data (i.e., phred quality >= 30); an attempt was made to resolve all sequencing
    </comment>
  + <feature name="repeat_region">
  + <feature name="repeat_region">
  + <feature name="repeat_region">
  - <feature name="repeat_region">
    - <qualifier name="rpt_family">
      <value>AT_rich</value>
    </qualifier>
    - <location type="single" complement="false">
      - <locationElement type="range" accession="AC002457" version="2" complement="false">
        <basePosition type="simple">175998</basePosition>
        <basePosition type="simple">176035</basePosition>
      </locationElement>
    </location>
  </feature>
  + <feature name="repeat_region">
  + <feature name="repeat_region">
  <sequence type="genomic DNA" length="177380" topology="linear"
  version="2">
    |aggcatttagtcaacattggttaaactgaccaacatttataataatctaaagtctctcctcctcagggaagaaacataaaa
    |gtttaacaattcttaggcaaaaaaagaatggctacattcaaaatatttggtataactftagaagaaagtaaacafftaaaaatg
    |gtattatttttaacttgaaatftctgagccatatagtccaaaagtgaagtgattttctctaaccaatctafaagttctgaggt
    |gcatcggcagcaagttcagccaagctgctctctttgaggctattctactcatcagttattacatctgagaagaatgatg
    |gaggtaaaagttaaacaggaaacaafftctgctcacatatttgacttacagacacctagcctctc caatacaataagctaca-
    |
    |
    |
  </entry>
</EMBL_Services>

```

*Gene sequence information from  
EMBL dbfetch service operation*

Figure 5.10. EMBL dbfetch service operation Result (partial).

in a certain gene sequence, but an available related service operation is "dbfetch" that returns not only gene sequence but also a bunch of terms or sections, which are not really requested (see Figure 5.10). These additional information not only occupies bandwidth and increases network traffic, but also adding post-processing time at the client side. By utilizing our system, web service result fusioner module is responsible to filter out the unnecessary data sections, and construct a more concise and exchangeable format at broker side before sending back to requester.

There are four sub-components inside the WSRF (Web Service Result Fusioner): cleaner, updater, QoS evaluator, and ranking tool. The WSRF gathers service results from the DIE component, and passes them to the QoS evaluator and updater components for calculating and updating QoS statistical information inside the web service ontology. Cleaner transforms and filters results gathered from different web services, and ranking tool compares the same biological entity based on varied conditions (completeness, versions, etc.) and checks consistency/integrity among them. The concise result will be constructed and return to client side through SOAP interface.

### 5.5.1 Single Result Processing

Due to the complexity of return data types (flat file, array, string, HashMap, FASTA, XML, and so on) for each service operation, we employ cleaner sub-component to transform and filter result. The first step of single result processing is to parse return data and transform them into XML format. Only the sections that match to the corresponding invoked UBSI service operation are extracted and inserted into DOM (Document Object Model). The second step is to add the QoS statistical information (pre-processing time, response time, post-processing time) collected from Service Pre-Selector, DIE component, and cleaner sub-component respectively into DOM. The final procedure is to transform the DOM document into result format specified by client request. For exam-

ple, we want to gather the related pathways information for a certain gene by invoking UBSI `getRelatedPathwayByGene` operation, and the WSQP (Web Service Query Planner) selects PharmGKB service operation - "searchGene" as the best candidate to fulfill this task. After invocation process, system obtains the result as shown in Figure 5.11. The WSRF (Web Service Result Fusioner) starts to parse and transform it into XML format, and the pathway section is extracted. The next step is to evaluate QoS values for this invocation and update record inside the WSO component. A XML service report that contains QoS information with query result is generated and send back to the client side (see Figure 5.12).

### 5.5.2 Batch Results Processing

With the aim of achieving high throughput and reduce the number of invocations at client sides. Our broker system can also accept batch invocation requests. In contrast to single invocation processing discussed above, batch invocation processing is the execution of a series of remote service operations/jobs at the same time. As the multiple invocation results come out, the WSRF component needs to handle batch results fusion. Figure 5.13 shows a batch query form that includes three UBSI service operations request, and specifies the QoS measure weights such as  $\overline{RT}_{t_i}$  (Response time) and  $STA_{t_i}$  (Stability). This QoS measure weight information will be used for evaluating scores for each service operation candidates according to current UBSI services request timestamp. The query result of a batch query is shown in Figure 5.14.

## 5.6 UBSI (Unified Biomedical Service Interface)

Because there is currently no agreed standard to define the service operation in terms of names and data types of input/output parameters, a client side(programmer, application, etc.) usually needs additional efforts (such as transformation and parsing



Table 5.7. Unified Biomedical Service Operation (partial).

<b>GENE</b>	<b>PROTEIN</b>	<b>DISEASE</b>
<i>getGeneInfo</i>	<i>getProteinInfo</i>	<i>getRelatedDiseaseByGene</i>
<i>getSimilarGene</i>	<i>getSimilarProtein</i>	<i>getRelatedDiseaseByProtein</i>
<i>getGeneSequence</i>	<i>getProteinSequence</i>	<i>getRelatedDiseaseByDrug</i>
<i>getRelatedGeneByGene</i>		<i>getRelatedDiseaseBySymptom</i>
<i>getRelatedGeneByProtein</i>		<i>getRelatedDiseaseByPathway</i>
<i>getRelatedGeneByDisease</i>	<b>PATHWAY</b>	
<i>getRelatedGeneByDrug</i>	<i>getRelatedPathwaysByGenes</i>	
<i>getRelatedGeneByPathway</i>	<i>getRelatedPathwaysByDrugs</i>	<b>DRUG</b>
<i>getGeneListByOrganism</i>	<i>getRelatedReactionsByPathway</i>	<i>getRelatedDrugByGene</i>
<i>getBasePairRatio</i>		<i>getRelatedDrugByProtein</i>
	<b>REFERENCE</b>	<i>getRelatedDrugByDisease</i>
	<i>getTaxonomyIdByName</i>	<i>getRelatedDrugByDrug</i>
	<i>getTaxonomyNameById</i>	<i>getRelatedDrugBySymptom</i>
	<i>getRelatedPublication</i>	<i>getRelatedDrugByPathway</i>
	<i>getXReferenceAccessionID</i>	<i>getDrugsByComposition</i>

procedures) before getting query results. Especially when some web service sources do not provide the detailed specification for their customized complex data types, it becomes a barrier for client to utilize their services.

The UBSI (Unified Biomedical Service Interface) is designed for achieving interoperability, reusability, and scalability across a varied base of underlying and changing services. A client side can remotely invoke a service operation through UBSI, and is not required to deal with complex/inconsistent interfaces of each web service. The invocation procedure generally takes a form such as *invoke(serviceInfo, operationName, QoSWeights, inputParameters, outputParameters)*, where *serviceInfo* defines the name of our QoS broker service ("BioServiceBroker"), its endpoint, etc.; *OperationName* is the unified biomedical service operation listed in Table 5.7; *QoSWeights* contains the weights of each QoS parameter assigned by service requesters. *InputParameters* and *outputParameters* refer to the query parameters given by service requester and result of invocation respectively. Table 5.7 shows partial set of UBSI operations.

## 5.7 Experimental Results

In this section, we will give an example to show the power of our QoS Service Analyzer since it is the most important component that determines the best service operation in service pool. We also propose solutions for two events that may lead to the bias decisions.

### 5.7.1 Service Analysis

In order to demonstrate how the Service Analyzer works, we assume that there is a service request on 10AM Friday 2008/04/25, and the Service Pre-Selector sub-component selects operation 36 and 128 as the candidates to gather the result. Based on above information, it will involve the following decision making processes.

Figure 5.15 compares the average response time and standard deviation of response time between service operations 36 and 128 with HOUR\_OF\_DAY grouping constraint. We can see that operation 36 has longer average response time than operation 128 does at all time frames, but its service is more stable because of lower standard deviation. In this case, if service requester assigns "1" to the weight of  $STA_{t_f}$ , the decision will be 100% dominated by  $STA_{t_f}$  and service operation 36 will be selected. In short, service requester's QoS weights can determine the service operation selected by the Service Analyzer.

However, it is probably not enough to provide the optimized solution. As shown in Figure 5.16, if we change the data grouping constraint to "DAY\_OF\_WEEK", service operation 128 should be picked regardless of the QoS weight values specified by requesters.

By considering QoS weights and data grouping granularity in temporal dimension, our approach gives the flexibility to requesters by allowing them to control more querying factors in order to improve their client system performance.

### 5.7.2 Events Handling

As we have discussed in section 5.7.1 and section 5.3.3, all QoS measures are calculated in accordance with stored data and grouping constraints. Therefore, we need to ensure that data collected during each invocation can fairly represent the performance of remote registered web service. Unfortunately, we have identified two types of events that may affect the eligibility of QoS measures.

- **Failures occur in local broker system**

Due to the failures (network disconnected, hardware malfunction, etc.) occurring in local side (BioServiceBroker), it will immediately and dramatically increase response time for all registered service operations that have been invoked during this period. However, these records should not be counted for evaluating QoS measures since they do not reflect the actual performance of invoked remote service operations during this period. We propose three control variables (Max. tolerance response time, Xref distance, and Failure probability) to filter out these data.

1. Max. tolerance response time –

Stands for the maximum response time that can be accepted by system (default value: 5 seconds). If the actual response time for a service operation is larger than Max. tolerance response time, we consider this invocation is unsuccess.

2. Xref distance –

Specifies the data in which time interval should be considered as reference (default value: 10 minutes).

3. Failure probability –

Failure probability =  $1 - \frac{\text{Number of Success Invocations}}{\text{Number of Invocations}}$

For example, a network connectivity problem arises in BioServiceBroker side on 10:00 Friday 2008/04/25, and all records collected during this period show pretty high response time.

If the Max. tolerance response time equals to 5 seconds, then each record whose response time is higher than 5 seconds will be considered as a unconfirmed record. Since this unconfirmed record is recorded on 10:00 Friday 2008/04/25 and the Xref distance is 20 minutes, all stored records (excluding data collected from the same web service site as this unconfirmed record) between 9:40 Friday 2008/04/25 and 10:20 Friday 2008/04/25 will be used to verify whether this unconfirmed record is valid or not.

If there are total 50 invocation records between 9:40 Friday 2008/04/25 and 10:20 Friday 2008/04/25 selected as references and 47 of them are unsuccessful (Actual Failure Probability = 0.94), the QoS Service Analyzer will discard this unconfirmed record because actual failure probability is higher than default value (0.90), which means that it may not be the problem of this particular remote invoked web service site.

- **Slow network speed in local broker system**

When our local site has high volume network traffic, it is going to increase the response time for all invocations. In this case, we should also ignore these records during QoS service analysis procedure. Instead of using Max. tolerance response time as a threshold, we observe if the response time of each record over three standard deviations of average response time during the period [72].

If the record whose response time over three standard deviations of its historical average response time during this period, then this record considered as a unconfirmed record. Similar to above discussion, all stored records between 9:40 Friday

2008/04/25 and 10:20 Friday 2008/04/25 will be used to verify whether this unconfirmed record is valid or not. If the actual failure probability is higher than default one, we will discard this record.

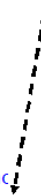
```

key list:
  geneRelatedPhenotypeDatasets
  geneRelatedDrugs
  geneSymbol
  geneName
  searchTerm
  geneAlternateSymbols
  geneRelatedPathways
  geneRelatedDiseases
  geneAlternateNames
geneRelatedPhenotypeDatasets:
  PA129411303
  PA145069470
  ....
  PA145080521
  PA150799043
geneRelatedDrugs:
  PA128406956
  PA451986
  .....
  PA452604
geneSymbol: ABCB1
geneName: ATP-binding cassette, sub-family B (MDR/TAP), member 1
searchTerm: PA267
geneAlternateSymbols:
  ABC20
  CD243
  CLCS
  .....
  P-gp
  PGY1
geneRelatedPathways:
  Antiplatelet Drug Clopidogrel Pathway (PK): /search/pathway/platelet/platelet-clopidogrel-pk.jsp
  Codeine and Morphine Pathway (PK): /search/pathway/codeine-morphine/codeineMorphine-pk.jsp
  Doxorubicin: /search/pathway/doxorubicin/doxorubicin.jsp
  Erlotinib Pathway: /search/pathway/erlotinib/erlotinib.jsp
  .....
  Taxane Pathway: /search/pathway/taxane/taxane.jsp
  Vinca Alkaloids PK: /search/pathway/vinca-alkaloids/vinca-alkaloids-pk.jsp
  Warfarin Pathway (PK): /search/pathway/warfarin/warfarin-pk.jsp
geneRelatedDiseases:
  PA443265
  PA443355
  .....
  PA447321
geneAlternateNames:
  ABC20
  ATP-BINDING CASSETTE, SUBFAMILY B, MEMBER 1; ABCB1
  Homo sapiens ATP-binding cassette, sub-family B (MDR/TAP), member 1 (ABCB1), mRNA.
  P glycoprotein 1
  P glycoprotein 1/multiple drug resistance 1
  .....
  P-gp
  colchicin sensitivity
  doxorubicin resistance
  multidrug resistance 1

```

Figure 5.11. PharmGKB Original Return Map (partial).

<< Web Service Query Report QoS section >>  
*Includes timestamp, id of the invoked service operation,  
 response time, pre processing time and post processing  
 time.*



```
<?xml version="1.0" encoding="iso-8859-1" ?>
- <QueryReport>
  <ServiceOperationId>36</ServiceOperationId>
  <Timestamp>2008-04-23 21:23:30.796</Timestamp>
  <ResponseTime>1329</ResponseTime>
  <PostProcessingTime>0</PostProcessingTime>
  <PreProcessingTime>0</PreProcessingTime>
- <getRelatedPathwayByGene>
  <element>Antiplatelet Drug Clopidogrel Pathway (PK)</element>
  <element>Codeine and Morphine Pathway (PK)</element>
  <element>Doxorubicin</element>
  <element>Erlotinib Pathway</element>
  <element>Etoposide</element>
  <element>Gefitinib Pathway</element>
  <element>Irinotecan Pathway</element>
  <element>Irinotecan Pathway (Cancer)</element>
  <element>Methotrexate Pathway</element>
  <element>Proton Pump Inhibitor (PK)</element>
  <element>Statin Pathway (Atorvastatin, Lovastatin and Simvastatin PK)</element>
  <element>Statin Pathway (PK)</element>
  <element>Statin Pathway (Pravastatin PK)</element>
  <element>Taxane Pathway</element>
  <element>Vinca Alkaloids PK</element>
  <element>Warfarin Pathway (PK)</element>
</getRelatedPathwayByGene>
</QueryReport>
```

Figure 5.12. UBSI getRelatedPathwayByGene Operation Return Query Report.



Figure 5.13. UBSI Batch Operations Query Form.



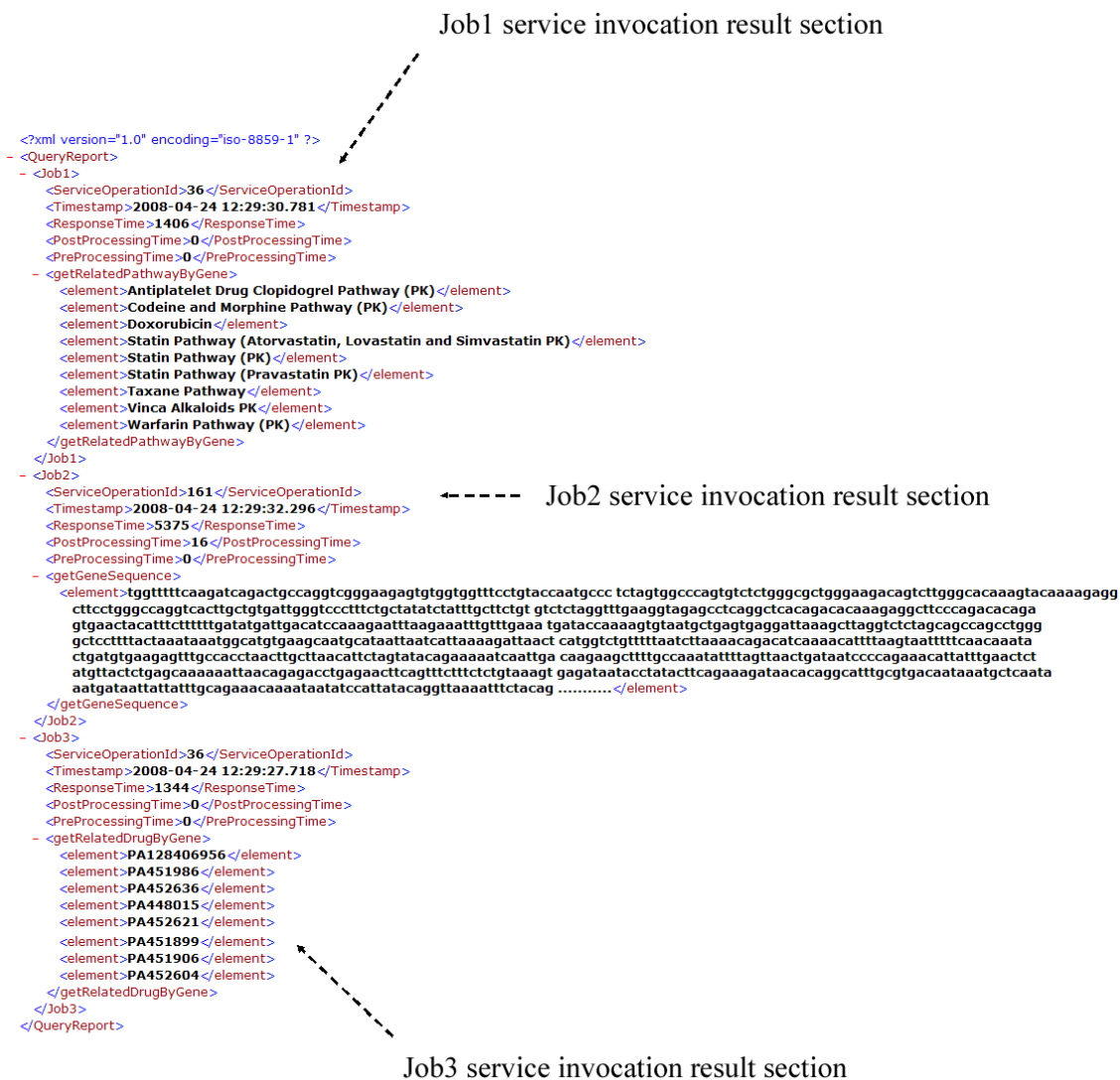


Figure 5.14. UBSI Batch Operations Return Query Report.

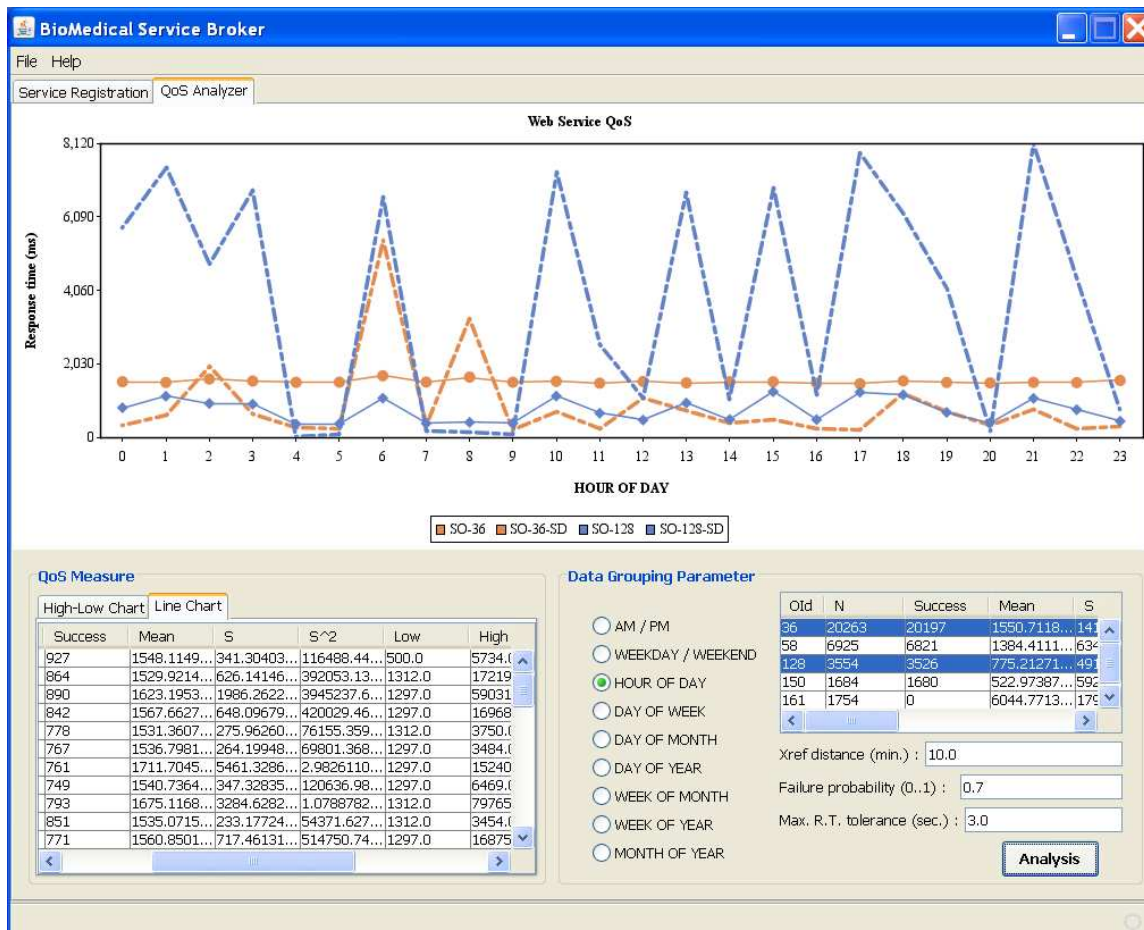


Figure 5.15. Service Operations 36 and 128 – Grouping by HOUR\_OF\_DAY.

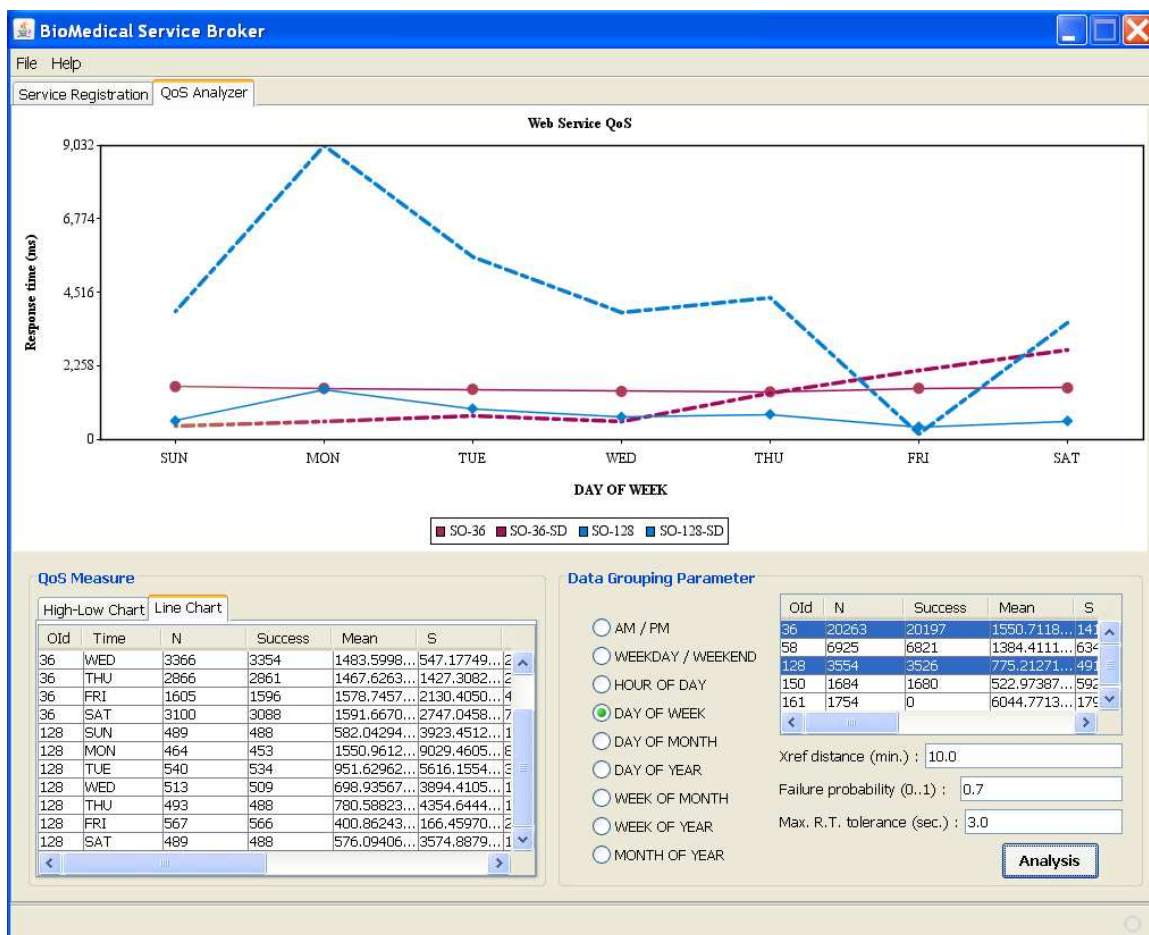


Figure 5.16. Service Operations 36 and 128 – Grouping by DAY\_OF\_WEEK.

## CHAPTER 6

### APPLICATIONS

We have introduced the BioServiceBroker system by describing each component with examples, and showed the power of its QoS service-based query processing scheme. In this chapter, we will present two examples that utilize our system as integrated service data source. In section 6.1, we describe a mediated taxonomy system that uses our broker system as a biomedical service provider to collect relevant information. In section 6.2, we discuss biomedical service workflows that can employ our system as a building block to compose a QoS-ensured workflow.

#### 6.1 Mediated Taxonomy System

Figure 6.1 demonstrates the query interface of our client mediator system [4]. The left frame shows a tree view of the ontology concepts. A user can browse the specific concept through several high level concepts: molecule, interaction, reaction, pathway, process, and so on. The rest of the more specific concepts are built up by domain relationships. Currently, we support `is_a`, `part_of`, `sequence_order`, `pathway_process`, `molecule_structure`, and OBO relationships [73].

Based on the discussion in section 5.3, we know that for the same queried concepts, there are many available service sources to provide the data instances. For example, both the NCBI and PharmGKB can provide gene information. When a concept is selected, the attributes, directly related concepts and their relationships are shown on the right frame. The user can select the attributes of interest from the current concept and/or the related concepts, and specify the weights of QoS parameters to control the quality of

**BioMediation**

Welcome | Admin | Query | Ontology Browser

**Browse the ontology concepts:**

- concept
  - molecule
    - DNA
    - hormone
  - protein
    - chaperone
    - enzyme
      - ATPase
      - dehydrogenase
      - DNA helicase
      - isomerase
      - kinase
      - oxidoreductase
    - receptor
    - transcription factor
  - RNA
  - structure
    - DNA sequence
    - lipid bilayer
    - protein domain
    - protein sequence
    - RNA sequence
  - interaction
    - bond
    - molecular interaction
  - reaction
    - enzymatic reaction
  - pathway
  - process
  - source
    - biogenic source
      - organism
        - Drosophila melanogaster
        - Escherichia coli

Please select the attributes or related concepts for the current concept: *protein*

- protein
  - name
  - organism
  - synonyms
  - structure
  - pathway
  - gene

Please set QoS control values:

Completeness: 0.1    Response Time: 0.4    Authority: 0.4    Stability: 0.05    Availability: 0.05

Submit    Clear

Associated instances:

protein_accession	protein.name	protein.organism	protein.synonyms	structure_accession	gene_accession
<a href="#">spt:P08833</a>	Insulin-like growth factor-binding protein 1 [Precursor]	Homo sapiens (Human)	IGFBP-1, Placental protein 12	<a href="#">pdb:1ZT5</a>	<a href="#">gbk:gene3484</a>
<a href="#">spt:P24591</a>	Insulin-like growth factor-binding protein 1 [Precursor]	Bos taurus (Bovine)	IGFBP-1, bIGFBP-1	<a href="#">pdb:1BOE</a>	<a href="#">gbk:gene3484</a>
<a href="#">spt:P47876</a>	Insulin-like growth factor-binding protein 1 [Precursor]	Mus musculus (Mouse)	IGFBP-1, IBP-1, IGF-binding protein 1	<a href="#">pdb:1BOE</a>	<a href="#">gbk:gene3484</a>
<a href="#">spt:P21743</a>	Insulin-like growth factor-binding protein 1 [Precursor]	Rattus norvegicus (Rat)	IGFBP-1, IBP-1, IGF-binding protein 1	<a href="#">pdb:1BOE</a>	<a href="#">gbk:gene3484</a>

Figure 6.1. Mediated Taxonomy System.

query results. For example, if a biologist wants to retrieve all the data about "IGFBP" protein, such as name, source of organism, 3D structure, and coding gene, we can first browse the concept tree to locate the "protein" concept. Then select all the attributes and related concepts. In the attribute "name", we employ filter option to construct calling parameter ("IGFBP"). In the QoS control, using sliding bar to set the desired degree of "completeness". The client query will be submitted through UBSI to invoke the corresponding operations. Currently, we already implemented the limited and high level operations, more complete operation sets will be added (see Table 5.7).

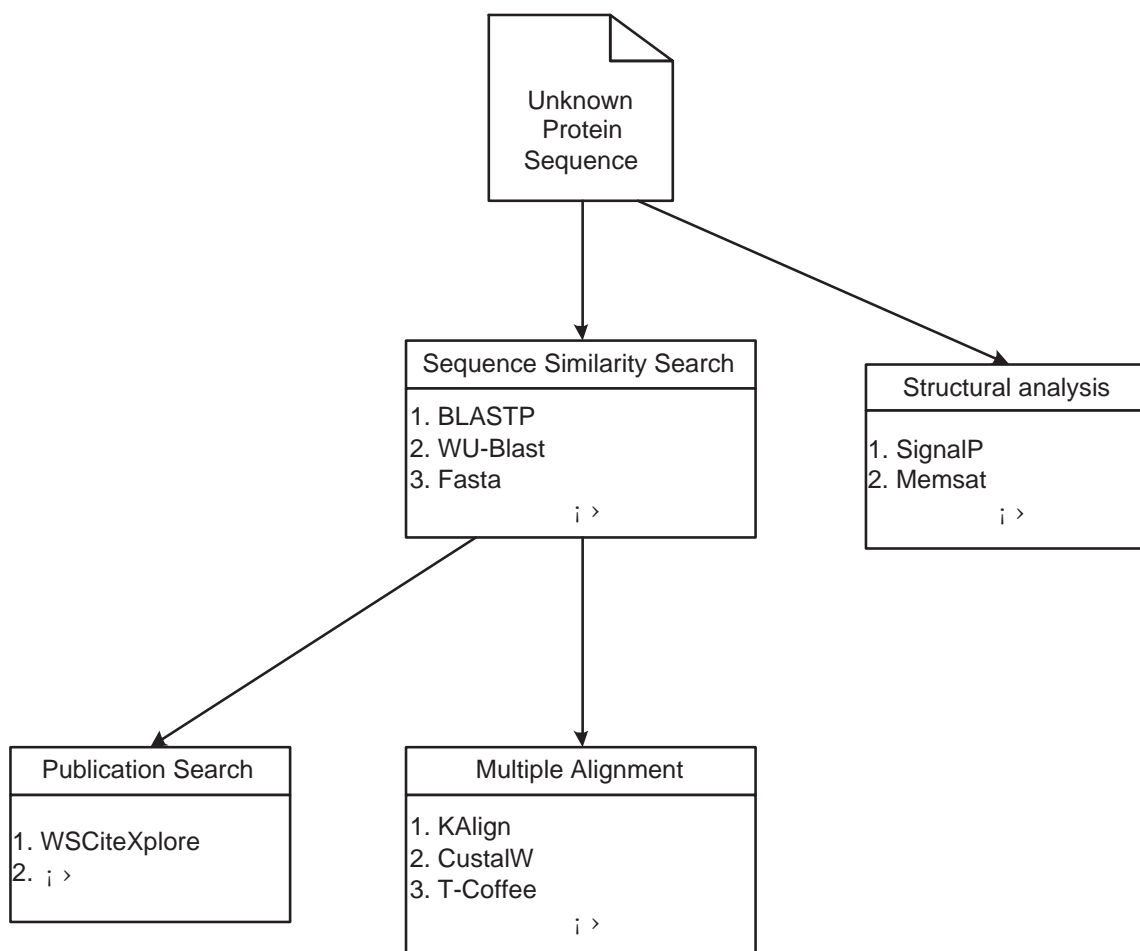


Figure 6.2. Web Service Type Workflow.

## 6.2 Biomedical Service Workflows

Figure 6.2 shows a service type workflow. This describes the interactions among several services to accomplish a particular user request of finding proteins and their related publications that are similar to a given unknown protein sequence.

**Example Workflow:** A sequence in FASTA format of an unknown protein  $p_0$  is submitted to a sequence similarity search service, which finds possible homologous sequences. One of the service instances, BLASTP, is selected as shown in Figure 6.3. The output of

BLASTP is a list of similar proteins  $p_1, p_2, \dots, p_j$ , with name, sequence, description, and e-value for each. The first hit with 100% identity is a known protein named "Dap1p", and its description is "Membrane associated progesterone receptor (MAPR)". A subsequent publication search service is invoked with the input of the keywords: Dap1p and/or MAPR. The output is a list of research papers describing some related work on this "Dap1p" protein. Also, the sequences in this protein list are submitted to a multiple alignment service, which invokes three programs: ClustalW, T-Coffee, and KAlign in parallel mode. The results are sequence alignment files, respectively. In order to compare the three different methods, the above three results are merged to produce a summary report on residue conservation in different methods. The last analysis of the unknown protein is to predict its structure. The sequence of this protein in FASTA format is submitted to a structure analysis service, which invokes two programs: SignalP and Memsat in parallel mode. The results of these two programs are also merged into a summary report on various structural features of this protein, such as signal peptide, transmembrane helices, and domains.

Figure 6.3 shows web service instances workflow. The box represents web service instances. The dashed rectangle box represents homogeneous services from the same service type. An unknown protein  $p_0$  sequence inputs to service instances: BLASTP, SignalP, and Memsat. The keyword/identifier is extracted from the output of BLASTP, and passed into CiteXplore to output the related literature and citation results. The similar protein sequences  $(p_1, p_2, \dots, p_j)$  obtained from BLASTP output are distributed to ClustalW, T-Coffee, and KAlign services. The outputs of these services are integrated by a merge tool to generate sequence alignment result. Similarly, the structural analysis results can be obtained by integrating the outputs of SignalP and Memsat services.

By introducing service relationships and enhanced QoS measures, we can create a weighted, directed service instance graph  $G = (V, E)$  with a score  $S$  on each edge

representing the QoS value for choosing a particular service operation instance (see Figure 6.4). Each vertex stands for a service operation within the service instance graph  $G$ . The score of each edge  $e_k = \langle v_i, v_j \rangle$  indicates the cost of executing service operation  $v_j$ . Each path  $p = \langle v_0, v_1, \dots, v_k \rangle$  of graph  $G$  represents an instance of a specific service workflow. The score of path  $p$  is the sum of the scores of its constituent edges (smaller score means better quality of service). We employ Dijkstra's algorithm [74] to solve this single-source shortest-paths problem on a weighted, directed graph.

```

1 function Dijkstra(Graph, source):
2   for each vertex v in Graph:
3     dist[v] := infinity
4     previous[v] := undefined
5   dist[source] := 0
6   Q := copy(Graph)
7   while Q is not empty:
8     u := extract_min(Q)
9     for each neighbor v of u:
10      alt = dist[u] + length(u, v)
11      if alt < dist[v]
12        dist[v] := alt
13        previous[v] := u
14   return previous[]

```

The result will choose the workflow instance that provides the best QoS at a particular time frame when the user submits the request.

Our approach will assist the users to create workflow types for specific biomedical applications. The workflow instances that can accomplish specific services of each type will be created automatically.



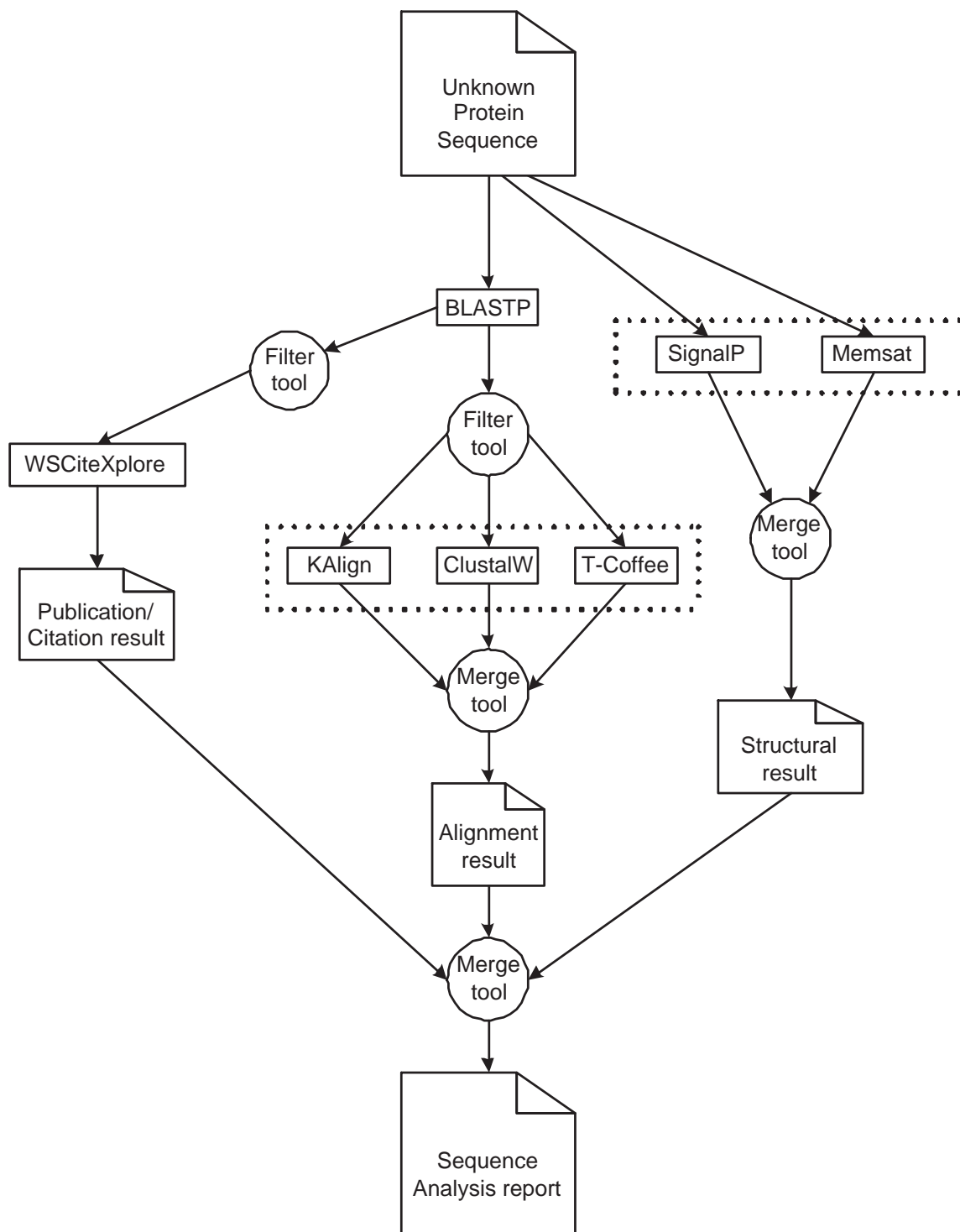


Figure 6.3. Web Service Instance Workflow.

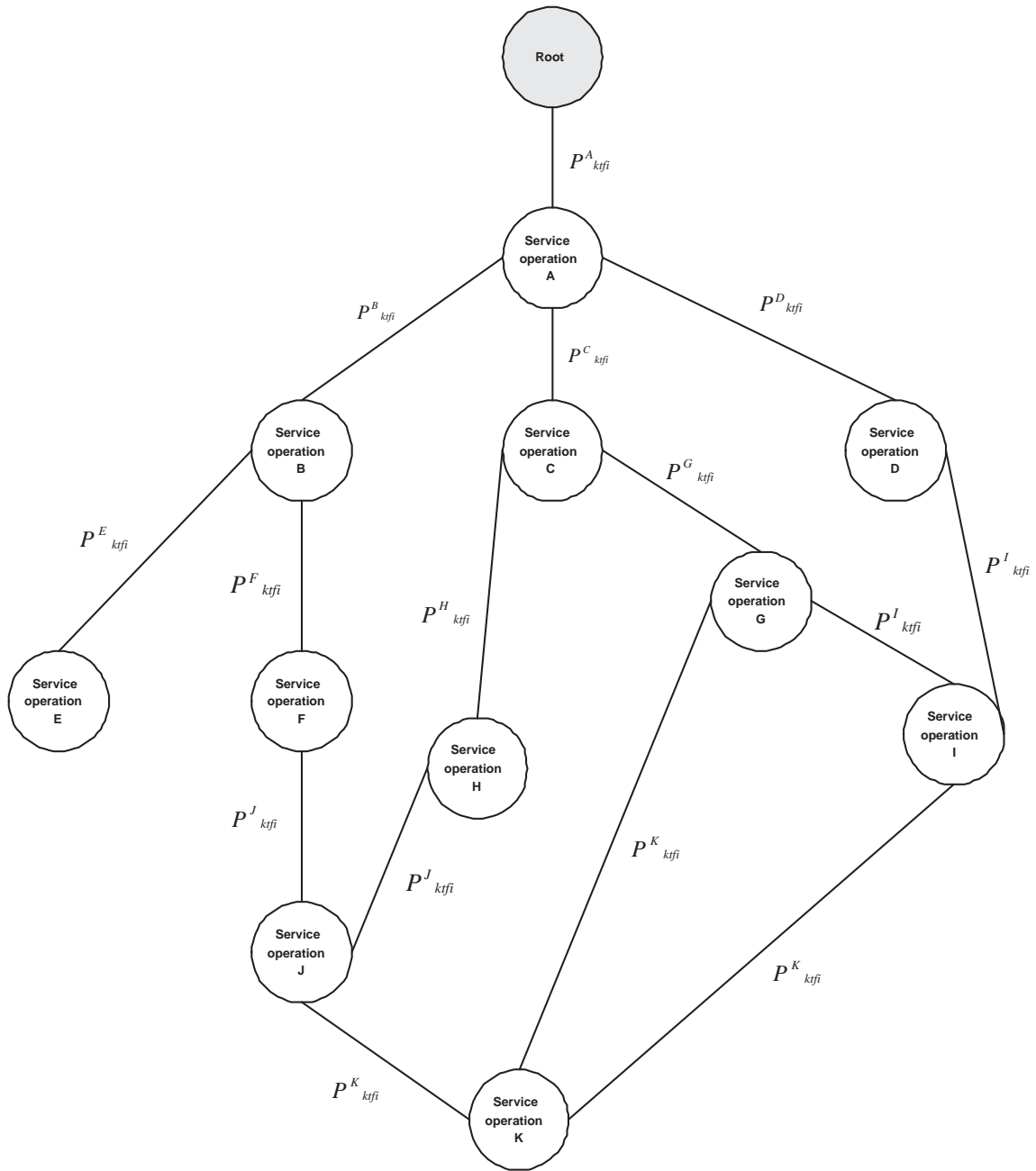


Figure 6.4. Web Service Instance Graph.

## CHAPTER 7

### CONCLUSIONS AND FUTURE DIRECTIONS

Biological and medical research creates large amounts of data spread over diverse databases such as GenBank, PDB (Protein Data Bank) [11, 12], etc., which need to be processed, integrated and organized in order to query them efficiently. Currently, many bioinformatics data and tools can be accessed/invoked through varied service operations provided by different web services, which increases the flexibility and scalability of bioinformatics data sources. This dissertation has proposed our BioServiceBroker system that integrates these bioinformatics web services to improve the service quality. We conclude our work and present future research directions in this chapter.

#### 7.1 Conclusions

This dissertation has addressed the problems of Integration in the biomedical domain from different perspectives (instance, schema, and service layers), and potential statistic bias hidden behind the existing QoS measures. We proposed a multi-level ontology-enabled service broker architecture for integrating biomedical web services to deal with above issues. Specific contributions of this dissertation are the following:

Unlike static web service providers, where the number of services provided to the end users is limited and the services are specified at design time. Our BioServiceBroker system can dynamically invoke corresponding service operations that confronts users' QoS (Quality of Service) query requirements. By utilizing our system, the application is no longer restricted to the original set of service operations that were specified and hard-coded at the design or compile time. In contrast, the capabilities of the application can be

extended at runtime. Our system can dynamically determine the best service operations chosen from candidates that can provide equivalent/homogeneous query result based on their history and current time frame.

In addition, our system utilizes enhanced QoS measures as an essential factor to determine the best service provider among service operation candidates selected from the service ontology. Each incoming service request (work load) is dynamically distributed in an efficient manner, and not only can the average response time be minimized but also the overall throughput can be increased. By utilizing our system, the web service result fusioner module is responsible to filter out the unnecessary data section, and construct a more exchangeable format before sending the result back to the requester. Our approach can reduce or eliminate the post-processing time at user/client program sides.

Our BioServiceBroker system can be used for the creation and performing of scientific web service workflows since we maintain the semantic and syntactic information for each service operation and define service relationships that can express the possible relationships between service operations. The proposed UBSI (Unified Biomedical Service Interface) is designed for achieving interoperability, reusability, and scalability across a varied base of underlying and changing services. A client side (programmer, application, etc.) can remotely invoke a service operation through UBSI, and is not required to deal with complex/inconsistent interfaces of each web service operation. The architecture can be also used with other mediator systems for cross referencing the diverse bioinformatics sources and be utilized in other application domains.

Our ultimate goal is to construct a public, scalable, interoperable biomedical service platform based on UBSI to benefit scientists in data searching and publishing.

## 7.2 Future Research Directions

The proposed architecture can be possibly improved by incorporating some features described below.

1. Design accurate QoS aggregation algorithms for the QoS Service Analyzer to handle service workflows composition.
2. Add ranking/filtering strategies for WSRF component
3. Support multi-stage data grouping constraints. Our current system supports single data grouping constraints, which can be enhanced by considering a series of single data grouping constraint to increase the flexibility of the service selection rule.
4. Automatically identify service patterns. The service patterns existing at each registered web service can be extracted as references to predict future performance by designing a pattern analysis algorithm.

## REFERENCES

- [1] J. Fu, F. Ji, and R. Elmasri, “Bioso: Bioinformatic service ontology for dynamic biomedical web services integration,” in *BIOT '07*, Colorado Springs, CO, USA, 2007.
- [2] —, “Multi-level biomedical ontology-enabled service broker for web-based inter-operation.” in *SAC '08*, Fortaleza, Ceara, Brazil, 2008.
- [3] R. Elmasri, J. Fu, and F. Ji, “Multi-level conceptual modeling for biomedical data and ontologies integration.” in *CBMS*, 2007, pp. 589–594.
- [4] F. Ji, J. Fu, R. Elmasri, N. Stojanovic, and G. Grant, “Mediated taxonomy system for bioinformatics data integration,” in *BIBM '07*, Silicon Valley, CA, USA, 2007.
- [5] T. N. Bhat, P. E. Bourne, Z. Feng, G. Gilliland, S. Jain, V. Ravichandran, B. Schneider, K. Schneider, N. Thanki, H. Weissig, J. D. Westbrook, and H. M. Berman, “The pdb data uniformity project,” *Nucleic Acids Research*, vol. 29, no. 1, pp. 214–218, 2001.
- [6] P. Bourne, H. Berman, B. McMahon, K. Watenpaugh, J. Westbrook, and P. Fitzgerald, “The macromolecular crystallographic information file,” 1995. [Online]. Available: [citeseer.ist.psu.edu/bourne95macromolecular.html](http://citeseer.ist.psu.edu/bourne95macromolecular.html)
- [7] R. Bagga, “Conceptual modeling for protein data,” *Unpublished MS thesis, The University of Texas at Arlington*, 2004.
- [8] Wikipedia, “Data integration — wikipedia, the free encyclopedia,” 2008. [Online]. Available: <http://en.wikipedia.org/w/index.php?title=Data.integration&oldid=202835722>

- [9] T. Hernandez and S. Kambhampati, “Integration of biological sources: current systems and challenges ahead,” *SIGMOD Rec.*, vol. 33, no. 3, pp. 51–60, September 2004. [Online]. Available: <http://portal.acm.org/citation.cfm?id=1031583>
- [10] F. S. Collins, E. D. Green, A. E. Guttmacher, and M. S. a. Guyer, “A vision for the future of genomics research.” *Nature*, vol. 422, no. 6934, pp. 835–847, April 2003. [Online]. Available: <http://dx.doi.org/10.1038/nature01626>
- [11] J. D. Westbrook, Z. Feng, S. Jain, T. N. Bhat, N. Thanki, V. Ravichandran, G. Gilliland, W. Bluhm, H. Weissig, D. S. Greer, P. E. Bourne, and H. M. Berman, “The protein data bank: unifying the archive,” *Nucleic Acids Research*, vol. 30, no. 1, pp. 245–248, 2002.
- [12] H. M. Berman, J. D. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne, “The protein data bank,” *Nucleic Acids Research*, vol. 28, no. 1, pp. 235–242, 2000.
- [13] P. J. Kersey, J. Duarte, A. Williams, Y. Karavidopoulou, E. Birney, and R. Apweiler, “The international protein index: An integrated database for proteomics experiments,” *Proteomics*, vol. 4, no. 7, pp. 1985–1988, 2004.
- [14] OMG, “Life sciences identifiers specification,” 2005. [Online]. Available: <http://www.omg.org/docs/dtc/04-05-01.pdf>
- [15] V. M. Markowitz, F. Korzeniewski, K. Palaniappan, E. Szeto, N. Ivanova, and N. C. Kyrpides, “The integrated microbial genomes (img) system: a case study in biological data management,” in *VLDB '05: Proceedings of the 31st international conference on Very large data bases*. VLDB Endowment, 2005, pp. 1067–1078.
- [16] V. Jakoniene and P. Lambrix, “Ontology-based integration for bioinformatics,” in *Proceedings of VLDB Workshop on Ontologies-based techniques for DataBases and Information Systems, Trondheim, Norway (2nd–3rd September 2005)*, 2005.

- [17] P. Mork, R. Shaker, and P. Tarczy-Hornoch, “The multiple roles of ontologies in the biomediator data integration system.” in *DILS*, 2005, pp. 96–104.
- [18] S. TriSZl, K. Rother, H. Muller, T. Steinke, I. Koch, R. Preissner, C. Frommel, and U. Leser, “Columba: an integrated database of proteins, structures, and annotations,” *BMC Bioinformatics*, vol. 6, no. 1, p. 81, 2005. [Online]. Available: <http://www.biomedcentral.com/1471-2105/6/81>
- [19] S. Drăghici, S. Sellamuthu, and P. Khatri, “Babel’s tower revisited: a universal resource for cross-referencing across annotation databases,” *Bioinformatics*, vol. 22, no. 23, pp. 2934–2939, 2006.
- [20] R. Elmasri and S. B. Navathe, *Fundamentals of Database Systems*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2006.
- [21] W3C, “Extensible markup language (xml) 1.0 (fourth edition),” 2006. [Online]. Available: <http://www.w3.org/TR/xml>
- [22] —, “Rdf vocabulary description language 1.0: Rdf schema,” 2004. [Online]. Available: <http://www.w3.org/TR/rdf-schema/>
- [23] A. Y. Halevy, “Answering queries using views: A survey,” *The VLDB Journal*, vol. 10, no. 4, pp. 270–294, 2001.
- [24] S. Potter and J. S. Aitken, “A semantic service environment: A case study in bioinformatics.” in *ESWC*, 2005, pp. 694–709.
- [25] T.-S. Jung and W.-S. Cho, “Spdbsw: A service prototype of spdb on the web.” in *BNCOD*. Springer-Verlag Berlin Heidelberg, Inc., 2007, pp. 49–57.
- [26] D. E. VanderMeer, A. Datta, K. Dutta, H. M. Thomas, K. Ramamritham, and S. B. Navathe, “Fusion: A system allowing dynamic web service composition and automatic execution,” in *CEC*, 2003, pp. 399–404.



- [27] P. M. Gordon, Q. Trinh, and C. W. Sensen, “Semantic web service provision: a realistic framework for bioinformatics programmers,” *Bioinformatics*, vol. 23, no. 9, pp. 1178–1180, 2007.
- [28] BioDAS, “<http://www.biodas.org/>,” 2007.
- [29] W3C, “Simple object access protocol (soap),” 2007. [Online]. Available: <http://www.w3.org/TR/soap/>
- [30] —, “Web services description language (wsdl) version 2.0 part 1: Core language,” 2007. [Online]. Available: <http://www.w3.org/TR/2007/REC-wsdl20-20070626>
- [31] A. E. Walsh, Ed., *Uddi, Soap, and Wsdl: The Web Services Specification Reference Book*. Prentice Hall Professional Technical Reference, 2002.
- [32] M. Y. Galperin, “The molecular biology database collection: 2008 update,” *Nucleic Acids Research*, vol. 36, no. suppl1, pp. D2–4, 2008. [Online]. Available: [http://nar.oxfordjournals.org/cgi/content/abstract/36/suppl\\_1/D2](http://nar.oxfordjournals.org/cgi/content/abstract/36/suppl_1/D2)
- [33] M. Keet, “Biological data and conceptual modelling methods,” *Journal of Conceptual Modeling*, vol. 29, 2003.
- [34] S. B. Navathe and U. Patil, “Genomic and proteomic databases and applications: A challenge for database technology,” in *DASF AA*, 2004, pp. 1–24.
- [35] F. Bry and P. Kröger, “A computational biology database digest: Data, data analysis, and data management,” *Distributed and Parallel Databases*, vol. 13, no. 1, pp. 7–42, January 2003. [Online]. Available: <http://dx.doi.org/10.1023/A:1021540705916>
- [36] T. N. Bhat, P. Bourne, Z. Feng, G. Gilliland, S. Jain, V. Ravichandran, B. Schneider, K. Schneider, N. Thanki, H. Weissig, J. Westbrook, and H. M. Berman, “The PDB data uniformity project,” *Nucl. Acids Res.*, vol. 29, no. 1, pp. 214–218, 2001.
- [37] S. R. Hall, “The star file: A new format for electronic data transfer and archiving,” *J. Chem. Inform. Comp. Sci.*, vol. 31, pp. 326–333, 1991.

- [38] N. Li, Y. Liu, O. A. Bukhres, and Z. Ben-Miled, “Biological database integration,” *Wiley Encyclopedia of Biomedical Engineering*, 2006.
- [39] S. Shah, Y. Huang, T. Xu, M. Yuen, J. Ling, and B. F. Ouellette, “Atlas - a data warehouse for integrative bioinformatics,” *BMC Bioinformatics*, vol. 6, no. 1, p. 34, 2005. [Online]. Available: <http://www.biomedcentral.com/1471-2105/6/34>
- [40] S. B. Davidson, J. Crabtree, B. P. Brunk, J. Schug, V. Tannen, G. C. Overton, and J. C. J. Stoeckert, “K2/kleisli and gus: experiments in integrated access to genomic data sources,” *IBM Syst. J.*, vol. 40, no. 2, pp. 512–531, 2001.
- [41] W. Fujibuchi, S. Goto, H. Migimatsu, I. Uchiyama, A. Ogiwara, Y. Akiyama, and M. Kanehisa, “Dbget/linkdb: an integrated database retrieval system,” *Pacific Symp Biocomputing*, pp. 683–694, 1998.
- [42] T. R. Gruber, “A translation approach to portable ontology specifications,” *Knowl. Acquis.*, vol. 5, no. 2, pp. 199–220, June 1993. [Online]. Available: <http://portal.acm.org/citation.cfm?id=173747>
- [43] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock, “Gene ontology: tool for the unification of biology. the gene ontology consortium.” *Nat Genet*, vol. 25, no. 1, pp. 25–29, May 2000. [Online]. Available: <http://dx.doi.org/10.1038/75556>
- [44] R. Stevens, N. W. Paton, P. Baker, G. Ng, C. A. Goble, S. Bechhofer, and A. Brass, “Tambis online: A bioinformatics source integration tool,” in *SSDBM '99: Proceedings of the 11th International Conference on Scientific on Scientific and Statistical Database Management*. Washington, DC, USA: IEEE Computer Society, 1999, p. 280.

- [45] M. Coletti and H. Bleich, “Medical subject headings used to search the biomedical literature,” *Journal of the American Medical Informatics Association*, vol. 8 (4), 2001.
- [46] J. Y. Chen and J. V. Carlis, “Genomic data modeling,” *Information Systems*, vol. 28, no. 4, pp. 287–310, 2003.
- [47] S. Ram and W. Wei, “Modeling the semantic of 3d protein structures,” in *The 23rd International Conference on Conceptual Modeling, (ER), LNCS 3288*, 2004, pp. 696–708.
- [48] M. K and L. J, *Biology*. Prentice Hall, Inc., 2008.
- [49] Wikipedia, “Cell (biology) — wikipedia, the free encyclopedia,” 2008, [Online; accessed 31-March-2008]. [Online]. Available: [\url{http://en.wikipedia.org/w/index.php?title=Cell\\_%28biology%29&oldid=200763941}](http://en.wikipedia.org/w/index.php?title=Cell_%28biology%29&oldid=200763941)
- [50] P. Benjamin, M. Erraguntla, D. Delen, and R. Mayer, “Simulation modeling at multiple levels of abstraction,” in *WSC '98: Proceedings of the 30th conference on Winter simulation*. Los Alamitos, CA, USA: IEEE Computer Society Press, 1998, pp. 391–398.
- [51] L. Hunter, Ed., *Artificial intelligence and molecular biology*. Menlo Park, CA, USA: American Association for Artificial Intelligence, 1993.
- [52] B. Smith, W. Ceusters, B. Klagges, J. Kohler, A. Kumar, J. Lomax, C. Mungall, F. Neuhaus, A. L. Rector, and C. Rosse, “Relations in biomedical ontologies,” *Genome Biology*, vol. 6, no. 5, 2005.
- [53] R. Elmasri, F. Ji, J. Fu, Y. Zhang, and Z. Raja, “Extending eer modeling concepts for biological data.” in *CBMS*, 2006, pp. 599–604.
- [54] Wikipedia, “Web service — wikipedia, the free encyclopedia,” 2008, [Online; accessed 27-April-2008]. [Online]. Available: [http://en.wikipedia.org/w/index.php?title=Web\\_service&oldid=207587490](http://en.wikipedia.org/w/index.php?title=Web_service&oldid=207587490)

- [55] D. L. Wheeler, T. Barrett, D. A. Benson, S. H. Bryant, K. Canese, V. Chetvernin, D. M. Church, M. Dicuccio, R. Edgar, S. Federhen, M. Feolo, L. Y. Geer, W. Helmberg, Y. Kapustin, O. Khovayko, D. Landsman, D. J. Lipman, T. L. Madden, D. R. Maglott, V. Miller, J. Ostell, K. D. Pruitt, G. D. Schuler, M. Shumway, E. Sequeira, S. T. Sherry, K. Sirotkin, A. Souvorov, G. Starchenko, R. L. Tatusov, T. A. Tatusova, L. Wagner, and E. Yaschenko, "Database resources of the national center for biotechnology information." *Nucleic Acids Research*, November 2007. [Online]. Available: <http://view.ncbi.nlm.nih.gov/pubmed/18045790>
- [56] A. Labarga, F. Valentin, M. Anderson, and R. Lopez, "Web services at the european bioinformatics institute." *Nucleic Acids Research*, vol. 35, no. Web Server issue, July 2007. [Online]. Available: <http://dx.doi.org/10.1093/nar/gkm291>
- [57] S. Miyazaki, H. Sugawara, K. Ieko, T. Gojobori, and Y. Tateno, "Ddbj in the stream of various biological data." *Nucleic Acids Research*, vol. 32, no. Database issue, January 2004.
- [58] M. Kanehisa, S. Goto, M. Hattori, K. F. Aoki-Kinoshita, M. Itoh, S. Kawashima, T. Katayama, M. Araki, and M. Hirakawa, "From genomics to chemical genomics: new developments in kegg," *Nucleic Acids Research*, vol. 34, no. Database issue, January 2006.
- [59] A. T. Kegg, "Genome informatics 14: 673–674 (2003) 673 kegg api: A web service using soap/wsdl to." [Online]. Available: [citeseer.ist.psu.edu/701973.html](http://citeseer.ist.psu.edu/701973.html)
- [60] J. A. Fox, S. McMillan, and B. F. F. Ouellette, "Conducting research on the web: 2007 update for the bioinformatics links directory," *Nucleic Acids Research*, vol. 35, no. suppl.2, pp. W3–5, 2007.

- [61] D. Hull, K. Wolstencroft, R. Stevens, C. Goble, M. R. Pocock, P. Li, and T. Oinn, "Taverna: a tool for building and running workflows of services," *Nucleic Acids Research*, vol. 34, no. suppl\_2, pp. W729–732, 2006.
- [62] S. Majithia, M. Shields, I. Taylor, and I. Wang, "Triana: A graphical web service composition and execution toolkit," *icws*, vol. 0, p. 514, 2004.
- [63] T. Liefeld, M. Reich, J. Gould, P. Zhang, P. Tamayo, and J. P. Mesirov, "Genecruiser: a web service for the annotation of microarray data." *Bioinformatics*, vol. 21, no. 18, pp. 3681–3682, 2005.
- [64] D. A. Menascé, "Qos issues in web services," *IEEE Internet Computing*, vol. 6, no. 6, pp. 72–75, 2002.
- [65] M. A. Serhani, R. Dssouli, A. Hafid, and H. Sahraoui, "A qos broker based architecture for efficient web services selection," in *ICWS '05: Proceedings of the IEEE International Conference on Web Services (ICWS'05)*. Washington, DC, USA: IEEE Computer Society, 2005, pp. 113–120.
- [66] M. Tian, A. Gramm, T. Naumowicz, H. Ritter, and J. Schiller, "A concept for qos integration in web services." [Online]. Available: [citeseer.ist.psu.edu/tian03concept.html](http://citeseer.ist.psu.edu/tian03concept.html)
- [67] S. Thakkar, J.-L. Ambite, and C. Knoblock, "Composing, optimizing, and executing plans for bioinformatics web services," *The VLDB Journal, Special Issue on Data Management, Analysis, and Mining for the Life Sciences*, vol. 14, no. 3, pp. 330–353, 2005.
- [68] D. Hull, R. Stevens, and P. Lord, "Describing web services for user-oriented retrieval," *W3C Workshop on Frameworks for Semantics in Web Services*, 2005.
- [69] P. Coffey, Ed., *Ontology Or The Theory Of Being: An Introduction To General Metaphysics*. Kessinger Publishing, LLC, 1918.

- [70] H. Han, “Conceptual modeling and ontology extraction for web information,” *Ph.D. dissertation, The University of Texas at Arlington*, 2002.
- [71] A. S. Tanenbaum and M. van Steen, *Distributed Systems: Principles and Paradigms*. New Jersey, NJ, USA: Prentice-Hall Publishing Co., Inc., 2002.
- [72] Wikipedia, “Central limit theorem — wikipedia, the free encyclopedia,” 2008, [Online; accessed 25-April-2008]. [Online]. Available: [http://en.wikipedia.org/w/index.php?title=Central\\_limit\\_theorem&oldid=206415373](http://en.wikipedia.org/w/index.php?title=Central_limit_theorem&oldid=206415373)
- [73] “Open biomedical ontologies.” [Online]. Available: <http://obofoundry.org/>
- [74] Wikipedia, “Dijkstra’s algorithm — wikipedia, the free encyclopedia,” 2008, [Online; accessed 25-April-2008]. [Online]. Available: [http://en.wikipedia.org/w/index.php?title=Dijkstra27s\\_algorithm&oldid=208050427](http://en.wikipedia.org/w/index.php?title=Dijkstra27s_algorithm&oldid=208050427)

## **BIOGRAPHICAL STATEMENT**

Sheng-Chieh Jack Fu was born in Taipei, Taiwan. He received his doctorate in Computer Science from University of Texas at Arlington in May 2008. He has also received M.S. degree in Computer Science from University of Texas at Arlington in 2001. His current research interests include the area of conceptual modeling for bioinformatics data/ontologies, biomedical service integration, and semantic web.